Models of Richtmyer-Meshkov Instability in Continuously Stratified Fluids

Thesis by
Mark Robert Meloon

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

1998
(Submitted November 13, 1997)

Acknowledgements

In science, interaction with colleagues is not only enjoyable and stimulating, it is oftentimes fruitful. The questions, suggestions, statements and criticisms from fellow researchers all play a part in shaping one's perspective and research philosophy. This section is an attempt to recognize those individuals who, I feel, contributed to my experience at Caltech.

Primary acknowledgement must be given to my advisor, Dan Meiron, who provided the guidance I needed to start upon a career in applied mathematics. As option representative for Applied Math, he also provided a stable environment that allowed me to concentrate on my research. By granting me the opportunity to act as a part time system administrator for the Applied Mathematics cluster of computers, he gave me the chance to expand my knowledge of the UNIX environment which, while rewarding in its own right, improved my productivity. Last, but certainly not least, he took care of the details of securing funding for my work so that I did not have to rely on a teaching assistantship.

When Ravi Samtaney accepted a postdoctoral position in Caltech's Applied Mathematics department, I was given a wonderful opportunity to interact with a respected researcher in the field of Richtmyer-Meshkov instability. Ravi did much to expand my knowledge of the field and, by constantly referring me to papers, helped introduce me to much of the important earlier work done. The calculations of initial circulation in this thesis using the Samtaney-Zabusky method were all performed using a code written by Ravi.

Numerical solutions of the Euler equations in this thesis were computed using the new CFD software package Amrita. Thanks are extended to author James Quirk for both allowing me to use Amrita before its official release date and taking time out of his busy schedule to answer questions. In addition, my numerous conversations with him helped me understand there is a difference between a well written piece of computer code and a poorly written one. I feel that this will be quite important as I work to improve my coding style in my career as an industrial applied mathematician.

There is more to life than just work and I have been fortunate to have enjoyed the friendship of several wonderful people. Special thanks must be extended to Lynda "Darna"

Gongora who was my first true friend at Caltech and provided much needed spiritual support during a very difficult and stressful first term. While everyone in Applied Mathematics has been friendly to me, my life was most enriched by those fellow students who became my friends: Imran Hashim, Mike Louie, Muruhan Rathinam, Russina Philippakos, Phil Love, Hui (pronounced "Helen") Si, and Paul LeMahieu.

Needless to say, none of this could have been done without years of encouragement and guidance from my mother, Leslee, and late father, Robert. From early on, they instilled a strong work ethic in me that has served, and will continue to serve, me well. In addition to sharing in my successes, they have also always been there to provide a strong shoulder of support for me when life got a little rough.

Final thanks is extended to Dan Meiron, Russina Philippakos and Mike Louie for taking the time to proofread an early version of this work and provide constructive criticism. This work has been supported by the Center for Advanced Fluid Dynamics Applications (CAFDA) subcontract No. B295121 under DOE Contract W-7405-ENG-48.

Abstract

The Richtmyer-Meshkov instability occurring when a planar shock wave passes through a sinusoidal region of continuous density gradient is studied numerically. Models are used to calculate the propagation of the shock through the inhomogeneity and to determine the late time behavior of the shocked fluid layer. The results from the models are compared with computations of the nonlinear Euler equations to determine ranges of validity. The models enjoy some success for problems involving weak incident shocks. As the Mach number is increased, however, the complex interactions between the transmitted and reflected fronts and the shocked density layer play an increasingly important role in the development of the flow and cause the models to fail.

The popular impulse approximation is applied to the continuously stratified fluid configuration through the use of a model due to Saffman and Meiron. The predictions of the late time growth rate of the interface and interfacial circulation from the model are compared with calculations from the nonlinear Euler equations. It is shown that for weak incident shocks the model is a very accurate prediction of the asymptotic behavior of the interface for a wide range of problems including those with interfaces of finite amplitude and thickness. For stronger shocks, post-shock values for Atwood ratio, amplitude and layer thickness are used in the model to obtain accurate predictions of late time growth rate for high Atwood ratio configurations. Poor agreement is seen for low Atwood ratios. Comparisons between circulation calculations and pointwise values of vorticity between the model and Euler simulations reveal that the impulse model does not predict the correct vorticity distribution for high or low Atwood ratios. A numerical implementation of the Biot-Savart law is used to calculate the growth rate strictly from the vorticity field in the compressible Euler simulations. The good agreement between the compressible and incompressible growth rates, as well as direct measurement of the discrete divergence in the flow, indicates that compressible effects are only important in the initialization of the instability and that the subsequent evolution is determined from the vorticity distribution. The vorticity generated by subsequent oscillations of the transmitted and reflected shocks is shown to have a non-negligible effect on the interfacial growth rate. It is conjectured that the success of the impulse approximation in predicting the asymptotic growth rate for problems involving moderate to strong shocks and high Atwood ratios is simply the result of fortuitous cancellation between regions of vorticity not computed accurately by the model.

The theory of Geometrical Shock Dynamics is used to propagate the shock through the region of density inhomogeneity and beyond. An important feature of the model is the neglect of interactions between the shock and the flow behind. A procedure for computing circulation in the entire flow from Geometrical Shock Dynamics is developed and implemented. For low to moderate Mach numbers, the initial circulation deposited on the layer is calculated with reasonable accuracy. At late times, however, the shape of the shock front does not agree with Euler calculations, resulting in incorrect calculation of the time evolution of total circulation of the flow. The agreement between the model and Euler simulations becomes poorer with increasing incident shock strength. By performing comparisons of local shock strength between the method and Euler simulation, it is shown that the method of Geometrical Shock Dynamics does not perform as well for problems involving nonuniform sound speed as had previously been believed. This suggests that the nonuniform flow conditions behind the shock, characterized by the vorticity baroclinically deposited on the interface during the shock refraction phase, plays a significant role in determining the evolution of the transmitted shock front.

Contents

List of Symbols

M_I — Mach number of incident shock

At — Initial Atwood ratio across interface

a_0 — Initial amplitude of interface

L — Characteristic length scale of thickness of density layer

M_T — Mach number of transmitted shock of one-dimensional shock-contact problem

M_R — Mach number of reflected shock of one-dimensional shock-contact problem

V — Velocity of contact surface of one-dimensional shock-contact problem

ρ_1 — Asymptotic value of initial density field as $x \to -\infty$

ρ_2 — Asymptotic value of initial density field as $x \to \infty$

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

When a shock wave passes through a region of non-uniform density, vorticity is created if the density gradient of the ambient fluid is not normal to the shock wave. An equation identifying the underlying mechanism for this creation can be derived from the two-dimensional, compressible Euler equations in the absence of body forces:

$$\frac{D}{Dt}\left(\frac{\omega}{\rho}\right) = \frac{\nabla\rho \times \nabla p}{\rho^3}. \tag{1.1}$$

This equation states that vorticity is generated whenever a pressure gradient and density gradient are misaligned. This is referred to as baroclinic generation of vorticity and happens in the above scenario because there is a pressure gradient across the shock.

Richtmyer [41] was the first to study the interaction between a shock wave and a sinusoidal interface of zero thickness separating two perfect gases of different densities. Assuming a small amplitude to wavelength ratio, Richtmyer linearized the Euler equations and solved them numerically for three cases. He found that the amplitude of the interface grew linearly with time after the initial collision. To predict the constant growth rate that the interface attained at late times, he proposed a model which has become known in the literature simply as the impulsive model. Richtmyer used the analysis of Rayleigh [40] and Taylor [59] for the motion of an interface separating two incompressible fluids of different densities under a body force and represented the effect of the shock on the interface as an impulsive force. This formulation produced an expression for the constant growth rate in terms of the wavelength of perturbation, initial amplitude, densities of the fluids and the constant velocity of the shocked interface. The amplitude of perturbation and densities on either side of the interface change discontinuously under the action of the impulsive force, and Richtmyer realized there was some ambiguity regarding which values to use. He found that the use of post-shock values in his model resulted in growth rate predictions which agreed well with those obtained from his numerical solutions of the compressible linearized Euler equations. The use of post-shock quantities was viewed as a way of accounting for compressible effects in the incompressible impulse theory and this approach subsequently became known as Richtmyer's model.

Nine years later, Meshkov [26] was the first to experimentally measure growth rates of an interface hit with a shock wave. He also observed a period of linear growth but with growth rates lower than those predicted by Richtmyer by roughly a factor of two. Meyer and Blewett [27] were the first to do numerical simulations of the full nonlinear Euler equations and found good agreement between their calculations and Richtmyer's linear theory. Whereas Richtmyer only considered situations where a shock was reflected back when the incident shock struck the interface, Meyer and Blewett also studied cases when a rarefaction was reflected. To obtain good agreement between the impulse model and the growth rates recorded from their simulations for the reflected rarefaction case, however, they needed to use the average of the pre-shock and post-shock amplitudes as the effective amplitude in Richtmyer's expression.

Meyer and Blewett chose the parameters in their simulations to model Meshkov's experiments. Meshkov had feared the discrepancy between his results and Richtmyer's was due to the fact that the initial amplitude of the interface in his experiments was not small compared to the wavelength, thus outside the range of validity of Richtmyer's model. Meyer and Blewett's computations produced growth rates which agreed with Richtmyer, however, indicating that nonlinearity was not the cause of the discrepancy. While they felt that the neglect of viscosity in Richtmyer's formula and their simulations might have been responsible for the disagreement with Meshkov's measurements, they were unable to provide an adequate explanation for the discrepancies. Since that time the Richtmyer-Meshkov instability has grown into a research field in its own right. Much of the interest is due to the detrimental effect this instability has on the efficient burn of capsules in inertial confinement fusion experiments [24]. In general, growth rates measured in experiments still do not agree with those obtained from simulation and effort is directed towards developing a predictive capability and a firm understanding of the instability itself. These discrepancies have prompted much debate [47] over the importance of experimental factors such as membranes (used to keep the gases separate prior to arrival of the shock), wall vortices, and the effects of Schlieren imaging though the boundary layers, all of which are unaccounted for in theories. Recent observations by Vetter and Sturtevant [63] have indicated that the membranes used in experiments of the Richtmyer-Meshkov instability lower the growth rate by a significant amount. It is possible that these membranes may be responsible for the disagreement in growth rates between experiments and theory.

Experiments of the Richtmyer-Meshkov instability that do not utilize membranes have been performed by Brouillette and Sturtevant [5, 6] and by Peyser *et al.* [34]. In the Brouillette-Sturtevant experiments, for example, the "interface" is, in fact, a layer of finite thickness where the two gases have begun to diffuse into each other. Growth rates can be calculated and, in principle, compared with theory. It is the aim of this thesis to examine the accuracy of some existing models for the Richtmyer-Meshkov instability occurring in diffuse layers by comparing their predictions against simulations of the nonlinear Euler equations. Once ranges of validity have been established, these theories could be used to compare with membraneless experiments, thus eliminating one possible source of discrepancies in growth rates of Richtmyer-Meshkov instabilities.

When performing numerical simulations, care must always be taken to ensure sufficient resolution of shocks and vortex sheets. Recently, techniques such as front tracking methods [20], adaptive mesh refinement [2], and parallel processing [54] have proved valuable in performing full nonlinear simulations of the Euler equations.

While Euler simulations have typically been performed for perfect gases, Samtaney and Meiron [54] have recently studied Richtmyer-Meshkov instabilities initiated by strong shock waves using the Ideal Dissociating Gas model to account for real gas effects. Attempts at the three-dimensional Richtmyer-Meshkov problem have been made, but computational limits on available resolution have kept these studies from being widely pursued.

Analytical theories and models do not suffer from computational issues but are oftentimes restricted to the linear (small amplitude) regime. The first rigorous theoretical work was performed by Fraley [15] who solved the linearized compressible Euler equations via Laplace transforms. He obtained an expression for the asymptotic growth rate of a shocked sinusoidal interface in the weak shock limit and outlined a procedure for calculating the growth for problems involving shocks of arbitrary strength. While the results in the weak shock limit agreed with Richtmyer's model, the results for stronger shocks, in general, did not. Because of the complexity of Fraley's solution, however, the impulse approximation remained a popular way of modeling the instability. Several researchers [6, 28, 29, 30, 49, 54] have used the idea of modeling the effect of the incident shock on the interface as an impulsive force to develop their own models. Other researchers have developed expressions for the growth rate using theories such as incompressible vortex models [52] and Padé approximation [72].

Recently interest in Richtymer's impulse model has been rekindled as examples of incorrect predictions of growth rate have been found. Yang *et al.* [69] compared simulations of the linearized Euler equations with Richtmyer's model and identified parameter ranges where the model gives incorrect results for the growth rate. Mikaelian [31, 32] has shown cases where the model predicts no growth of the shocked interface (the so called "freeze-out" phenomenon) but simulations show definite growth. Conversely, situations where the model predicts growth when simulations show none can be found. Velikovich [62] developed an analytical theory for the reflected rarefaction case. He also pointed out that the Richtmyer-Meshkov instability occurs for general Riemann problems not involving shocks or density gradients, and used this to argue that the use of the impulsive approximation to model Richtmyer-Meshkov instability as a "shock-excited Rayleigh-Taylor instability" is incorrect. Most recently, Wouchuk and Nishihara [68] have shown that the impulsive approximation is not needed to obtain the correct growth rate in the weak shock limit. They have also pointed out that the existence of vorticity in the fluid away from the interface violates the assumption of irrotational flow implicit in the Richtmyer model. By taking this extra vorticity into consideration, and assuming incompressible flow, they have obtained an approximate expression for the late time growth rate for problems involving shocks of arbitrary strength.

Relatively little work has been done to study the Richtmyer-Meshkov instability for the case of diffuse interfaces with a continuous density distribution. Brouillette and Sturtevant [5, 6] have performed experiments for diffuse interfaces and found the finite thickness of the interface to drastically reduce the growth rate. Some theory has been developed and simulations have been carried out for the Richtmyer-Meshkov instability occurring in continuously stratified fluids [6, 30, 36, 49]. However, the models have not been tested throughly. The development of analytical theories is complicated by the fact that no complete solution of the one-dimensional problem of shock propagation into a region of continuous density gradient exists, although an approximate theory has been developed by Chisnell [10]. De-

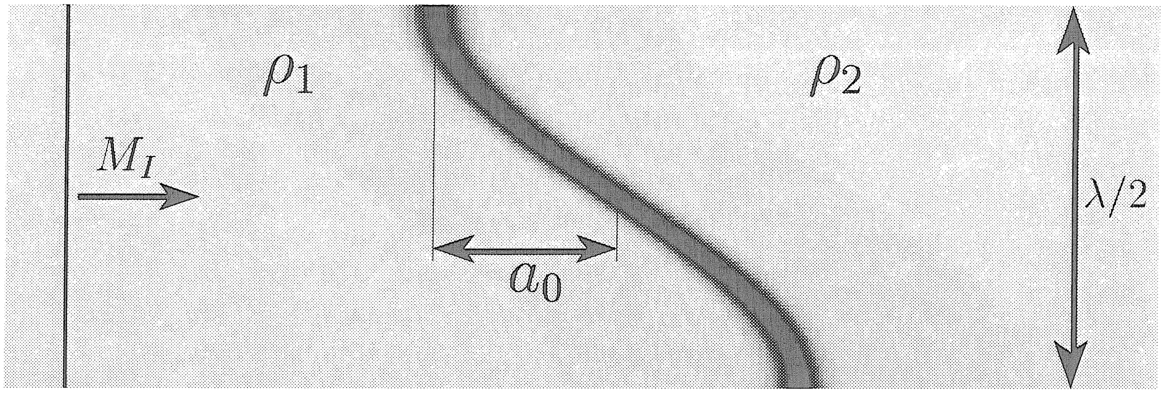


Figure 1.1: Richtmyer-Meshkov problem for a diffuse sinusoidal interface.

velopment of an analytic theory of small amplitude Richtmyer-Meshkov instability for the case of a diffuse interface by linearizing about the one-dimensional problem, as is typically done for the classical problem, is not feasible. This work seeks to extend and verify existing models so they may be used to compare with experiments, such as those done by Brouillette and Sturtevant. In comparing simplified models against simulations of the Euler equations, a better understanding of what effects are important in the instability can be achieved.

1.2 Development and Characteristics of the Instability

A schematic of the problem in its initial state is shown in figure 1.1. The picture is a computer generated Schlieren-type image where high density gradients are visualized as dark regions. A description of the algorithm used to create these Schlieren images is presented in Appendix B. The problem consists of half a sinusoidal layer of large density gradient at rest with a planar shock wave incident from the left. The boundaries along the top and bottom of the domain are reflecting, slip boundaries. These act as solid walls (with no boundary layers) or lines of symmetry in the fully periodic problem. The density layer has a characteristic thickness length scale of L and is a region of continuous, non-negligible density gradient whose mid-line is a cosine wave with initial amplitude a_0 and wavelength $\lambda = 2\pi$.

The fluid ahead of the incident shock is a perfect gas of uniform pressure and ratio of specific heats. Recent work [54] has shown that inclusion of real gas effects in simulations of hypervelocity Richtmyer-Meshkov instabilities does not alter the values of diagnostic quantities of the flow (such as interfacial growth rate and circulation) significantly. The density field ahead of the shock is continuous and asymptotes to ρ_1 to the left of the layer and ρ_2 to the right. The problem is determined by four parameters: the Mach number of the incident shock, M_I, the ratio of initial amplitude to wavelength, a_0/λ, the ratio of layer thickness to wavelength, L/λ, and the Atwood ratio across the diffuse interface, At, defined as:

$$At = \frac{\rho_2 - \rho_1}{\rho_2 + \rho_1}. \tag{1.2}$$

The state of the fluid ahead of the shock is specified by the uniform pressure, $p = 1$, the uniform ratio of specific heats, $\gamma = 1.4$, zero flow velocity and a density field of

$$\rho(x, y) = \frac{1}{2}\left[1 + At \tanh\left(\frac{x - x_{\mathrm{dl}} - a_0 \cos(y)}{L}\right)\right], \tag{1.3}$$

where x_{dl} is the x-coordinate about which the mid-line of the layer is centered.

The time evolution of the instability is depicted in a sequence of pictures in figure 1.2. As the incident shock moves through the density layer, part of the shock wave is transmitted and a wave is reflected back. The reflected wave can be either another shock wave, as shown in this picture, or a rarefaction. When the ratio of specific heats, γ, is constant, the condition for reflected rarefaction is $At < 0$ (shock moving into less dense media) and that for reflected shock is $At > 0$. When γ is not constant the character of the reflected wave is not determined strictly by the Atwood ratio [69]. Certain parameter ranges can even lead to complete transmission of the shock with no reflected wave generated.

The pressure gradient across the incident shock and density gradient across the layer interact to create vorticity in the layer as described by equation (1.1). This misalignment of pressure and density gradients is often referred to as "baroclinic generation" of vorticity.

In figure 1.2 both the transmitted and reflected shocks undergo Mach reflection from the top and bottom boundaries. The reflected shock of each Mach system sweeps across the shocked layer as the flow evolves. Vortex sheets are always present at each three-shock intersection – the so-called "triple-points" – and are sources of additional vorticity in the flow field. Shocks are super-stable structures and the transmitted and reflected shocks oscillate in a damped fashion to reduce their perturbations [42]. This oscillation creates

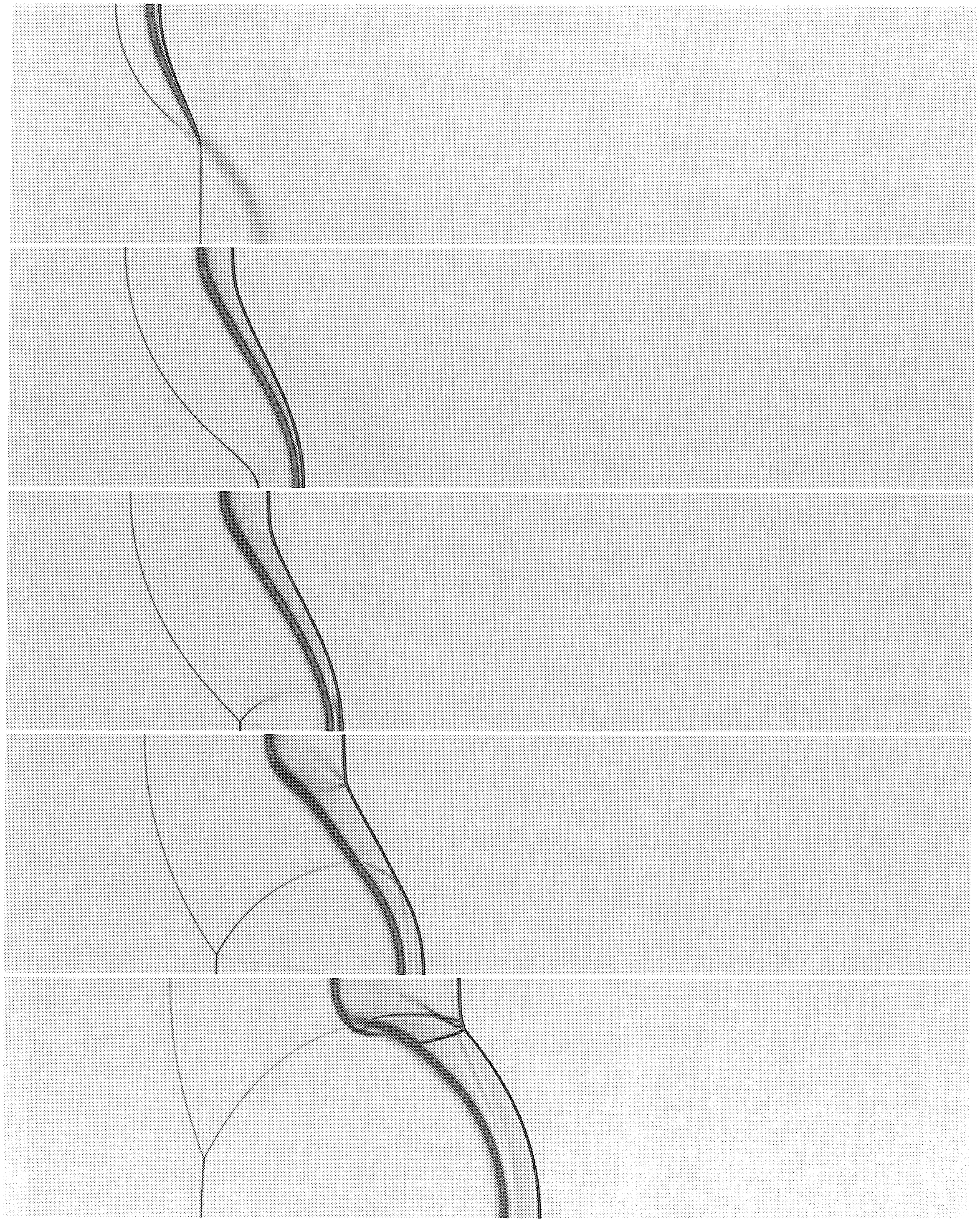


Figure 1.2: Time evolution of the Richtmyer-Meshkov instability. (Continued on next page. $M_I = 4$, $At = 0.8$, $a_0 = 1$, $L = 0.1$)

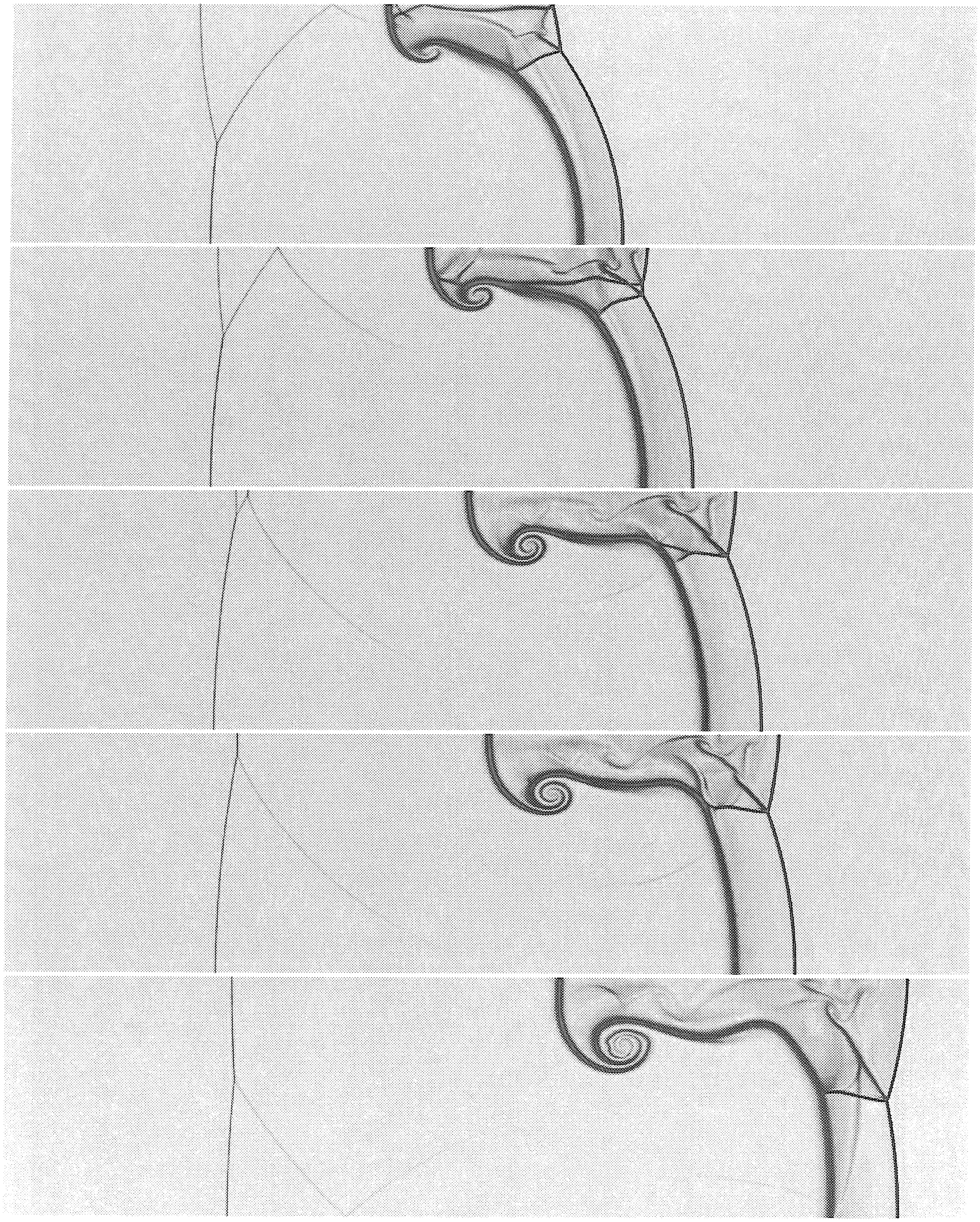


Figure 1.2 continued.

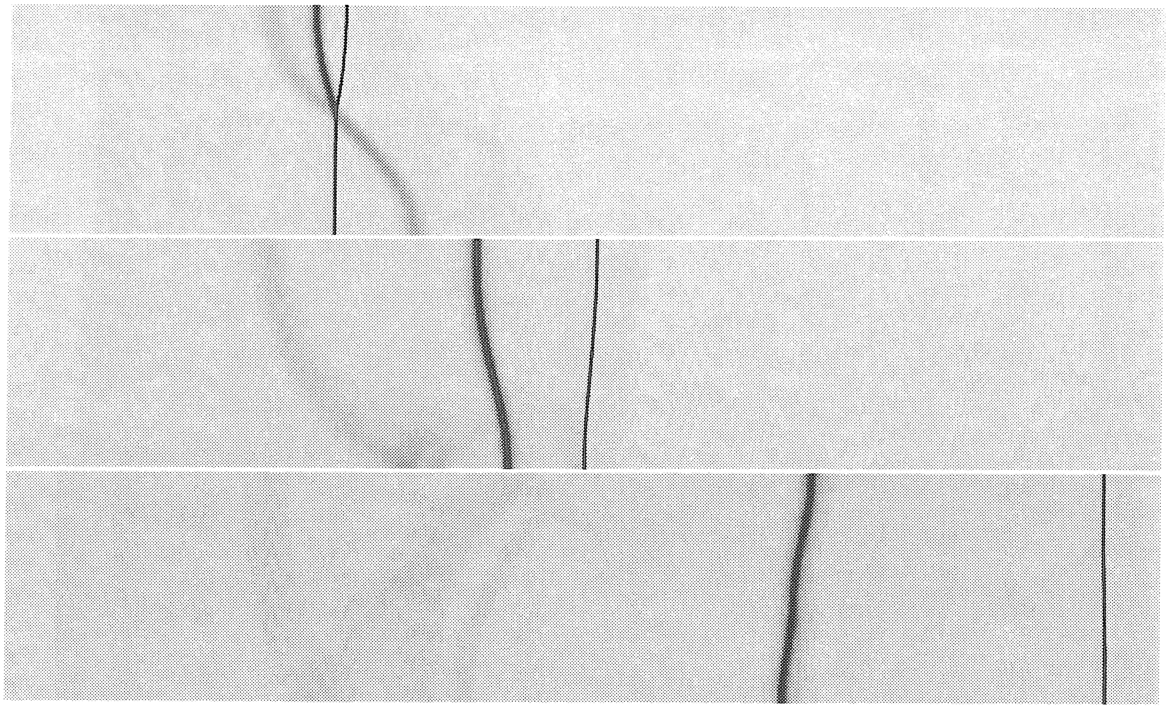


Figure 1.3: Phase reversal of interface for negative Atwood ratio. A rarefaction is reflected from the density layer. ($M_I = 2$, $At = -0.2$, $a_0 = 1$, $L = 0.1$)

pressure waves in the flow behind the shocks and modifies the growth of the interface [41]. Even as the transmitted and reflected shocks move away from the layer, they continue to modify the vorticity distribution on it through additional $\nabla\rho \times \nabla p$ interactions resulting from the pressure perturbations and reflected shock waves from triple points striking the layer. At late times the interface begins to roll up due to the vorticity on it.

When the Atwood ratio is negative, and the shock moves from a region of more dense fluid into that of less dense, a rarefaction is reflected from the layer rather than a shock. In addition, the shocked layer changes phase, as shown in figure 1.3. In the first Schlieren image of the sequence, the transmitted shock has greater velocity than the incident shock, due to the greater sound speed to the right of the layer, and moves ahead of the incident shock below it. The shock emerges from the layer with the peak along the top boundary and trough along the bottom, in contrast to the shock shown in figure 1.2 for a positive Atwood ratio. The amplitude of the interface decreases to zero and continues its motion, causing the interface to undergo phase reversal. The amplitude then continues to increase without bound. The phase reversal of the interface is due to the action of vorticity. In the

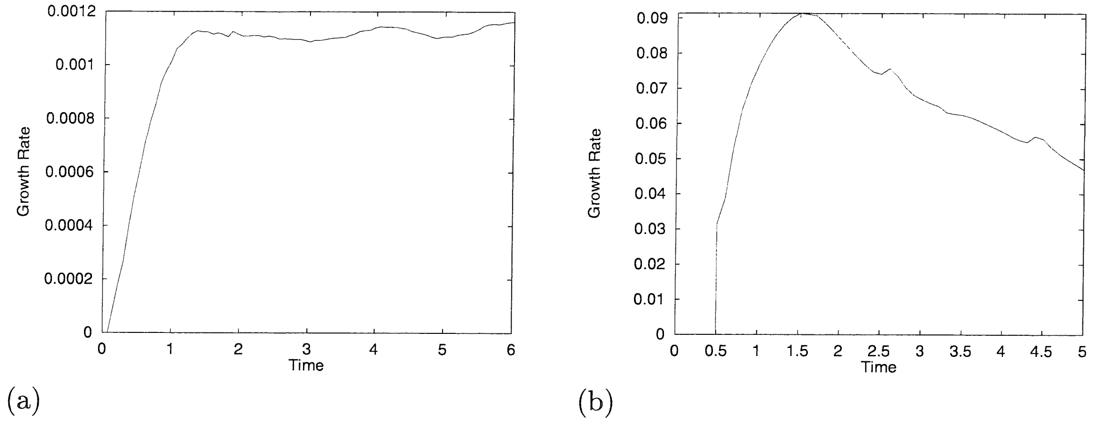
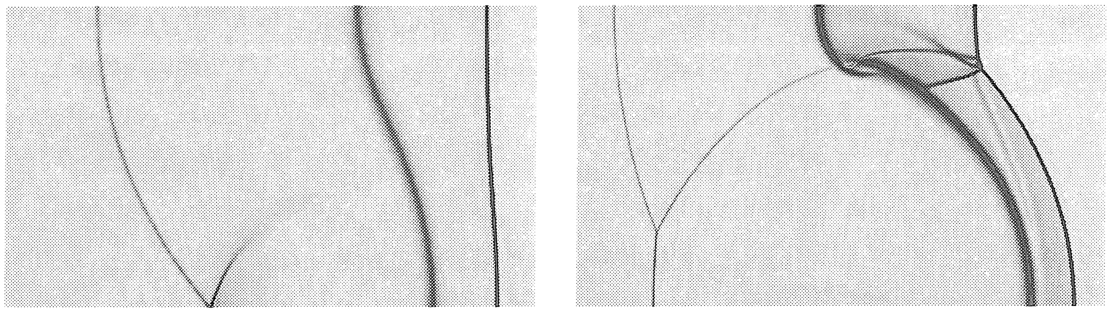


Figure 1.4: Growth rate for two problems differing only in their initial amplitudes. The run with larger amplitude does not have a period of linear growth. ($M_I = 2$, $At = 0.2$, $L = 0.1$, and (a) $a_0 = 0.01$ and (b) $a_0 = 1$)

problem shown in figure 1.3 the interaction between the incident shock and density layer has deposited negative vorticity in the layer, in contrast to the positive vorticity created in the problem shown in figure 1.2 for a positive Atwood ratio.

The initial amplitude affects both the magnitude and time evolution of the growth rate. Richtmyer [41] showed that for small amplitudes the growth of the amplitude is linear in time. For large amplitudes, the growth rate is not constant but decreases with time because much of the kinetic energy of the flow is used to distort the interface (to create roll-up structures, for example) rather than simply increase the amplitude. A comparison of growth rate for two problems, differing only in their initial amplitude, is shown in figure 1.4, using the scaling parameters (3.15) to non-dimensionalize the growth rates and times. The initial amplitude also determines the magnitude of vorticity deposited in the layer through equation (1.1). The larger the amplitude, the greater the angle between the pressure and density gradients. Note the differing order of magnitude of the values of growth rate in figure 1.4.

The initial amplitude, Atwood ratio, and Mach number also determine whether shocks are reflected back towards the shocked density layer. A comparison of Schlieren images between two problems differing only in Atwood ratio is shown in figure 1.5. The reflected shock in the $At = 0.2$ case is undergoing regular reflection but the reflected portion of that front is quite weak. In the $At = 0.8$ case the reflected shock undergoes Mach reflection from



(a) (b)

Figure 1.5: Secondary shock waves can form in the region between the primary shocks and the density layer. ($M_I = 4$, $a_0 = 1$, $L = 0.1$ and (a) $At = 0.2$ and (b) $At = 0.8$)

the bottom wall and the reflected shock from the Mach system intersects the layer. The transmitted shock undergoes Mach reflection from the top wall in the $At = 0.8$ case, and both the reflected shock and the vortex sheet from the triple point are clearly visible in the figure. No reflection occurs along the transmitted shock in the $At = 0.2$ case. The initial amplitude and incident shock Mach number will also determine what kind of reflection, if any, the transmitted and reflected shocks undergo along the top and bottom walls. Shock reflections do not typically occur for problems with small amplitude interfaces.

Little work has been done to determine the effect which the characteristic length scale of the thickness of the density layer, L, has on the development of the instability. Previous work [6, 30, 36, 49] has shown that increasing the thickness of the layer reduces the growth rate of the interface. The values L=0.01, 0.1, and 1 correspond to thin, medium, and very thick interfaces as seen in figure 1.6.

1.3 One-Dimensional Shock-Contact Problem

In this section we examine the solution of the one-dimensional problem of shock-contact interaction shown in figure 1.7. The solution will provide us with parameters needed in future sections. A derivation of the general problem can be found in [69]. Here, we restrict our analysis to the case of a reflected shock wave.

The analogous problem where the contact discontinuity is replaced by a layer of continuous density gradient has been treated by Chisnell [10]. The solution is significantly more complicated than that for the shock-contact problem due to the infinite set of wave reflec-

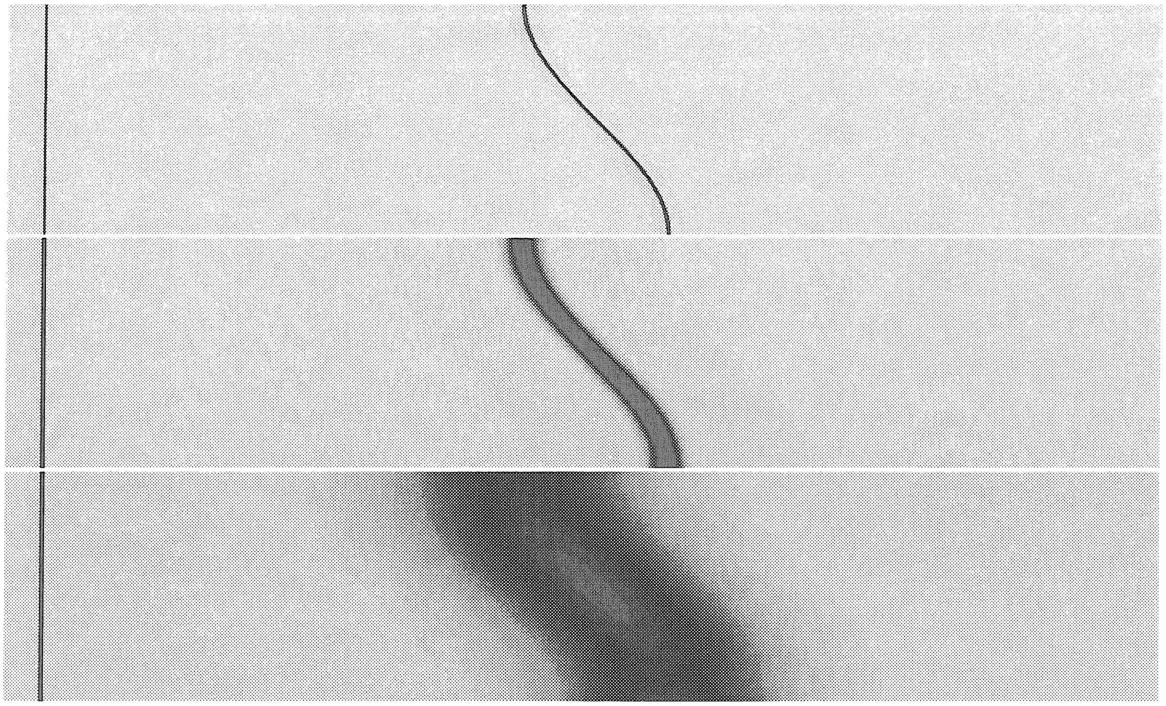


Figure 1.6: Parameter L=0.01, 0.1 and 1 represents a thin, medium and very thick initial density layer.

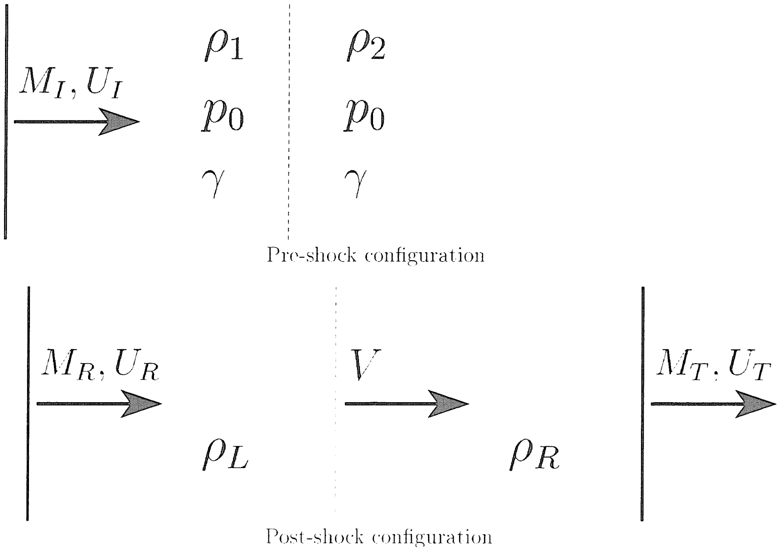


Figure 1.7: Schematic of the one-dimensional shock-contact problem. The contact discontinuity is drawn with a dashed line.

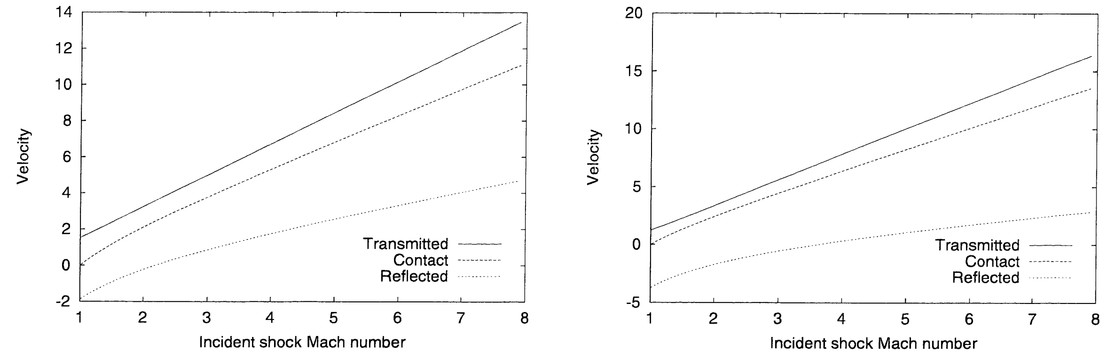
tions within the layer. At very large distances from the layer, the solution to that problem closely approximates the solution to the shock-contact problem so we do not consider it here. Interested readers are referred to the thesis of Pham [35] for a numerical solution based on Chisnell's technique.

As a shock of Mach number M_I with velocity U_I strikes a contact discontinuity, a shock of strength M_T with velocity U_T is transmitted into the second fluid, the contact acquires a constant velocity V, and a wave is reflected. In general, the reflected wave can be either a shock or a rarefaction. The nature of the wave depends on the ratio of specific heats, γ, and initial densities on either side of the contact. The exact criteria have been given by Yang *et al.* [69]. We restrict our attention to the case where the values of γ for both fluids are the same and a shock is reflected. The conditions for generation of a reflected shock then simplify down to $At > 0$, where At is defined by equation (1.2), and we denote the Mach number of the reflected shock by M_R and its velocity as U_R. Knowing the structure of the solution allows us to solve the problem by matching the pressure and velocity at the contact and using the Rankine-Hugoniot conditions at the shocks. The solution requires finding the roots of a nonlinear equation which we perform numerically using the bisection method.

The velocities of the transmitted and reflected shocks and the contact discontinuity are plotted against the incident shock strength in figure 1.8 for two Atwood ratios. For weak incident shocks the velocity imparted to the contact is very small. For stronger shocks, the velocity of the contact is much closer in value to the transmitted shock than the reflected. For incident shocks of weak and moderate strengths, the velocity of the reflected shock is negative.

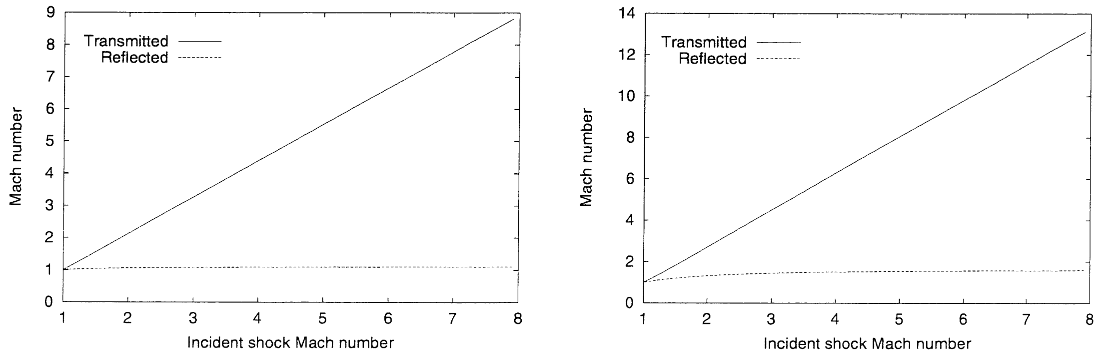
The Mach numbers of the transmitted and reflected shocks, M_T and M_R respectively, are plotted versus incident shock strength M_I in figure 1.9. The transmitted shock is much stronger than the reflected shock for both Atwood ratios. While the increase in M_T is nearly linear with M_I, the reflected shock strength M_R asymptotes to a constant value (which depends on At and γ) as $M_I \to \infty$.

The densities of the fluids on either side of the shocked contact, ρ_L and ρ_R, will differ from their initial values, ρ_1 and ρ_2, resulting in a change in Atwood ratio across the contact. The post-shock Atwood ratio, $At_{\text{post}} = (\rho_R - \rho_L)/(\rho_R + \rho_L)$, is plotted in figure 1.10 for both an initial Atwood ratio of $At = 0.2$ and $At = 0.8$. The post-shock Atwood ratio is always



(a) (b)

Figure 1.8: Velocities of shocks and contact discontinuity in the post-shock state of one-dimensional shock-contact problem. (a) $At = 0.2$, (b) $At = 0.8$



(a) (b)

Figure 1.9: Mach numbers of the transmitted and reflected shocks, M_T and M_R respectively, in the one-dimensional shock-contact problem. (a) $At = 0.2$, (b) $At = 0.8$

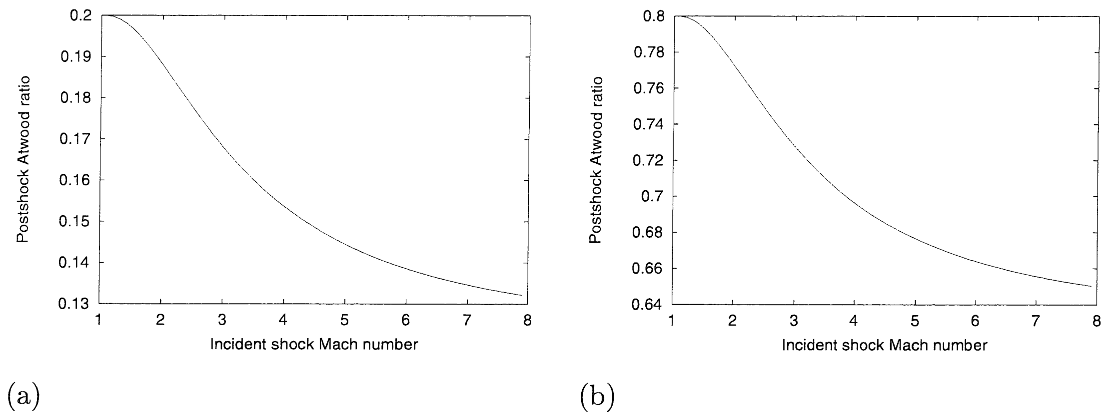


Figure 1.10: Post-shock Atwood ratios across the contact in the one-dimensional shock-contact problem. (a) $At = 0.2$, (b) $At = 0.8$

smaller than the pre-shock ratio with greater reduction occurring for stronger shocks. In the weak shock limit, $M_I \to 1$, the post-shock Atwood ratio approaches the pre-shock value.

1.4 The Richtmyer Impulsive Model

The most popular model used in the field of Richtmyer-Meshkov instability was proposed by Richtmyer himself in the very first paper on the subject [41] and has become known in the literature simply as "the impulsive model." The interaction between a shock and a sinusoidal interface of zero thickness separating two perfect gases is modeled as an impulsive force acting on the interface. The starting point of the model is the analysis of Rayleigh [40] and Taylor [59] for two incompressible, irrotational fluids separated by a sinusoidal interface of amplitude $a(t)$ being acted upon by some force $g(t)$ as shown in figure 1.11. Assuming the amplitude of the interface to be small compared to the wavelength, $ka \ll 1$, and neglecting viscosity and surface tension, the Euler equations are linearized about the unperturbed state and a dynamic equation can be obtained for the amplitude:

$$\ddot{a}(t) = kg(t)a(t)At, \tag{1.4}$$

with At given by equation (1.2) for the configuration shown in figure 1.11. If the force is constant over time (i.e.: gravity), $g(t) = g_0$, this expression can be integrated once to obtain

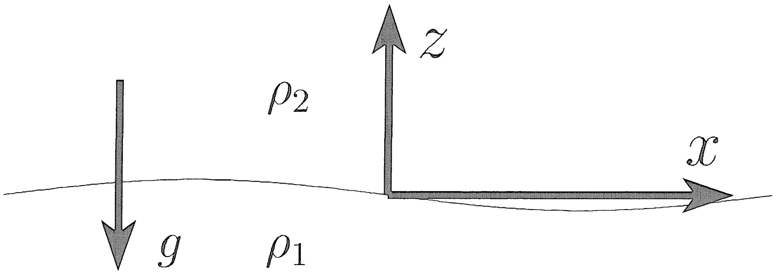


Figure 1.11: Configuration of Rayleigh-Taylor instability.

an expression for the growth rate of the interface:

$$\dot{a}(t) = ina_0e^{int}, \tag{1.5a}$$

$$n = -\sqrt{-g_0kAt}. \tag{1.5b}$$

The interface becomes unstable and grows exponentially if the less dense fluid is accelerated into the more dense. If the more dense fluid is accelerated into the less dense, the interface is stable and oscillates with fixed amplitude. This is the well-known Rayleigh-Taylor instability.

Richtmyer's idea was to consider the interaction between a shock wave and an interface of small amplitude not under the influence of any body forces as a "shock excited Rayleigh-Taylor" instability by using an impulsive force in the equation of motion of the interface. Denoting the change in velocity of the undisturbed interface due to shock acceleration as Δv, the forcing term in equation (1.4) becomes $g(t) = \Delta v\delta(t)$ where $\delta(t)$ is the Dirac delta distribution and $t = 0$ is the time of collision. Using this in equation (1.4) and integrating results in an expression for the time derivative of the amplitude:

$$\dot{a}_{\text{Richt}} = ka(0)\Delta vAt(0). \tag{1.6}$$

The result differs from the Rayleigh-Taylor growth rate in two important regards. The first is that the growth rate is linear rather than exponential. The second is that while a reduction in amplitude occurs initially when $\Delta vAt(0) < 0$, the amplitude decreases to zero and then becomes negative, resulting in a phase reversal of the interface, as in figure 1.3. The absolute value of the amplitude continues its constant growth until the linear theory is no longer valid. Thus, there is no stable configuration where the amplitude remains bounded for all time as there is for the Rayleigh-Taylor instability, the case of constant acceleration.

The impulsive approximation can be thought of as a way of computing the shock-interface interaction without actually resolving the details of the shock refraction process. The growth rate (1.6) is a prediction of the asymptotic value of the time dependent growth rate when the transmitted and reflected fronts have moved far enough away from the interface that they no longer play a significant role in its evolution and the resulting flow is almost incompressible.

Richtmyer compared the predictions of the model to the growth rate obtained from numerical simulation of the linearized Euler equations. This calculation utilized finite differences in the fluid regions between the transmitted and reflected shocks and the interface. Boundary conditions at the shocks were determined from the shock jump conditions and those at the interface from continuity of pressure and normal velocity. Richtmyer only considered cases where a shock was reflected. Richtmyer realized there was some ambiguity regarding the exact value of amplitude and Atwood ratio to use in equation (1.6). These values change instantaneously at $t = 0$ and it is not obvious what values to use. Performing three runs by feeding data tapes into the first real computer, the MANIAC, and computing until storage requirements for the simulation exceeded machine capacity, he found that use of post-shock values for amplitude and Atwood ratio for $a(0)$ and $At(0)$ in (1.6) resulted in a relative error in growth rate of only 5-10%. Use of pre-shock values resulted in predictions off by roughly a factor of two. The post-shock Atwood ratio, At_{post}, can be computed from the solution to the one-dimensional shock-contact problem described in §1.3. Richtmyer determined the post-shock amplitude from kinematic arguments with the assumption of small amplitudes and obtained the expression

$$a_{0,\text{post}} = a_0 \left(1 - \frac{\Delta v}{U_I}\right), \tag{1.7}$$

where U_I is the velocity of the incident shock and a_0 is the pre-shock amplitude. Similar expressions were obtained for the post-shock amplitudes of the transmitted and reflected shocks. Meyer and Blewett [27] later performed full nonlinear numerical simulations of Richtmyer-Meshkov instability including cases where a rarefaction was reflected rather than a shock. They found in order to get good agreement between their growth rate results and Richtmyer's model, they had to use the average of the post-shock and pre-shock amplitudes as $a(0)$ in (1.6) when the reflected wave was a rarefaction.

The impulsive approximation became a standard tool of researchers in the field of Richtmyer-Meshkov instability throughout the 1980s and into the 1990s. Recently, the model has been the subject of new investigations [31, 32, 62, 69] revealing that its predictions of growth rate can sometimes be inaccurate. In the weak incident shock limit, however, Fraley [15] has shown that (1.6) is the first order approximation to the true asymptotic growth rate from the linearized Euler equations for cases when a shock is reflected. In the weak limit, the pre-shock and post-shock values of amplitude and Atwood ratio are the same so there is no ambiguity regarding which to use.

The model has been quite popular because of its simplicity, its accuracy in computing growth rate for certain parameter ranges, and its intuitive explanation of the instability. This explanation, however, has recently been called into question by Velikovich [62], who, like Li *et al.* [23], has found interfacial instabilities that closely resemble the Richtmyer-Meshkov instability resulting from expansion waves rather than shocks. Wouchuk and Nishihara [68] have shown the impulsive approximation is not necessary to obtain the correct growth rate in the weak shock limit, and, further, the assumption of irrotationality implicit in the impulse model is not accurate for stronger shocks. Fraley has claimed the use of post-shock values of amplitude and Atwood ratio in equation (1.6) results in accurate predictions of the growth rate in some cases simply because the post-shock values are less than pre-shock values (see equation (1.7) and figure 1.10). The true compressible growth rate obtained from solutions of the linear Euler equations is typically less than the incompressible result (equation (1.6) using pre-shock values) and this reduction sometimes matches up with the reduction resulting from using postshock values.

Since Richtmyer's paper, the impulsive approximation has been used by several different researchers. Brouillette and Sturtevant [6] used the idea to adapt a model for Rayleigh-Taylor instabilities with diffuse interfaces to the Richtmyer-Meshkov problem to allow comparison with their experiments. Mikaelian [28] used the impulsive approximation to obtain results for amplitude growth for an arbitrary number of fluid layers subject to shock acceleration and, later, to continuous density profiles [30]. Mikaelian [29] also used the impulse idea to adapt experimental results for Rayleigh-Taylor instabilities to developing an analytical expression for the mix width of turbulence resulting from the Richtmyer-Meshkov instability. The extension of the impulse model to flows with equilibrium dissociation chemistry has recently been done by Samtaney and Meiron [54]. Pham and Meiron [36] used an impulse model due to Saffman and Meiron [49] to generate the initial condition for nonlinear incompressible simulations of the instability. The Saffman-Meiron model will be discussed in detail in §3.1.

Chapter 2

Numerical Solution of the Compressible Euler Equations

In this thesis, the accuracy of simplified models of the Richtmyer-Meshkov instability for diffuse layers is judged by comparing results against those obtained from numerical solutions of the full nonlinear Euler equations. This chapter discusses specifics of the numerical methods used to perform the Euler simulations. It is assumed that the reader has some familiarity with the basics of numerical solution of systems of nonlinear conservation laws such as the material covered in the book by LeVeque [22]. This chapter discusses the specific techniques used, details of, and warnings about numerical methods used to compute the solutions. Euler simulations were performed using the new computational fluid dynamics software package *Amrita*.

2.1 Details of Discretization

The compressible Euler equations of fluid dynamics in two dimensions are:

$$\mathbf{U}_t + \mathbf{F}(\mathbf{U})_x + \mathbf{G}(\mathbf{U})_y = 0, \tag{2.1}$$

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix} \qquad \mathbf{F}(\mathbf{U}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ u(\rho + E) \end{bmatrix} \qquad \mathbf{G}(\mathbf{U}) = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ v(\rho + E) \end{bmatrix}$$

where E is the total energy and is related to the pressure, p, by $p = (\gamma - 1)(E - \frac{1}{2}\rho(u^2 + v^2))$. In a numerical solution the continuous problem is replaced with a discrete one by dividing the spatial domain into small units called grid cells and by sampling the solution at a finite number of specific instants of time. By replacing the continuous time coordinate by a discrete one, a one-dimensional 'grid' is effectively placed on that coordinate as well and the intervals are named time steps. In choosing a grid cell and time step size, a balance between achieving accurate answers and obtaining results within reasonable computational limits must be struck. The Richtmyer-Meshkov problem, in particular, requires a grid with a fine enough spatial resolution to resolve sharp features and a large enough spatial extent to keep the transmitted and reflected fronts in the computational domain throughout the simulation.

2.1.1 Spatial Resolution and Adaptive Mesh Refinement

Use of a uniformly fine grid over the entire computational region is not needed and is quite wasteful. As seen in the numerical Schlieren images of §1.2, only a relatively small portion of the entire flow domain needs to be resolved sharply. The technique of adaptive mesh refinement has been used to obtain high resolution in areas which need it while keeping computational costs reasonable. The particular algorithm used is due to Quirk [39], a brief summary of which can be found in appendix A. Given appropriate refinement criteria, the code will overlay specific cells of the computational mesh with finer ones, forming another level of grids. The user is allowed to determine how many grid levels to construct as well as the ratio of grid cell size across two consecutive levels.

Features of interest in the Richtmyer-Meshkov problem include shocks and regions of strong density gradient. We have found the `DensityGradient` test included in the standard distribution of *Amrita* to be satisfactory. After each time step, each cell is tested and flagged for refinement based on the following criteria:

$$\text{if } |\rho_{i+1,j} - \rho_{i,j}| > \tau \text{ then } \text{flag}_{i+1,j} = 1 \text{ and } \text{flag}_{i,j} = 1, \tag{2.2a}$$

$$\text{if } |\rho_{i,j+1} - \rho_{i,j}| > \tau \text{ then } \text{flag}_{i,j+1} = 1 \text{ and } \text{flag}_{i,j} = 1, \tag{2.2b}$$

where the tolerance, τ, was chosen to be

$$\tau = \begin{cases} (0.0025)\min(\rho_1, \rho_2) & \text{if } M_I \leq 1.5, \\ (0.1)\min(\rho_1, \rho_2) & \text{if } M_I > 1.5. \end{cases} \tag{2.2c}$$

The densities ρ_1 and ρ_2 are the asymptotic values of the initial density field as $x \to -\infty$

and $x \to \infty$ respectively and $\text{flag}_{i,j}$ identifies which cells are scheduled to be refined. The minimum of ρ_1 and ρ_2 merely serves as a characteristic density by which to scale. For weak incident shocks, a smaller coefficient is used in the test to ensure that the shock is refined even though the density gradient across it is very small. A comparison between the numerical Schlieren image of a simulation and the computational grid is shown in figure 2.1. The grid structure consists of a coarse grid which covers the entire domain and two additional levels containing finer cells in regions of strong density gradients. The `DensityGradient` test has resulted in the barely-visible vortex sheet originating from the triple point on the reflected shock being covered by the first additional grid level but not the second.

Numerical results should strive to be converged and not depend on mesh spacing. Pointwise convergence, however, is never achieved in shock capturing methods of computational fluid dynamics. Due to numerical viscosity inherent in the schemes, discontinuities are smeared out over several grid cells. This smearing out is a self-similar process: increases in the grid resolution do not reduce the number of cells across which the discontinuity is resolved over but do reduce the spatial extent. Shocks are an example of a flow feature that never converges in a pointwise sense. We can, however, require that gross diagnostic values of the flow, such as net circulation in the region or growth rate of the interface, do not vary with increased mesh refinement. In this thesis, Euler simulations were always performed with enough resolution that numerical data obtained did not change appreciably as the mesh was refined. It is in this sense that we claim our results in this work are converged.

2.1.2 Time Stepping

At any point in the computation, the time step for a mesh patch, Δt, is automatically determined by the program according to the two-dimensional CFL criterion:

$$\Delta t = \min\left(\frac{\Delta t_i \Delta t_j}{\sqrt{\Delta t_i^2 + \Delta t_j^2}}\right) \tag{2.3a}$$

$$\Delta t_i = \frac{\Delta x}{|u| + c} \qquad \text{and} \qquad \Delta t_j = \frac{\Delta y}{|v| + c} \tag{2.3b}$$

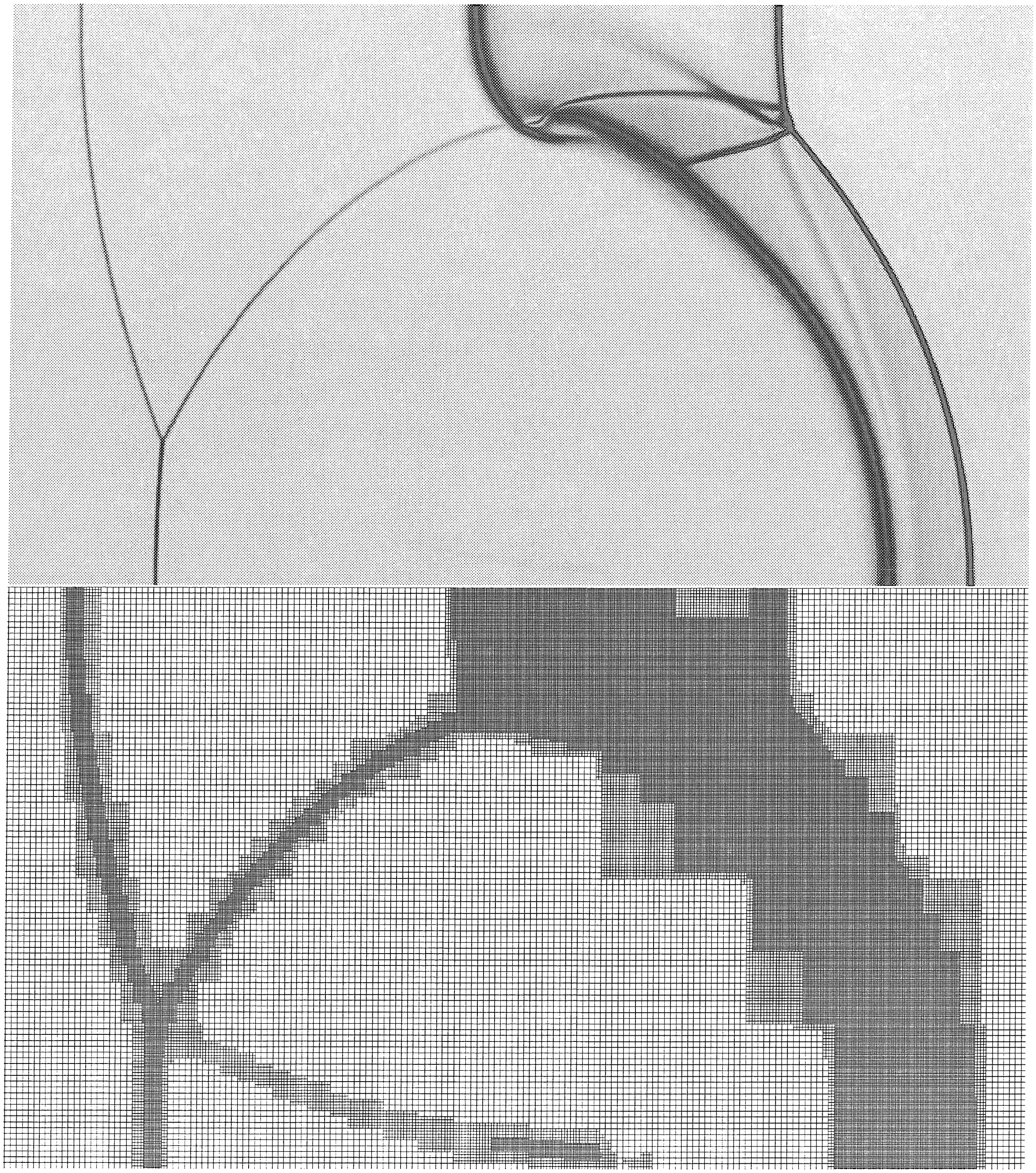


Figure 2.1: Numerical Schlieren-type image and the corresponding computational grid. Note regions of density gradients have been refined. These images show only a portion of the entire flow of figure 1.2.

where u and v are the fluid velocities in the x and y directions respectively and c is the speed of sound. In all our Euler simulations, the computation was actually marched with a time step half as large as that given by (2.3a) to ensure numerical stability, i.e., we specified a CFL number of 0.5 in our program.

The particular flow solver used blended the first order Roe flux function with the second order Lax-Wendroff flux function via a flux limiter. Details are presented in §2.3. Here, we merely state that this solver is second order accurate in both time and space away from discontinuities. To use the method (which was developed for one dimensional problems) in two dimensions, the solution in each cell is updated by performing Roe's method in one coordinate direction and then in the other. That is,

$$\mathbb{U}_{i,j}^{n+1} = \mathcal{L}_i \mathcal{L}_j \mathbb{U}_{i,j}^{n}, \tag{2.4}$$

where the vector $\mathbb{U}$ is given in (2.1) and $\mathcal{L}_i$ and $\mathcal{L}_j$ are the Roe solver operators in one dimension for the i-index and j-index directions, respectively. This approach is commonly known as operator-splitting. An alternative approach is to compute the fluxes in each direction and then use them simultaneously to update the flow solution:

$$\mathbb{U}_{i,j}^{n+1} = \mathcal{L}_{i,j} \mathbb{U}_{i,j}^{n}. \tag{2.5}$$

This is often called the finite volume approach. The difference between equations (2.4) and (2.5) is only in the order in which the fluxes are applied to $\mathbb{U}_{i,j}^{n}$ and $\mathcal{L}_{i,j}$ is not an operator which solves a truly multidimensional Riemann problem. The advantage in using operator splitting was shown by Strang [57]. If the operators $\mathcal{L}_i$ and $\mathcal{L}_j$ are each second order accurate in time, then a splitting of the form:

$$\mathbb{U}_{i,j}^{n+1} = \mathcal{L}_i^{\Delta t/2} \mathcal{L}_j^{\Delta t} \mathcal{L}_i^{\Delta t/2} \mathbb{U}_{i,j}^{n}, \tag{2.6}$$

where $\mathcal{L}^{\Delta t/2}$ indicates a half time step, results in a method that is second order accurate in two dimensions. Each application of an $\mathcal{L}$ operator is a computationally expensive operation and one would like to limit the number of these operations per time step. Use of the update operator (2.6) to evolve the solution results in the following computation for $\mathbb{U}_{i,j}^{n+1}$:

$$\mathbb{U}_{i,j}^{n+1} = \mathcal{L}_i^{\Delta t/2} \mathcal{L}_j^{\Delta t} \left(\mathcal{L}_i^{\Delta t} \mathcal{L}_j^{\Delta t} \right)^{n-1} \mathcal{L}_i^{\Delta t/2} \mathbb{U}_{i,j}^{0}, \tag{2.7}$$

where $\mathbb{U}_{i,j}^{0}$ denotes the initial condition. Except for the first two and the last operators, this is equivalent to using time stepping scheme (2.4). In a typical simulation, thousands of time steps are performed and the difference in the solutions resulting from using (2.4) or (2.6) is insignificant. The time step sequence (2.4) is chosen over (2.6) because it requires less operations.

It is a common practice among CFD researchers to use the sequence

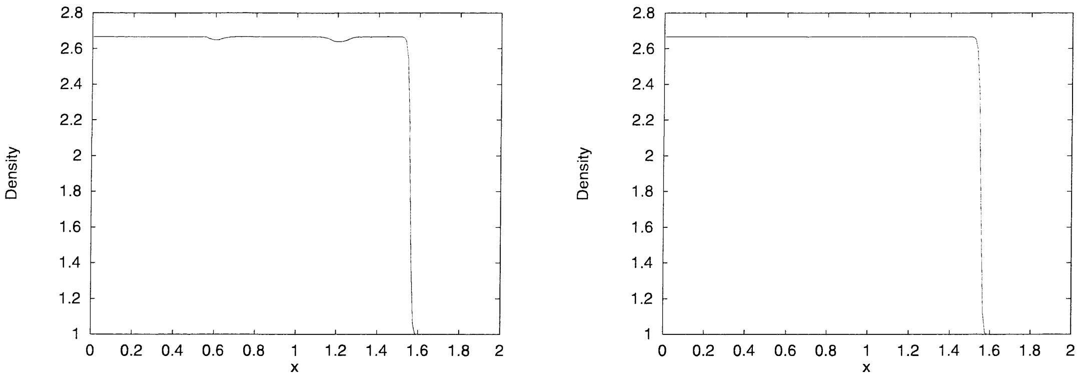
$$\mathbb{U}_{i,j}^{n+2} = \left(\mathcal{L}_i^{\Delta t} \mathcal{L}_j^{\Delta t} \right) \left(\mathcal{L}_j^{\Delta t} \mathcal{L}_i^{\Delta t} \right) \mathbb{U}_{i,j}^{n}, \tag{2.8}$$

combining the theoretical rigor of (2.6) with the computational cheapness of (2.4). However, taking two time steps together results in grid refinement occurring half as many times as if (2.4) was used instead and one runs the risk of a discontinuity running off the edge of a fine mesh patch [39].

2.2 Removal of Startup Errors Created by the Shock

Shock capturing schemes smear a shock wave across several mesh cells because of numerical diffusion. This smeared profile is determined purely by the numerical details of the simulation and has no physical or mathematical meaning. The Mach number of the shock, the CFL number used to determine the time step, and the specific flow solver used all dictate the exact profile of the shock. The action of an initially discontinuous discrete shock smearing out to its numerically determined profile causes disturbances to propagate back along characteristics and particle paths. The density profile of an initially discontinuous Mach 2 shock propagating is shown in figure 2.2(a) where anomalies are visible.

One way to eliminate this problem is to first allow an initially discontinuous shock to propagate long enough for it to settle down to its preferred profile and then use this smeared profile as the new initial condition for a simulation. In practice, this is accomplished by performing a simulation where the shock propagates into quiescent media with the same computational parameters that will be used in the Richtmyer-Meshkov simulation until the



(a) (b)

Figure 2.2: Shock propagation illustrating startup errors. (a) Initially discontinuous shock, (b) initially smeared profile.

startup errors are well behind the shock. These anomalies are erased by simply overwriting them with the correct post-shock values of density, velocity and pressure. This smeared numerical profile is then fed into the initial condition of the two-dimensional Richtmyer-Meshkov problem. This new shock will propagate without generating any startup errors, as shown in figure 2.2(b), since it is already at its numerically prescribed profile.

The examples shown in figure 2.2 were done without adaptive mesh refinement. Use of adaptive mesh refinement on the shock propagation problem also reduces startup errors. A density profile of an initially discontinuous shock, propagating with two finer grid levels of refinement employed, is shown in figure 2.3. The startup errors are barely visible. We believe this is because the shock lies within the fine-grid region and the startup errors seep back into the underlying coarser grid. The anomalies are short wavelength disturbances on the coarser mesh and are damped out by the numerical dissipation of the scheme. In all Euler simulations we do in this work, we use the technique of using a correctly smeared shock as the initial condition.

2.3 Flow Solvers and Flux Limiting

There exist a large number of shock capturing schemes for the Euler equations, each with their own advantages and disadvantages. In choosing a flow solver, one must consider the accuracy, robustness, dissipation and computational cost of each scheme. One must also be

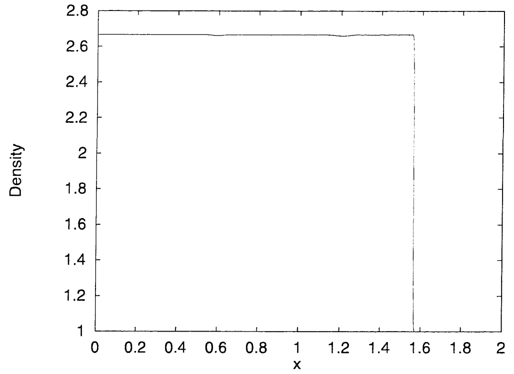


Figure 2.3: Shock wave propagation using an initially discontinuous shock with adaptive mesh refinement.

on guard against numerical artifacts created by the solver. In comparing the performance of several flow solvers on the Richtmyer-Meshkov problem, we have found some to suffer from a failing we believe to be the "odd-even decoupling" first reported by Quirk [38]. A comparison of numerical Schlieren images for simulations done with parameters $M_I = 10$, $At = 0.2$, $a_0 = 0.5$, and $L = 0.01$ and a variety of flow solvers is shown in figure 2.4. These runs were performed on uniform grids of 400x400 cells with no adaptive mesh refinement. The schemes employed for this comparison were basic first order schemes with no MUSCL reconstruction [61] or flux limiting [58] used to achieve higher resolution and accuracy.

The images in figure 2.4 clearly show the Godunov-type method [60] and the Roe scheme [44, 45] used have caused severe oscillations behind the shock. These oscillations are most noticeable near the top and bottom of the domain where the transmitted shock is closely aligned with the grid. The schemes known as EFM [37], HLLE [14], and AUSM [25] do not suffer from this failing. In general, these schemes tend to be somewhat more diffusive at contact surfaces than the Godunov and Roe methods. The EFM and HLLE schemes, in particular, cause an initially stationary density gradient to diffuse. This is in contrast to Riemann solver-based methods which preserve the gradient until nonzero velocities are present in the cells containing the non-uniform density region. The results of a simulation performed with the addition of Harten's [70] entropy fix* to the Roe solver is also shown. Comparison

---

*It is well known Roe's method can compute nonphysical solutions such as expansion shocks. Several modifications to Roe's method have been proposed, Harten's being one of the most popular. The fix, however, does not have any true justification and can be thought of simply as a convenient way of adding extra dissipation to the scheme [38].

with the basic Roe method reveals the additional viscosity added to the method from the fix has resulted in suppression of the oscillation. All these observations are in agreement with those of Quirk [38] for a failure of Riemann solvers he has named “odd-even decoupling.” We have noted this failure only occurs for Richtmyer-Meshkov problems involving very strong shocks.

The oscillations seen in figure 2.4(a) have been observed by Samtaney [50] using an independent code employing a Godunov-type method and in this work using an independent code equipped with a hybrid Godunov-Roe-type flow solver devised by Jacobs [21]. This evidence strongly suggests the numerical artifacts are due to shortcomings of the methods rather than coding errors. The only instance we have found in the literature where this failing has been mentioned is in the work of Samtaney and Meiron [54]. We believe this is simply due to the relatively small amount of work done involving very strong incident shocks. As more researchers investigate the Richtmyer-Meshkov instability generated by very strong shocks, this failure may become more widespread.

We believe the judicious use of artificial viscosity, such as that added by Harten’s entropy fix or that inherent in more diffusive schemes such as EFM, is necessary to suppress the failing. Researchers at Lawrence Berkeley National Laboratory have found that a Godunov-type method developed by Colella [11, 12] employing an artificial viscosity performs satisfactorily in their simulations of hypervelocity Richtmyer-Meshkov instabilities [16].

We have chosen to use Roe’s method, with the addition of Harten’s entropy fix, in the simulations with second order accuracy obtained through the use of a flux limiter. The flux limiter blends the low order Roe flux with the higher order Lax-Wendroff flux in a nonlinear way, such that the resulting method is second order accurate in time and space away from sharp gradients in the discrete solution. Theory and implementation of flux limiters is presented in [22] and [58]. We have experimented with the flux limiters known as Superbee [46], van Leer [61], and minmod [43] and found Superbee to be the least dissipative and minmod the most. This agrees with observations reported by Sweby [58] who compared the performance of several flux limiters on several test problems. The highly compressive Superbee limiter, however, was found to occasionally produce numerical artifacts in the solution. Thus, we have restricted ourselves to the van Leer and minmod limiters in all the simulations. In most simulations the minmod limiter was used and the larger dissipation was compensated for with increased spatial resolution.

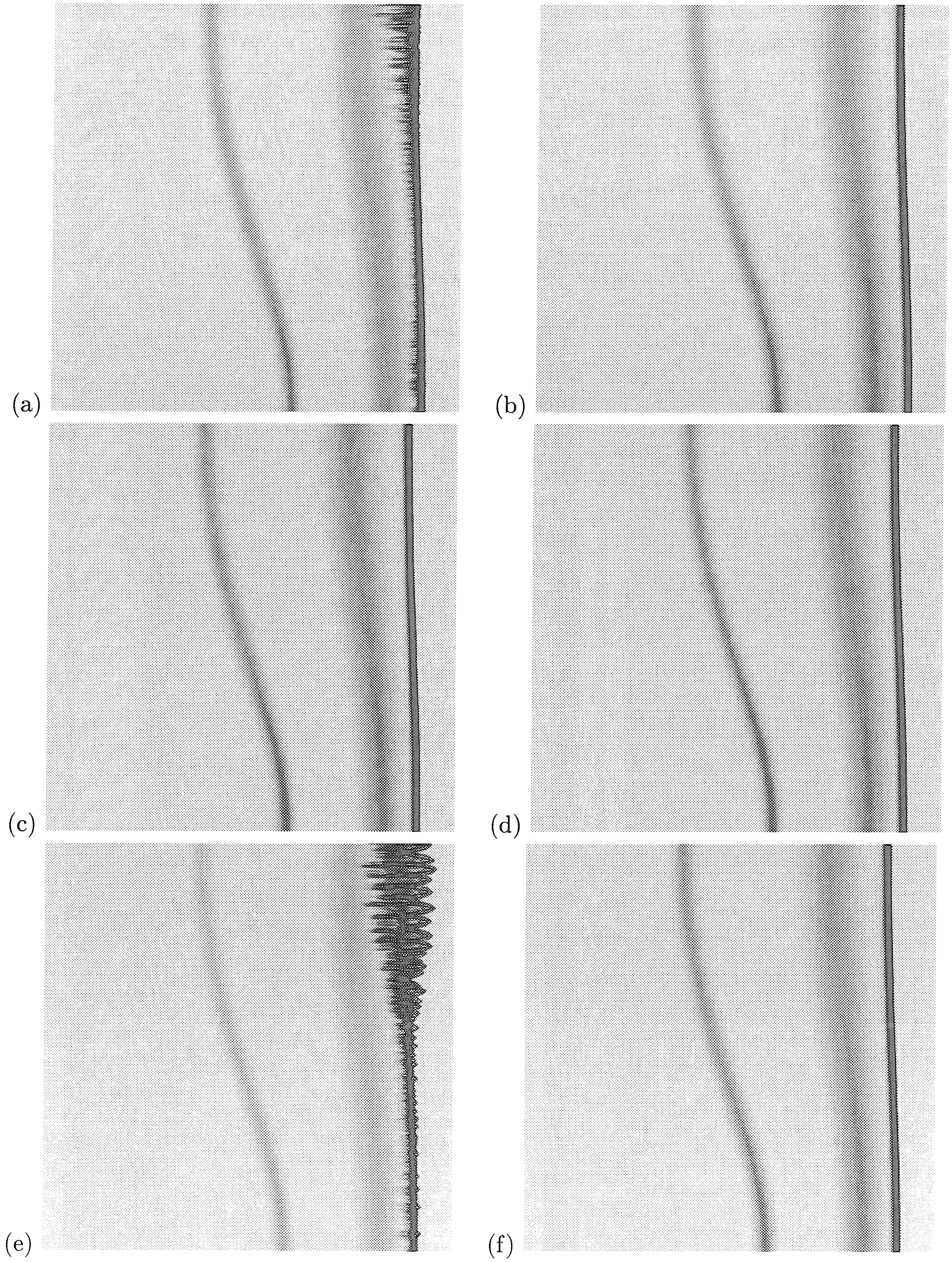


Figure 2.4: Some solvers cause oscillations behind the shock for hypervelocity Richtmyer-Meshkov problems: (a) Godunov's method, (b) EFM, (c) HLLE, (d) AUSM, (e) Basic Roe's method, (f) Roe's method with Harten entropy fix.

Chapter 3

Impulse Model

After 37 years, the most popular and widely known model in the field of Richtmyer-Meshkov instability is still Richtmyer's [41] original impulse model (discussed in §1.4). The model calculates an asymptotic value of the growth rate of a sharp, sinusoidal interface with a small amplitude to wavelength ratio by modeling the initial interaction with the incident shock as an impulsive force. The resulting expression, given by equation (1.6), provides a quick and simple way to determine the growth rate. Richtmyer found it to be quite accurate when compared to the growth rate computed from numerical solutions of the linearized Euler equations for three test cases provided post-shock parameter values were used in the model. Although Richtmyer only considered the case of a reflected shock wave, Meyer and Blewett [27] later found that the impulse model gave accurate predictions for the case of a reflected rarefaction if the average of the post-shock and pre-shock amplitudes was used in the model. The impulse approximation has been valuable to researchers by providing a means of modeling the instability without explicitly calculating the initial interaction between the incident shock and the interface. Numerous researchers [6, 28, 29, 30, 49, 54] have used the impulse approximation to develop other models for different problem configurations. Several researchers, however, have pointed out that the growth rate predicted by Richtmyer's model is not always accurate [15, 31, 32, 62, 68, 69].

The first experimental measurements of interfacial growth rates were performed by Meshkov [26]. Comparing his results against the growth rates predicted by the Richtmyer model, he found significant discrepancies. Meyer and Blewett [27] simulated Meshkov's experiments numerically and found disagreement between their nonlinear simulations and Meshkov's results. Since that time there have been numerous experimental and numerical studies of the instability, but discrepancies in growth rates have yet to be adequately explained [47]. One possible reason for the lower growth rates measured from experiments is the use of membranes to prevent mixing of the two gases on either side of the interface before the arrival of the incident shock. Vetter and Sturtevant [63] have recently provided evidence that these membranes reduce the growth rate by an appreciable amount. It is possible that these membranes may be responsible for the disagreement in growth rates between experiments and theory. This has prompted some researchers to study Richtmyer-Meshkov instabilities experimentally in configurations not requiring membranes [5, 6, 34]. These experiments could be compared with theories, thus eliminating a possible source of discrepancies. It is in this spirit that we study the Richtmyer-Meshkov instability occurring in continuously stratified fluids, such as in the experiments of Brouillette and Sturtevant [5, 6].

While impulse models have been proposed for the Richtmyer-Meshkov instability in continuously stratified fluids by Brouillette and Sturtevant [6], Mikaelian [30], and Saffman and Meiron [49], detailed comparisons against solutions of the Euler equations, such as those done by Yang *et al.* [69] for sharp interfaces, have not yet been done. In this chapter the accuracy of the Saffman-Meiron model, when compared to numerical simulations of the nonlinear Euler equations, is examined for both weak and strong incident shocks. We then investigate the compressibility and vorticity distribution of the flow calculated from Euler simulations and consider their implications for the validity of the popular impulse approximation.

3.1 Formulation of the Model and Numerical Implementation

The Saffman-Meiron impulse model computes the flow resulting from an impulsive force acting on an incompressible stratified fluid. Starting from conservation of momentum governed by the incompressible Euler equations,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x}, \tag{3.1a}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y}, \tag{3.1b}$$

the force which generates the instability is created by means of an impulsive pressure gradient at $t = 0$. We write $p(x, y, t) = P(x, y)\delta(t)$ where $\delta(t)$ is the Dirac delta distribution. The convective terms are eliminated through a dominant balance argument as follows. The pressure gradient terms are singular at $t = 0$ and other terms in equations (3.1a) and (3.1b) need to be of similar order to balance the equation. If the velocities, u and v, also had delta distribution singularities at $t = 0$, the time derivative terms would be even more singular. Since no terms would be left to balance those more singular terms, this possibility cannot be correct. It is the time derivative terms that balance the pressure gradients and the convective terms are small enough to be ignored at $t = 0$.

The equations simplify to:

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho}\frac{\partial p}{\partial x}, \tag{3.2a}$$

$$\frac{\partial v}{\partial t} = -\frac{1}{\rho}\frac{\partial p}{\partial y}. \tag{3.2b}$$

Integrating, formulas for the velocities are obtained in terms of the unknown function $P(x, y)$:

$$u_0 = -\frac{1}{\rho_0}\frac{\partial P}{\partial x}H(t), \tag{3.3a}$$

$$v_0 = -\frac{1}{\rho_0}\frac{\partial P}{\partial y}H(t), \tag{3.3b}$$

where $H(t)$ is the Heaviside function and $\rho_0(x, y)$ is a specified density field. It is here that the assumption of incompressibility is used. Although a shock wave is an inherently compressible phenomenon, the interaction with the flow is simulated as an impulsive force acting on an incompressible fluid. The divergence-free condition of fluid velocity, $\nabla \cdot \mathbf{u} = 0$, coupled with equations (3.3a) and (3.3b), gives the following equation for the function $P(x, y)$:

$$\frac{\partial}{\partial x}\left(\frac{1}{\rho_0}\frac{\partial P}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{1}{\rho_0}\frac{\partial P}{\partial y}\right) = 0. \tag{3.4a}$$

To apply this model to the Richtmyer-Meshkov problem, we let $\rho_0(x, y)$ be the density distribution given by equation (1.3) and choose $x_{\mathrm{dl}} = 0$. The incident shock imparts a constant velocity to the diffuse interface. The mean velocity of the layer is V, the velocity of the contact surface in the analogous one-dimensional shock-contact problem described in §1.3 which results from setting $a_0 = 0$ and $L = 0$. To simulate this behavior in the model, the impulsive force is chosen such that the induced x-velocity field, u_0, asymptotes to V far from the density layer. The flow in the impulse model is infinite in extent in the x-direction and periodic in y. As in the Euler simulations, only half a wavelength of the interface is simulated and $y = 0$ and $y = \lambda/2$ are considered to be reflecting wall boundaries. No flow through the walls is allowed so $v_0 = 0$ along $y = 0$ and $y = \lambda/2$. From equations (3.3a) and (3.3b) the requirements placed on the velocity fields $u_0(x, y)$ and $v_0(x, y)$ correspond to conditions on the function $P(x, y)$:

$$\frac{\partial P}{\partial y} = 0, \qquad y = 0, \lambda/2, \tag{3.4b}$$

$$\frac{\partial P}{\partial x} \to \begin{cases} -\rho_2 V, & x \to +\infty, \\ -\rho_1 V, & x \to -\infty, \end{cases} \tag{3.4c}$$

where ρ_1 and ρ_2 are the asymptotic values of $\rho_0(x, y)$ as $x \to -\infty$ and $x \to \infty$, respectively. Equations (3.4a)-(3.4c) constitute an elliptic boundary value problem for the function $P(x, y)$. This can be solved and the velocity field induced by the impulsive pressure force computed from equations (3.3a) and (3.3b).

We now describe the numerical solution of this boundary value problem. Although the boundary conditions in x are only satisfied in the limit as $x \to \pm\infty$, a numerical domain must be finite in extent. Also, much higher grid resolution is required in the vicinity of the interface, centered at $x = 0$, than in the rest of the domain. For these reasons, we choose to solve equations (3.4a)-(3.4c) in a stretched coordinate system. Following Pham and Meiron [36], the following change of variable from x to η is made:

$$\eta = \tanh(\gamma x), \qquad \frac{\partial}{\partial x} = \frac{d\eta}{dx}\frac{\partial}{\partial \eta}, \qquad \frac{d\eta}{dx} = \gamma(1 - \eta^2), \tag{3.5}$$

where the parameter γ has been introduced to control the degree of stretching. As γ is increased, the density of grid lines near $x = 0$ is increased. The grid reverts back to uniform spacing in the limit as $\gamma \to 0$. An example of such a stretched grid with 50 points in the η direction and 20 in the y direction is shown superimposed on the numerical Schlieren image of a small amplitude Richtmyer-Meshkov field in figure 3.1. Applying the change of coordinates to equations (3.4a)-(3.4c), the boundary value problem becomes

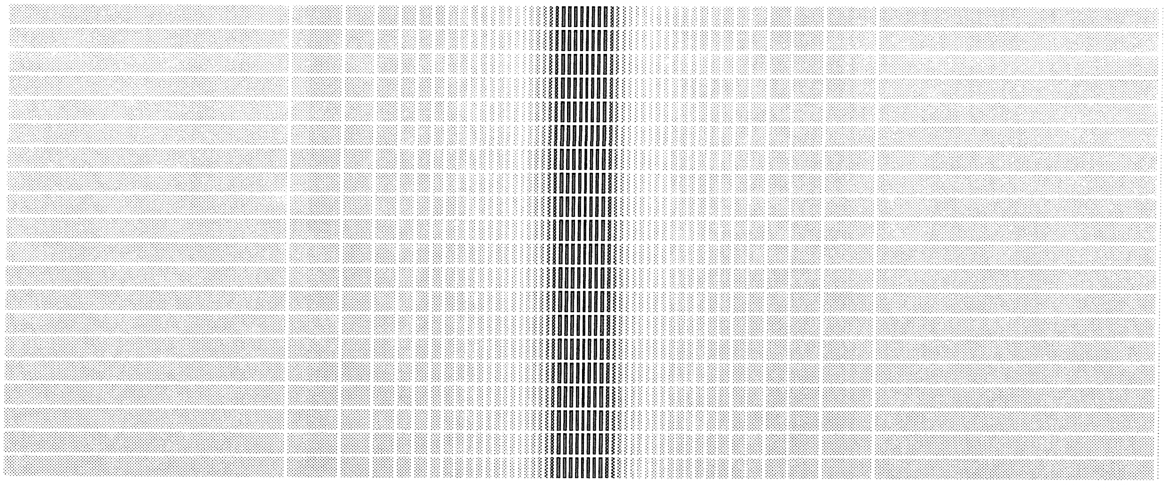


Figure 3.1: Example of a 50x20 stretched grid (with $\gamma = 1$) superimposed on numerical Schlieren image of density layer with $a_0 = 0.01$ and $L = 0.1$.

$$\gamma(1-\eta^2)\frac{\partial}{\partial\eta}\left(\frac{\gamma(1-\eta^2)}{\rho_0}\frac{\partial P}{\partial\eta}\right) + \frac{\partial}{\partial y}\left(\frac{1}{\rho_0}\frac{\partial P}{\partial y}\right) = 0, \tag{3.6a}$$

$$\frac{\partial P}{\partial y} = 0, \qquad y = 0, \lambda/2, \tag{3.6b}$$

$$\gamma(1-\eta^2)\frac{\partial P}{\partial\eta} \to \begin{cases} -\rho_2 V, & \eta \to +1, \\ -\rho_1 V, & \eta \to -1. \end{cases} \tag{3.6c}$$

Note that the transformation has resulted in the infinite interval $x \in (-\infty, \infty)$ being mapped to the finite range $\eta \in [-1, 1]$. From (3.6c) it is seen that $\partial P/\partial\eta$ must become singular as $\eta \to \pm 1$ to give a nonzero boundary condition. To avoid numerical errors, the problem must be reformulated to remove this singular behavior.

The full two-dimensional density distribution in front of the shock, $\rho_0(x, y)$, is given by equation (1.3). The corresponding one-dimensional density profile (with $a_0 = 0$) is denoted by $\bar{\rho}(x)$ and the problem is reformulated in terms of density and pressure fluctuations, $\rho'(x, y)$ and $p'(x, y)$, where

$$\rho_0(x, y) = \bar{\rho}(x) + \rho'(x, y), \tag{3.7}$$

$$P(x, y) = -V\int_0^x \bar{\rho}(\xi)d\xi + p'(x, y), \tag{3.8}$$

to obtain the following elliptic problem:

$$\gamma(1-\eta^2)\frac{\partial}{\partial\eta}\left(\frac{\gamma(1-\eta^2)}{\rho_0}\frac{\partial p'}{\partial\eta}\right)+\frac{\partial}{\partial y}\left(\frac{1}{\rho_0}\frac{\partial p'}{\partial y}\right)=V\frac{\partial}{\partial x}\left(\frac{\bar{\rho}}{\rho_0}\right), \tag{3.9a}$$

$$\frac{\partial p'}{\partial y}=0, \qquad y=0,\lambda/2, \tag{3.9b}$$

$$\frac{\partial p'}{\partial\eta}\to 0, \qquad \eta\to\pm1. \tag{3.9c}$$

The fluid velocity is obtained through:

$$u_0(x,y)=V\frac{\bar{\rho}}{\rho_0}-\frac{1}{\rho_0}\frac{\partial p'}{\partial x}, \tag{3.10a}$$

$$v_0(x,y)=-\frac{1}{\rho_0}\frac{\partial p'}{\partial y}. \tag{3.10b}$$

Equation (3.9a) is discretized using central differences at half grid points. A uniform mesh in η and y is used with spacing $\Delta\eta$ and Δy respectively. Dropping the subscript "0" from the density field, the discretized equation is:

$$\begin{aligned}\left[-\frac{\gamma(1-\eta_i)^2}{\Delta\eta^2}\left(\frac{\gamma\left(1-\eta^2_{i+\frac{1}{2}}\right)}{\rho_{i+\frac{1}{2},j}}+\frac{\gamma\left(1-\eta^2_{i-\frac{1}{2}}\right)}{\rho_{i-\frac{1}{2},j}}\right)-\frac{1}{\Delta y^2}\left(\frac{1}{\rho_{i,j+\frac{1}{2}}}+\frac{1}{\rho_{i,j-\frac{1}{2}}}\right)\right]p'_{i,j}&\\
+\frac{1}{\Delta\eta^2}\frac{\gamma\left(1-\eta^2_{i+\frac{1}{2},j}\right)}{\rho_{i+\frac{1}{2},j}}p'_{i+1,j}+\frac{1}{\Delta\eta^2}\frac{\gamma\left(1-\eta^2_{i-\frac{1}{2},j}\right)}{\rho_{i-\frac{1}{2},j}}p'_{i-1,j}&\\
+\frac{1}{\Delta y^2}\frac{1}{\rho_{i,j+\frac{1}{2}}}p'_{i,j+1}+\frac{1}{\Delta y^2}\frac{1}{\rho_{i,j-\frac{1}{2}}}p'_{i,j-1}=f_{i,j},&\end{aligned} \tag{3.11}$$

where the function $f(x,y)$ is the partial derivative $f(x,y)=V(\bar{\rho}(x)/\rho_0(x,y))_x$ and i and j are the grid indices in the η and y directions, respectively. The Neumann boundary conditions are incorporated by modifying the stencil at the boundaries. For example, at the left edge of the computational domain, $p'_{i-1,j}$ is replaced with a value consistent with the Neumann condition. From (3.9c) the boundary condition is zero normal derivative so $p'_{i-1,j}=p'_{i+1,j}$ to the order of the central difference formula.

3.2 Calculation of Diagnostics of the Flow

Two important diagnostic quantities of the Richtmyer-Meshkov problem are the growth rate of the interface and the net circulation. To determine the total circulation of the flow at any point in time the line integral of fluid velocity along a closed circuit enclosing the computational domain was computed numerically using the trapezoidal rule. To compute the growth rate of the density layer, the interface was defined as the mid-line of the density layer, $x(y) = a_0 \cos(y)$ in the model. The growth rate was computed from the fluid velocities at the peak and trough of the interface:

$$\dot{a}_{\text{model}} = \frac{1}{2}\left[u_0(a_0, 0) - u_0(-a_0, \lambda/2)\right]. \tag{3.12}$$

In general, grid points do not lie exactly on the interface so the velocities are obtained from linear interpolation using the two points straddling the interface along $y = 0$ and $y = \lambda/2$.

To make direct comparisons with the model, a definition of the interface in the Euler simulations must also be made. The so-called "level set" formulation of interface tracking of Mulder *et al.* [33] was implemented. This was done by adding a passive scalar function $\psi(x, y)$ to the unknowns of the Euler equations (2.1). The new field is simply advected along with the flow and does not interact with any of the other fields. Its purpose is simply to identify specific fluid particles by "tagging" them with a specific value of ψ. The Euler equations remain unchanged except for the addition of a fifth equation describing the evolution of $\psi(x, y)$:

$$\frac{\partial(\rho\psi)}{\partial t} + \nabla \cdot (\rho\psi\mathbf{u}) = 0. \tag{3.13}$$

At $t = 0$, $\psi(x, y)$ is initialized as:

$$\psi(x, y) = \frac{2[\rho(x, y) - \rho_1]}{\rho_1 - \rho_2} + 1, \tag{3.14}$$

where $\rho(x, y)$ is the quiescent Richtmyer-Meshkov density field given by equation (1.3) and ρ_1 and ρ_2 are the asymptotic values of $\rho(x, y)$ as $x \to -\infty$ and $x \to \infty$, respectively. This tags all fluid particles on the mid-line of the initial density layer, $x = x_{\text{dl}} + a_0 \cos(y)$ in equation (1.3), with the value $\psi = 0$. At any point in the simulation, the interface was defined as the locus of points satisfying $\psi(x,y) = 0$. The instantaneous growth rate was determined by subtracting interpolated u-velocities at the peak and trough of the interface. It was assumed that these were located along the top and bottom boundaries of the domain.

3.3 Results for Weak Incident Shocks

The impulsive approximation is known to accurately predict the growth rate of an interface of zero thickness and small amplitude when the Richtmyer-Meshkov instability is generated by weak incident shocks [15]. The mean velocity imparted to the interface, V, by a weak incident shock is small compared to the speed of the transmitted and reflected shocks. In addition, the growth of the instability is small for weak incident shocks. The residence time of the shocks in the vicinity of the interface, then, is small compared to the timescale of the growth of the interface. The time interval during which the incident shock interacts with a diffuse interface of finite amplitude will be larger and it is not clear whether the impulse approximation is still accurate. Some justification for the approximation for layers of finite thickness, L, was provided by Pham [35] who investigated the solution of the one-dimensional shock-layer problem (setting $a_0 = 0$ in equation (1.3)) numerically using Chisnell's theory [10].

To determine the accuracy of the impulse model of Saffman and Meiron, growth rates obtained from the impulsive model are compared with those obtained from full Euler simulations. The Euler equations are non-dimensionalized with the following characteristic quantities:

$$u_c = V, \qquad l_c = \lambda/2 = \pi, \qquad \rho_c = \rho_L + \rho_R, \tag{3.15}$$

where ρ_L and ρ_R are the post-shock densities and V, the speed of the contact surface in the one-dimensional shock-contact problem discussed in §1.3. Because of the choice of u_c, the mean velocity of the shocked density layer in the Euler simulations will be 1. To make direct comparisons against the model, the value of V specified in equation (3.9a) of the model is chosen to be 1.

A second check of the accuracy of the impulse model is performed by comparing total circulation in the entire computational domain computed from both simulations. In com-

Table 3.1: Impulse model computational parameters for weak shocks.

a_0	At	L	γ	$\eta_{\max}$
0.01	0.2	0.01	2.0	0.9995
		0.1	1.0	0.999
		1	0.2	0.9
	0.8	0.01	2.0	0.9995
		0.1	1.0	0.999
		1	0.2	0.9
1	0.2	0.01	0.2	0.85
		0.1	0.2	0.9
		1	0.2	0.95
	0.8	0.01	0.2	0.85
		0.1	0.2	0.9
		1	0.2	0.95

paring the circulation in the domain from the impulse model with that obtained from Euler simulations, the assumption is being made that any vorticity in the domain not located in the shocked layer is negligible. We will oftentimes refer to such vorticity as being located "in the bulk of the fluid," a naming convention used by Wouchuk and Nishihara [68] to differentiate this from vorticity on the interface. They show that the bulk vorticity is of second order in shock strength for Richtmyer-Meshkov problems with sharp, sinusoidal interfaces of infinitesimal amplitude. We make the assumption that this is true for thick interfaces of finite amplitude as well and do not differentiate between interfacial and total circulation in this section.

Numerical experiments demonstrated that the numerical solution of the impulse model, as described in §3.1, required much higher resolution in the η-dimension than the y-dimension. The elliptic problem was solved using a banded LU solver. Convergence studies indicated that a grid resolution of 6000x40 points was more than sufficient to obtain converged results for the growth rate and circulation from the model. The arbitrary constant in the solution (arising from use of Neumann boundary conditions on all sides) was fixed by setting $p'(6000, 40) = 0$ in the solution of the matrix problem. The asymptotic boundary conditions in the η-direction were enforced at $\eta = \pm\eta_{\max}$. Table 3.1 lists the computational parameters used in the numerical solution of the model. In this chapter only positive Atwood ratio cases are considered simply to limit the number of cases studied.

Comparisons between results from Euler simulations and solutions of the impulse model are shown in figures 3.2-3.5. The incident shock strength used in the Euler simulations was $M_I = 1.01$. No Mach number need be specified in the model. The strength of the impulsive pressure force is determined by the choice of V in (3.9a). Adaptive mesh refinement was used in the Euler simulations to obtain converged growth rates and circulations in the manner specified in §2.1.1. The coarsest grid level had a spatial resolution of 100 cells in the y-direction. The x extent of the domain needed to ensure that the transmitted and reflected shocks remained in the computational domain during the entire simulation was calculated at run time using the solution of the equivalent one-dimensional shock-contact problem described in §1.3. Requiring that $\Delta x = \Delta y$, the number of cells in the x-direction was determined automatically. Each additional grid level of refinement was composed of cells whose Δx and Δy were half as large as the next coarser grid. It was found that in order to get converged results for the growth rate, three additional grid levels were needed for simulations of problems with $L = 0.01$. Two levels were needed when $L = 0.1$ and no additional levels were needed for $L = 1$ cases.

From figures 3.2-3.5 it is seen that the impulsive model of Saffman and Meiron accurately predicts the limiting value of growth rate and circulation for almost every case, spanning a wide range of Atwood ratios, initial amplitudes and layer thicknesses. Each figure displays the growth rate (left column) and circulation (right column) calculations for a fixed Atwood ratio and initial amplitude. Each row corresponds to a specific layer thickness.

In each plot of growth rate (left column of figures 3.2-3.5), the result from the Richtmyer impulse model (see §1.4) for an equivalent interface of zero thickness is also shown. For small amplitude interfaces (figures 3.2-3.3) the discrepancy between the Richtmyer result and the Saffman-Meiron result is due to nonzero values of layer thickness, L. While the Saffman-Meiron growth rates are very close to the Richtmyer predictions for $L = 0.01$ cases, the reduction in growth rate seen for $L = 1$ layers (bottom rows in figures) is substantial. The Saffman-Meiron model correctly calculates this reduction.

A sharp spike in the early time growth rate is seen in some of the plots. These are artifacts resulting from the fact that the growth rates are computed using the discrete u-velocities $u_{i,j}$ (see §3.2) and these spikes are not meaningful. The spikes occur when the incident shock, which has a finite numerical thickness, passes through the point on the interface (the curve $\psi = 0$) at $y = 0$ (the "peak" of the interface). The numerical u-velocity at the peak is not physical since the $u_{i,j}$ values within the thick shock are dictated purely

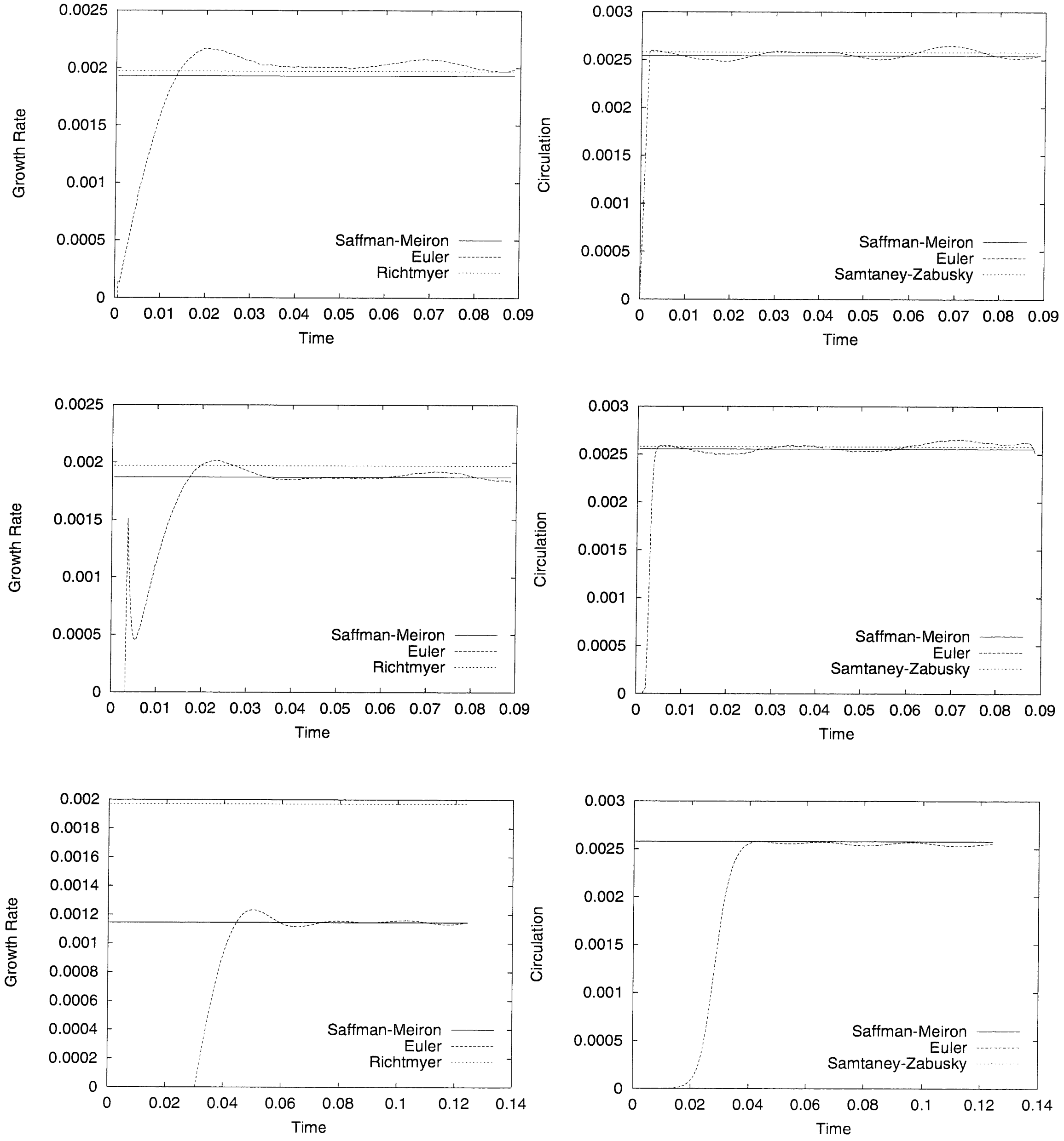


Figure 3.2: Comparisons between models and Euler simulations for $At = 0.2$ and $a_0 = 0.01$. (Columns: left: growth rate, right: circulation, Rows: (top to bottom) $L = 0.01$, $L = 0.1$, $L = 1$)

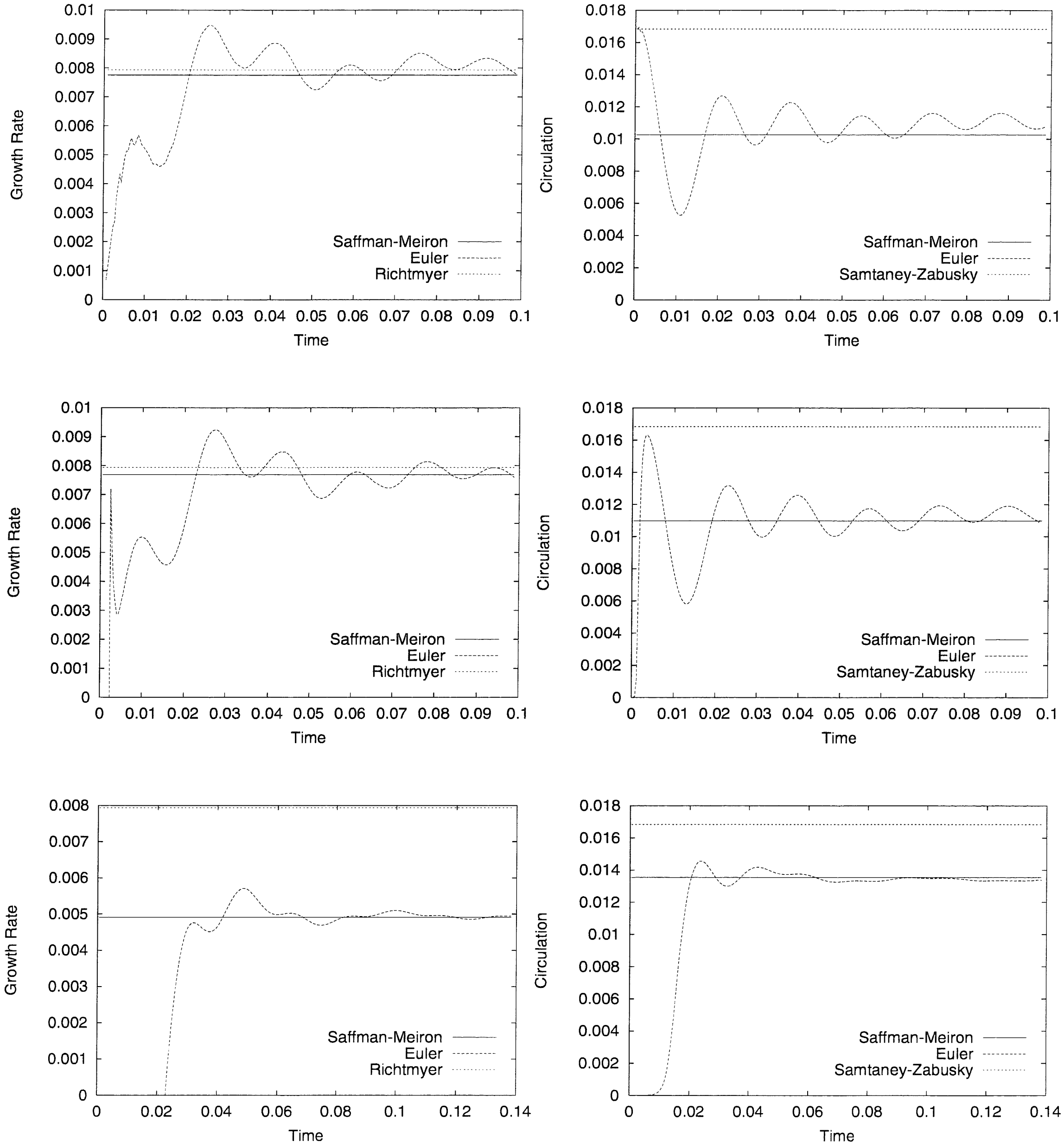


Figure 3.3: Comparisons between models and Euler simulations for $At = 0.8$ and $a_0 = 0.01$. (Columns: left: growth rate, right: circulation, Rows: (top to bottom) $L = 0.01$, $L = 0.1$, $L = 1$)

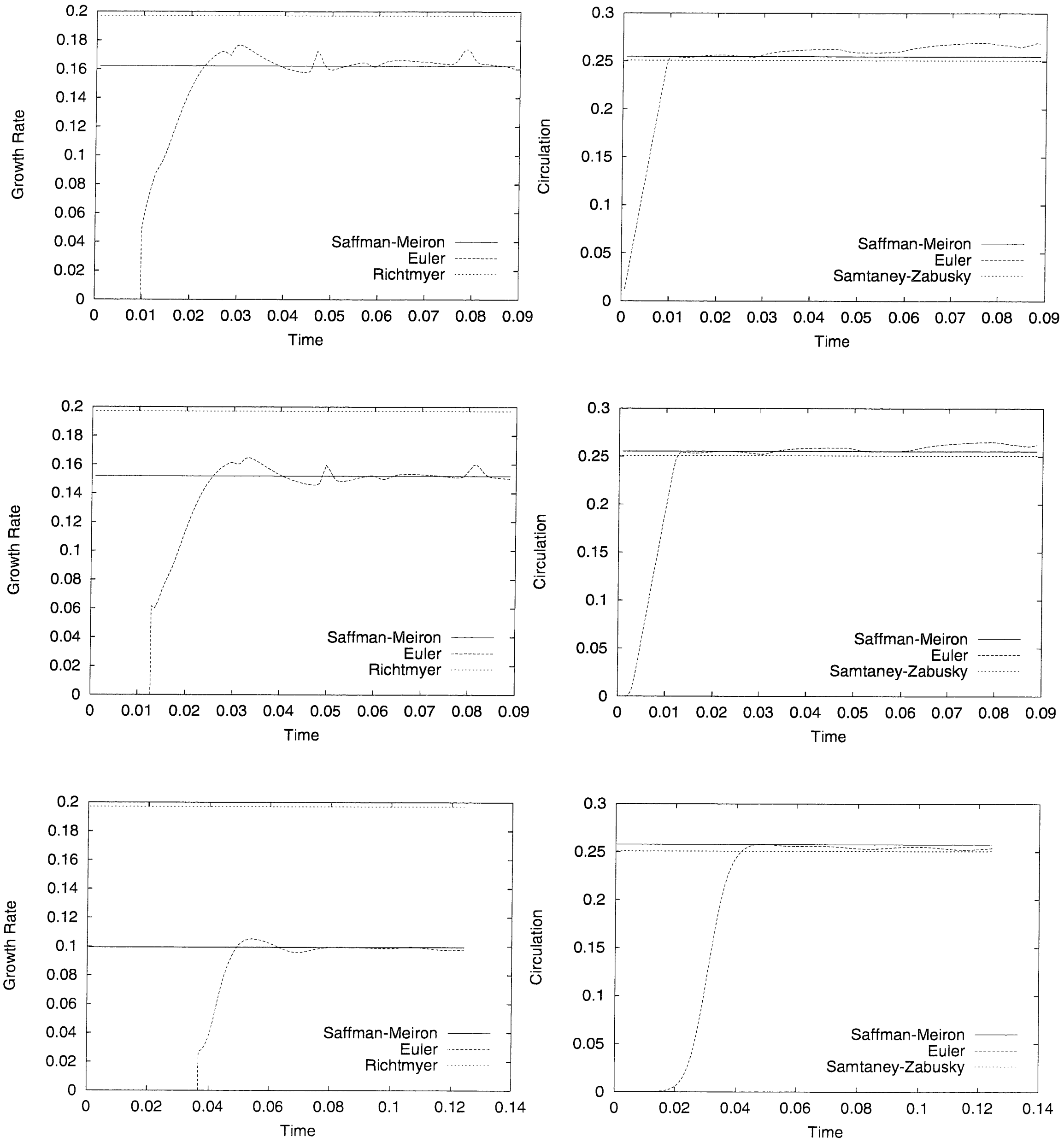


Figure 3.4: Comparisons between models and Euler simulations for $At = 0.2$ and $a_0 = 1$. (Columns: left: growth rate, right: circulation, Rows: (top to bottom) $L = 0.01$, $L = 0.1$, $L = 1$)

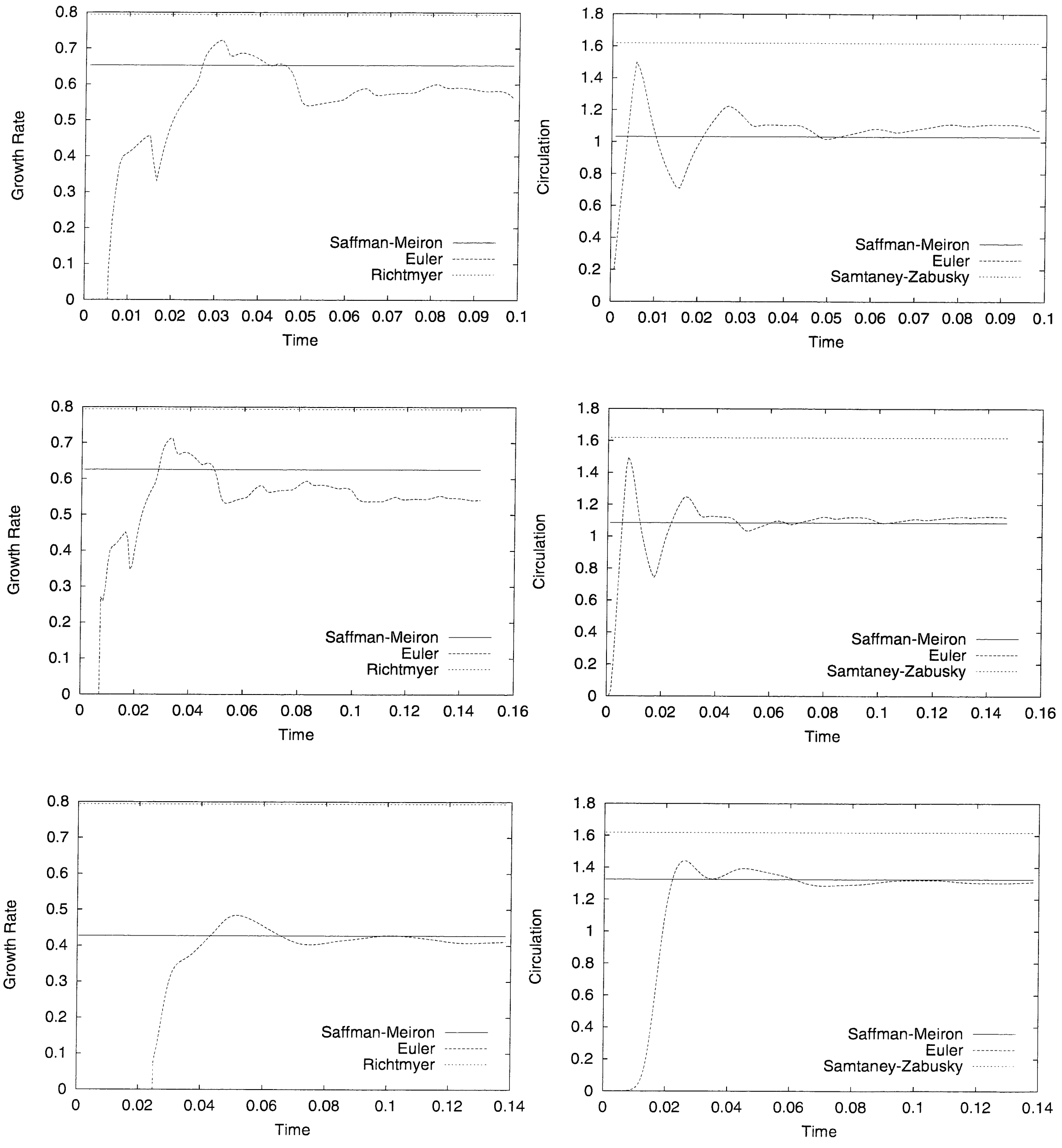


Figure 3.5: Comparisons between models and Euler simulations for $At = 0.8$ and $a_0 = 1$. (Columns: left: growth rate, right: circulation, Rows: (top to bottom) $L = 0.01$, $L = 0.1$, $L = 1$)

by numerical issues (see §2.2). Technically, this velocity should not be used to compute the growth rate. Similarly, when the shock passes through the interface at $y = \lambda/2$ (the "trough"), a spike in the growth rate also occurs. This spike is not seen in the plots of figures 3.2-3.5 because these spikes drive the growth rate below zero and the abscissa range of the plots are restricted to positive numbers.

The oscillations in growth rates are real, however, and are the result of pressure perturbations bouncing between the transmitted and reflected shocks and the shocked density layer [41]. These waves are generated by the corrugated shock fronts as they oscillate in the process of stabilizing themselves [42]. Troughs along a shock strengthen as the front propagates, sending compression waves back into the flow field. Similarly, peaks along a shock front are expansive regions and weaken as the shock moves, sending expansive disturbances back into the flow behind. The temporal oscillations in the growth rate about the asymptotic value represent the interaction of these pressure waves with the interface. A closer look at these interactions is delayed until §3.6.

For the large amplitude cases shown in figures 3.4 and 3.5, there is a discrepancy in growth rate between the two impulsive models even for thin $L = 0.01$ interfaces. The failure of the Richtmyer model to obtain the correct growth rate in these cases is due to the assumption of a small amplitude to wavelength ratio inherent in the Rayleigh-Taylor analysis used in the Richtmyer theory (see §1.4). The Saffman-Meiron model places no such restrictions on the amplitude of the interface.

The growth rates predicted by the Saffman-Meiron model for the $At = 0.8$ case with large initial amplitudes, $a_0 = 1$ (figure 3.5), show some disagreement with Euler simulations for the $L = 0.01$ and $L = 0.1$ cases. This is not true of the corresponding $At = 0.2$ cases shown in figure 3.4. Numerical Schlieren-type images of the flow for both $At = 0.2$ and $At = 0.8$ cases with $a_0 = 1$ and $L = 0.1$ are shown in figure 3.6 for comparison. The transmitted shock in the $At = 0.8$ case is undergoing regular reflection along the top of the domain. The high Atwood ratio case contains significant secondary waves between the transmitted shock and density layer. These waves modify the evolution of the interface in ways not accounted for in the impulse model. We believe these waves are responsible for the small disagreement in growth rate for the $At = 0.8$ cases.

The right column of figures 3.2-3.5 compares the calculations of total circulation. To examine the variation of circulation with layer thickness, the estimate of initial circulation

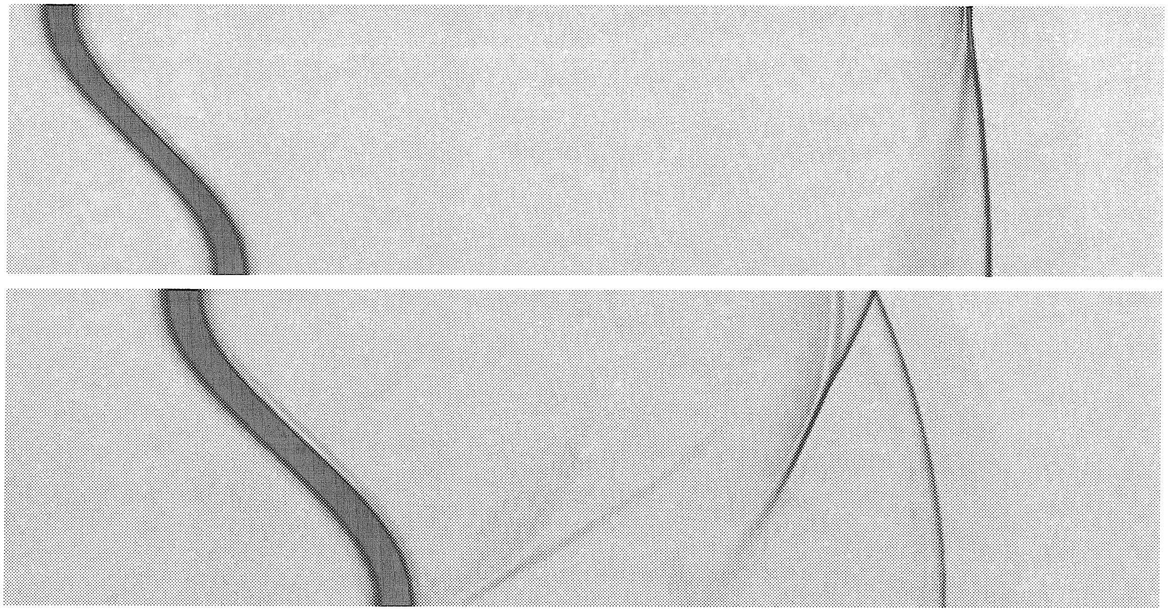


Figure 3.6: Schlieren images of two simulations with $a_0 = 1$ and $L = 0.1$ (Top: $At = 0.2$, Bottom: $At = 0.8$). The $At = 0.8$ case has secondary waves affecting the evolution of the density layer.

deposited on an equivalent interface of zero thickness computed from a method due to Samtaney and Zabusky [51] has also been plotted in each graph. From the graphs it is seen that although there is some difference between the Samtaney-Zabusky result for $L = 0$ interfaces and the maximum value of circulation computed from Euler simulations for interfaces with non-zero L (particularly the $At = 0.8$ cases), the reduction due to finite thickness of the density layer is not nearly as pronounced as the corresponding reduction in growth rate. This implies that it is not the total magnitude of circulation which is important in determining the growth rate of the interface but the distribution of the vorticity and its proximity to the interface. Note that while the Samtaney-Zabusky method produces estimates of the initial circulation, the impulse model of Saffman and Meiron accurately predicts the late time circulation which ultimately determines the asymptotic growth rate of the diffuse interface.

For low Atwood ratio interfaces the total circulation calculated from Euler simulations varies little with time after achieving its initial post-shock value. For high Atwood ratios the circulation first attains a maximum value, then drops to a lower value before rising again to oscillate about an asymptotic value. The initial drop in circulation is due to compression waves generated by the oscillations of the transmitted and reflected shocks modifying the interfacial vorticity through secondary baroclinic interactions [52]. Note that these drops correspond to the small decrease in the growth rates for the high Atwood ratio cases [52]. These pressure waves bouncing between the shocks and the layer continue to modify the vorticity distribution in the layer as evidenced by the oscillation in circulation with time. Thicker layers do not experience as great a vorticity modification as thinner layers do. We believe this is because these waves reflect back towards the shocks after hitting the outer edges of the thick layer. In effect, the majority of the vorticity deep inside the layer is better insulated from pressure waves for thicker layers than thinner ones.

The results in this section show that the Saffman-Meiron model accurately models the Richtmyer-Meshkov instability occurring in continuously stratified media. It correctly predicts the reduction in the growth rate due to non-zero values of L. With the use of any standard banded LU solver, it can be considered a simple method of simulating the Richtmyer-Meshkov instability for weak incident shock waves.

3.4 Results for Stronger Incident Shocks

We now turn our attention to the case of stronger incident shocks. While the difference in mean velocities of the transmitted (and reflected) shock and interface, $U_t - V$ (and $V - U_r$), increases with M_I (see figure 1.8), the growth rate of the interface increases much faster. In contrast to weak incident shock cases, then, the residence time of the shocks in the vicinity of the interface is not necessarily small compared to the timescale of growth of the interface for stronger incident shocks. Thus, the model of Richtmyer-Meshkov instability as an impulsive force acting on an interface followed by passive evolution of an isolated layer in an incompressible fluid is suspect. The transmitted and reflected shocks might modify the evolution of the interface.

To obtain reasonable predictions, we found it necessary to use post-shock values of interface parameters in the Saffman-Meiron model just as Richtmyer used in his. The input to the model is the two-dimensional quiescent density distribution, $\rho_0(x, y)$. We make the assumption that the post-shock density field is still described by equation (1.3), just as the pre-shock field was, but with new values of At, a_0, and L.

The values of At and a_0 used were At_{post} and $a_{0,\text{post}}$ whose calculation is explained in §1.4. In lieu of a formula, the determination of post-shock layer thickness, L_{post}, was done for each problem by manually determining an effective value for L. A one-dimensional shock-layer problem was run and the post-shock density profile of the layer was compared to that given by equation (1.3) with $At = At_{\text{post}}$ and $a_0 = 0$. Parameter L (and x_{dl}) was chosen so the two profiles overlapped. Although this procedure is admittedly crude, we believe it is the most accurate way to obtain a value of L to use in the model. Although Chisnell [10] considered the one-dimensional problem of shock propagation through a density layer, his solution was not simple to obtain and only resulted in an approximate description of the shocked density layer. Once a suitable value for L_{post} was calculated in this manner, the function $\rho_0(x, y)$ input to the Saffman-Meiron model was given by equation (1.3) with $At = At_{\text{post}}$, $a_0 = a_{0,\text{post}}$, and $L = L_{\text{post}}$. Care was taken to calculate values of L_{post} immediately after passage of the shock to minimize errors resulting from the diffusion of the layer due to numerical viscosity in the Euler simulations.

For Richtmyer-Meshkov problems involving strong shocks, not all the vorticity in the flow is located within the shocked density layer. As the incident shock undergoes refraction through the density layer, both a corrugated transmitted and reflected shock is generated. These shocks oscillate as they propagate, leaving vorticity in their wake. For strong shocks, this vorticity left in the bulk of the fluid can be non-negligible [68]. At this point, then, a distinction between the circulation in the shocked layer, which we will call "interfacial circulation," and the total circulation in the entire flow field is made. We now define the density layer precisely to avoid any ambiguities. The interfacial circulation is defined as the area integral of vorticity over the region containing all points such that $|\psi(x, y)| \leq 0.99$, where $\psi(x, y)$ is the "level set" passive scalar field introduced in §3.2. In the Euler code, this integral is calculated using the two-dimensional trapezoidal rule and central differences to calculate $\omega_{i,j}$:

$$\Gamma_{\text{int}}(t) = \sum_{l=0}^{l_{\max}} \Delta x_l \Delta y_l \sum_{k=1}^{\text{NGA}_l} \sum_{\substack{i,j \\ \text{flag}_{i,j}=0 \\ |\psi_{i,j}| \leq 0.99}} \left[\frac{v_{i+1,j} - v_{i-1,j}}{2\Delta x_l} - \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta y_l} \right], \tag{3.16}$$

where $l_{\max}$ is the number of grid levels in the AMR grid hierarchy (see appendix A), NGA_l is the number of grid patches on level l, and $\text{flag}_{i,j}$ identifies which cells have been flagged for refinement. The innermost summation loops over all i and j in mesh patch k of grid level l. If a computational cell (i, j) in patch k of level l has been covered with finer cells in a higher grid level, the adaptive mesh refinement machinery changes $\text{flag}_{i,j}$ from zero to one. This indicates that the summation should not use the data in the cell on grid level l but the data in the corresponding finer cells in grid level $l + 1$. In this way the sum (3.16) uses the most accurate data at any spatial location in the computational domain. The interfacial circulation is much more important in determining the growth rate than the total circulation. Comparison between the two will provide a measure of the amount of vorticity deposited in the bulk of the fluids by the shocks.

Comparisons between models and Euler simulations are shown in figures 3.7-3.10. All cases considered have the same small initial amplitude of $a_0 = 0.01$. This was done to ensure that the interface grew with a constant growth rate. Richtmyer-Meshkov instabilities with large amplitudes initiated by strong incident shocks do not have a constant limiting growth rate because much of the energy in the flow goes into creation of vortex roll-up structures (see figure 1.2) rather than pure increase in amplitude. The growth rate decays with time (see figure 1.4) for large amplitude interfaces. The Saffman-Meiron model is not restricted to cases with a small amplitude to wavelength ratio and could be used to calculate the vorticity field resulting from the shock-layer interaction for interfaces of amplitude larger than $a_0 = 0.01$. However, because the growth rate for these cases varies with time, the impulse model, which does not contain a time variable, cannot be used by itself to compute the growth of the interface. The vorticity field generated from the model could be used as an initial condition of a time-dependent incompressible Euler simulation to compute the growth rates for these cases. This approach has been followed by Pham and Meiron [36]. This does require a second code, however, and increases the amount of computational effort needed to obtain a growth rate from the model. For simplicity, we restrict ourselves to small amplitude interfaces, where the growth rate oscillates about a constant value, to assess the accuracy of the impulsive approximation for strong shocks.

Each figure shows comparisons of the growth rate and circulation for a fixed interface configuration specified by At and L. Each row corresponds to a specific incident shock strength. Thin layer cases ($L = 0.01$) are not considered because the results were very similar to the $L = 0.1$ cases. As in §3.3, two additional grid levels were used in the AMR grid of the Euler simulations for $L = 0.1$ cases and no refinement was used for $L = 0$ cases.

Computational parameters used in numerical solution of the impulse model are shown

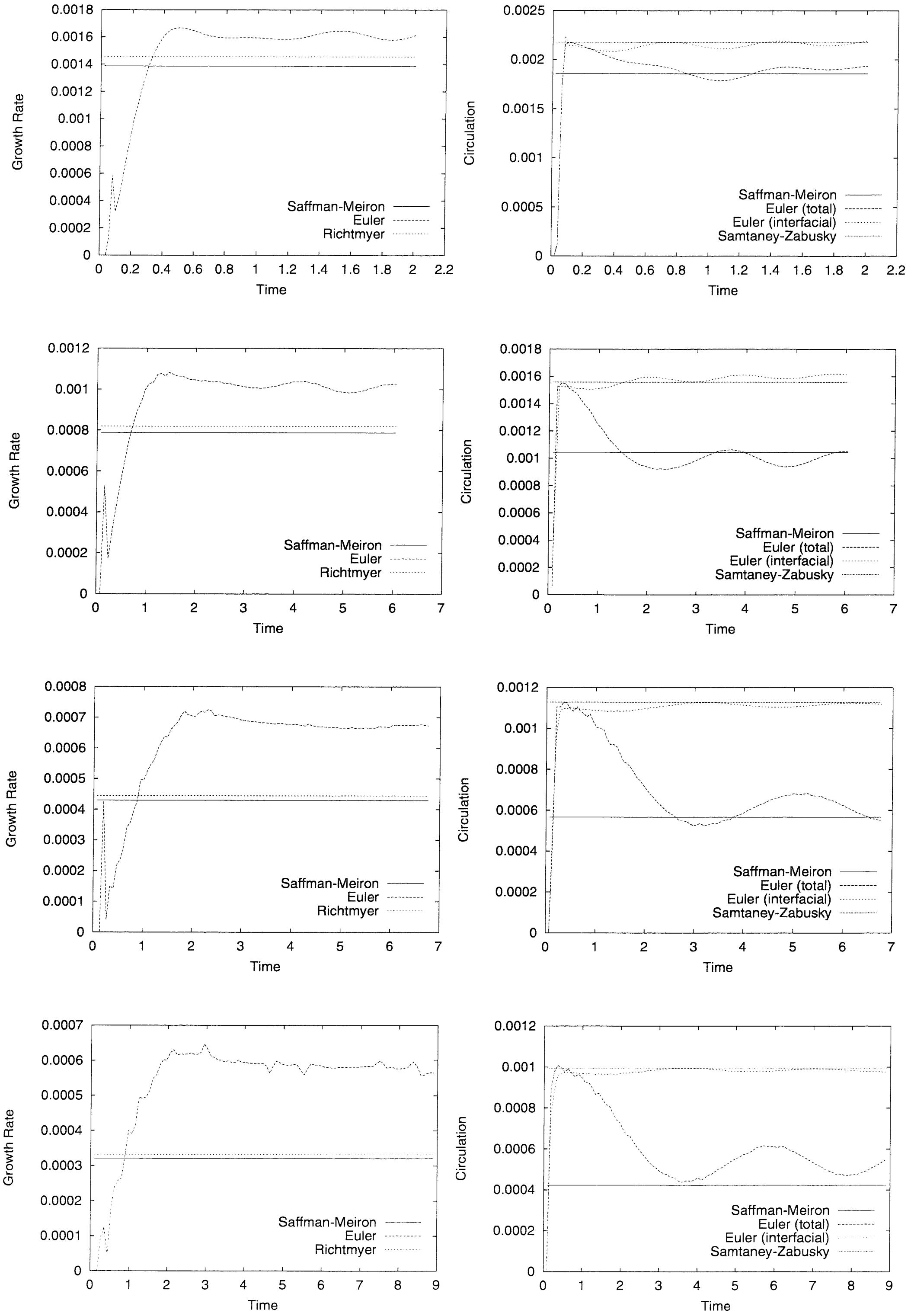


Figure 3.7: Comparisons between models and Euler simulations for $At = 0.2$, a_0=0.01, $L = 0.1$. (Rows: (top to bottom) $M_I = 1.25$, $M_I = 2$, $M_I = 4$, and $M_I = 8$)

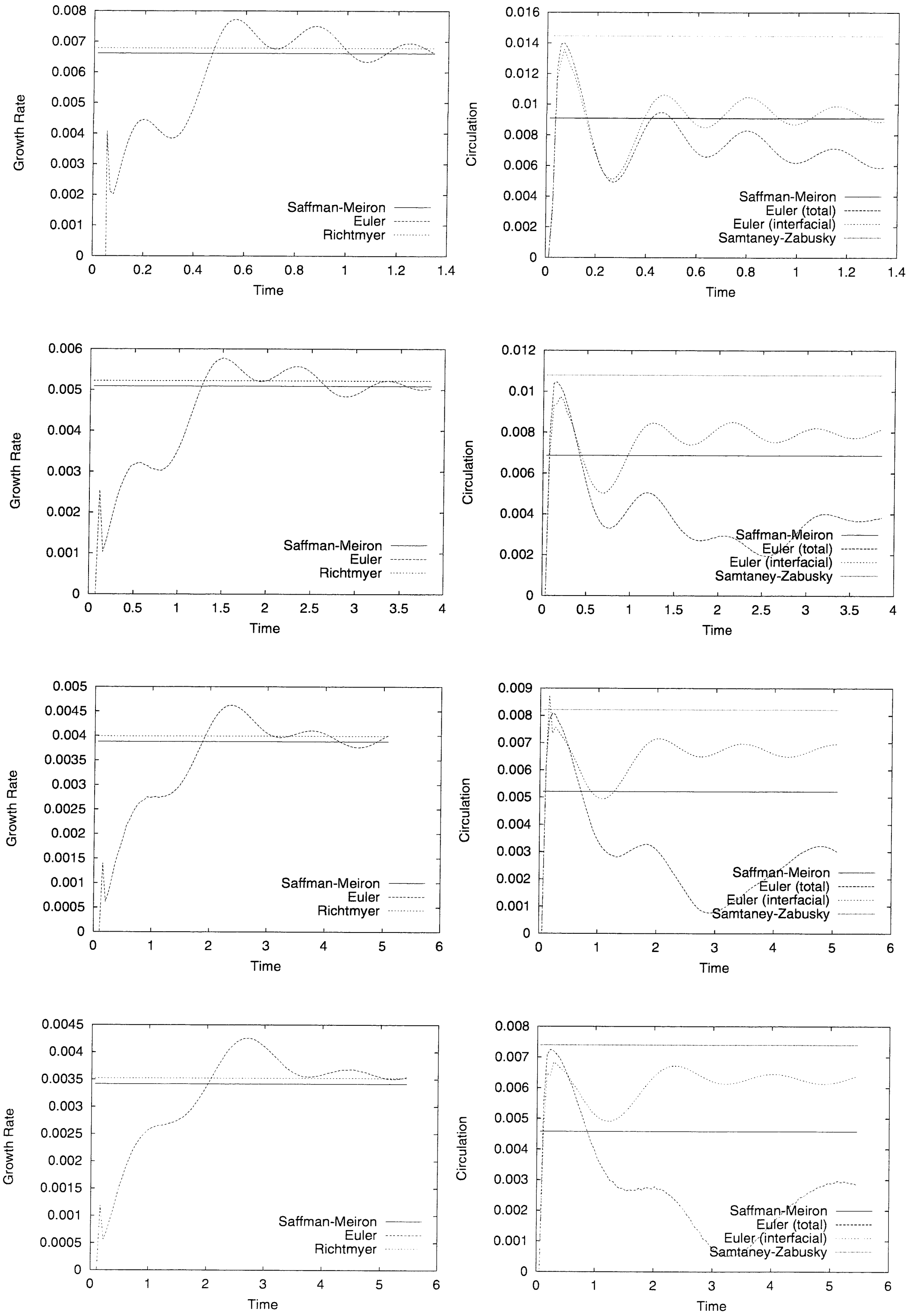


Figure 3.8: Comparisons between models and Euler simulations for $At = 0.8$, a_0=0.01, $L = 0.1$. (Rows: (top to bottom) $M_I = 1.25$, $M_I = 2$, $M_I = 4$, and $M_I = 8$)

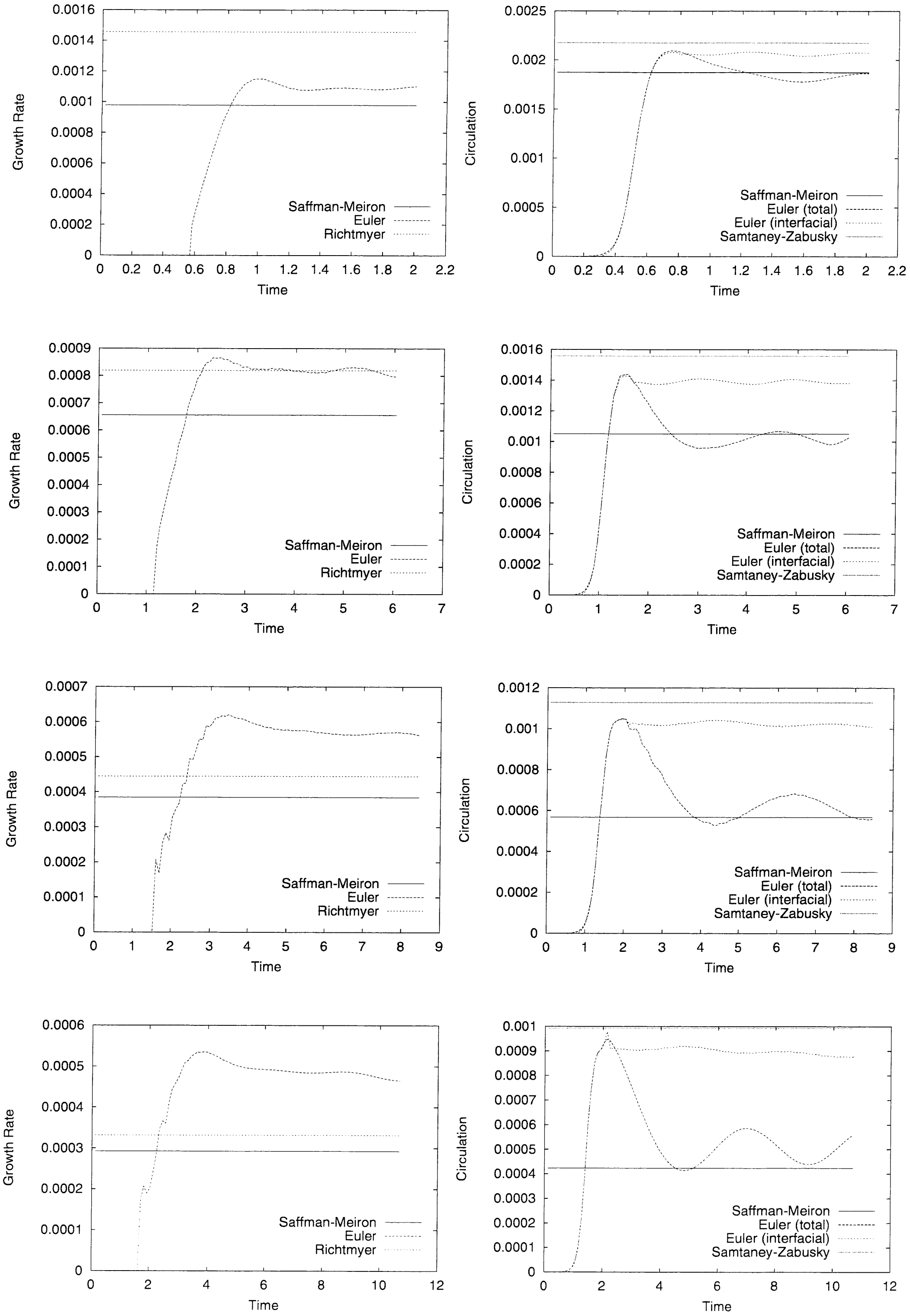


Figure 3.9: Comparisons between models and Euler simulations for $At = 0.2$, a_0=0.01, $L = 1$. (Rows: (top to bottom) $M_I = 1.25$, $M_I = 2$, $M_I = 4$, and $M_I = 8$)

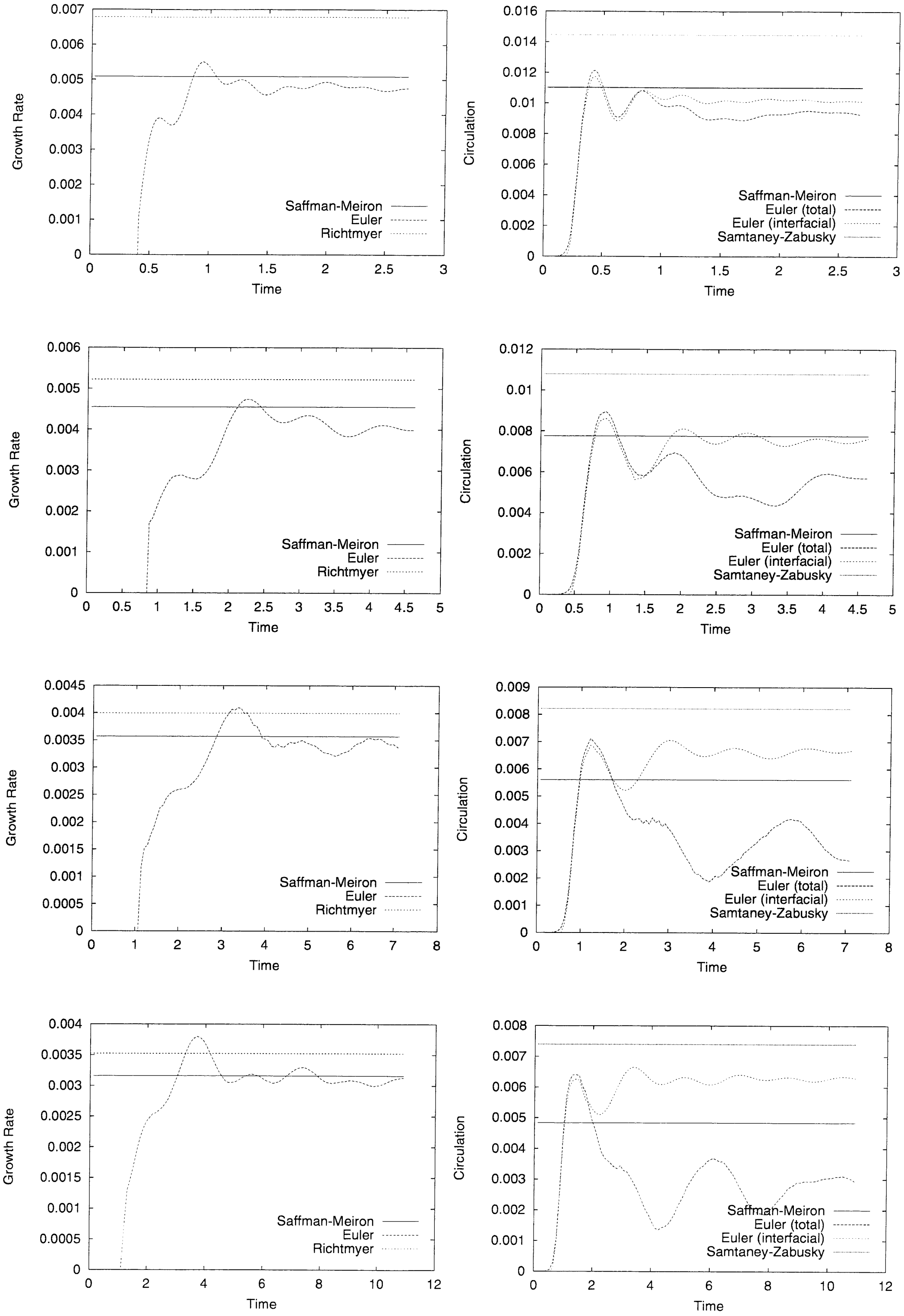


Figure 3.10: Comparisons between models and Euler simulations for $At = 0.8$, a_0=0.01, $L = 1$. (Rows: (top to bottom) $M_I = 1.25$, $M_I = 2$, $M_I = 4$, and $M_I = 8$)

Table 3.2: Impulse model computational parameters for strong shocks.

M_I	At	At_{post}	$a_{0,\text{post}}$	L	L_{post}	γ	$\eta_{\max}$
1.25	0.2	0.19943	7.3017e-3	0.1	0.070	1.0	0.9990
				1	0.670	0.3	0.9000
	0.8	0.79861	8.4943e-3	0.1	0.065	1.0	0.9990
				1	0.600	0.3	0.9000
2	0.2	0.18807	4.3554e-3	0.1	0.045	1.5	0.9995
				1	0.350	0.5	0.9900
	0.8	0.77226	6.7625e-3	0.1	0.045	1.5	0.9995
				1	0.300	0.5	0.9900
4	0.2	0.15360	2.8947e-3	0.1	0.038	1.5	0.9995
				1	0.220	0.8	0.9990
	0.8	0.69627	5.7373e-3	0.1	0.045	1.5	0.9995
				1	0.230	0.8	0.9990
8	0.2	0.13190	2.5153e-3	0.1	0.037	1.5	0.9995
				1	0.190	0.8	0.9990
	0.8	0.64990	5.4192e-3	0.1	0.045	1.5	0.9995
				1	0.210	0.8	0.9990

in table 3.2. While the incident shock strength did not enter into the impulse model computations for weak shocks, here the value is factored in indirectly through the post-shock parameters At_{post}, $a_{0,\text{post}}$ and L_{post}. Parameters At, a_0 and L listed in the table are the pre-shock values. All computations were performed on a 6000x40 grid.

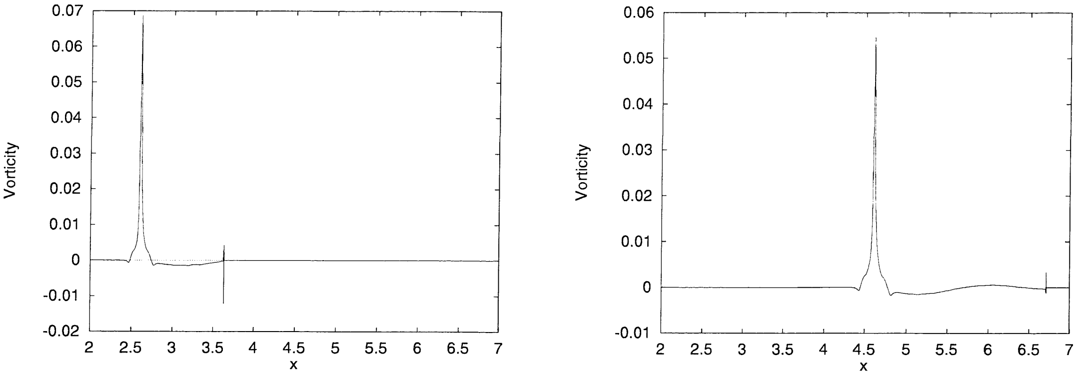
The growth rate comparisons for $At = 0.2$ cases in figures 3.7 and 3.9 show the agreement between the impulse approximation and the Euler simulations is not good. Even for shocks as weak as $M_I = 1.25$ the relative error is about 15%. Stronger shocks show worse agreement. For $At = 0.8$ cases, however, the impulse model predicts the late time growth rate much more accurately. Shocks as strong as $M_I = 8$ show good agreement for both thin ($L = 0.1$) and thick ($L = 1$) layers. These observations are similar to those communicated by Yang *et al.* [69] who compared growth rates calculated from Richtmyer's impulsive model to solutions of linearized Euler equations for interfaces of zero thickness.

For the $L = 1$, $At = 0.8$ cases shown in figure 3.10, the growth rates computed from the Saffman-Meiron model were in better agreement with Euler simulations than those from the Richtmyer formula provided post-shock values of L were used. For example, using $L_{\text{post}} = L = 1$ in the model for the $M_I = 8$, $At = 0.8$, $L = 1$ case resulted in a growth rate of $\dot{a}_{\text{model}} = 0.0021$ in contrast to the much more accurate $\dot{a}_{\text{model}} = 0.0032$ calculated using

$L_{\text{post}} = 0.21$.

The circulation comparisons in the right columns of figures 3.7 and 3.9 show that the impulse model substantially under-predicts the interfacial circulation for low Atwood ratio cases. This is not surprising in light of the fact that the model also under-predicts the growth rate. Remarkably, the circulation from the model is quite close to the asymptotic value of total circulation in the region with the exception of the $M_I = 8$ cases. It is not at all clear why the impulse model should capture this value. The initial drop in total circulation for $At = 0.2$ cases is due to the creation of a large region of negative vorticity by the oscillation of the transmitted shock. As the shock oscillates, it lays down patches of vorticity in the flow behind of alternating sign. This is shown in the two pictures of figure 3.11 where vorticity along the line $y = \lambda/4$ is plotted for the $M_I = 2$, $L = 0.1$ case. In figure 3.11(a) the deposition of negative vorticity behind the transmitted shock (the shock is the thin spike of vorticity – a numerical artifact resulting from the use of central differencing to compute $\omega_{i,j}$ – located at $x \approx 3.6$) is seen. Later in the simulation, the shock generates positive vorticity and then more negative vorticity as shown in figure 3.11(b). It is this additional generation of vorticity that is responsible for the temporal oscillation of the total circulation about an average value. Note that these oscillations are not, in general, in phase with the oscillations of the interfacial circulation which are caused by the action of pressure waves modifying the vorticity distribution in the layer. While it is tempting to infer that the small difference between the impulse model circulation and the average value of the total circulation from Euler simulations for $At = 0.2$ cases is meaningful, we do not see how this is possible.

The plots of circulation for $At = 0.8$ cases (figures 3.8 and 3.10) show that, in general, the model does not calculate the correct interfacial or total circulation. This is surprising in light of the good agreement in growth rate for these cases. Had the impulse model correctly calculated the vorticity distribution in the layer, the circulation from the model would have agreed quite well with the average interfacial circulation from Euler simulations. It would appear, on the basis of these results, that the impulse model has somehow calculated a vorticity distribution which is not correct but which has the same net effect on the interface growth rate as the true distribution. The difference between the interfacial and total circulations from Euler simulations indicates the presence of substantial vorticity away from the layer which is not accounted for in the impulse model.



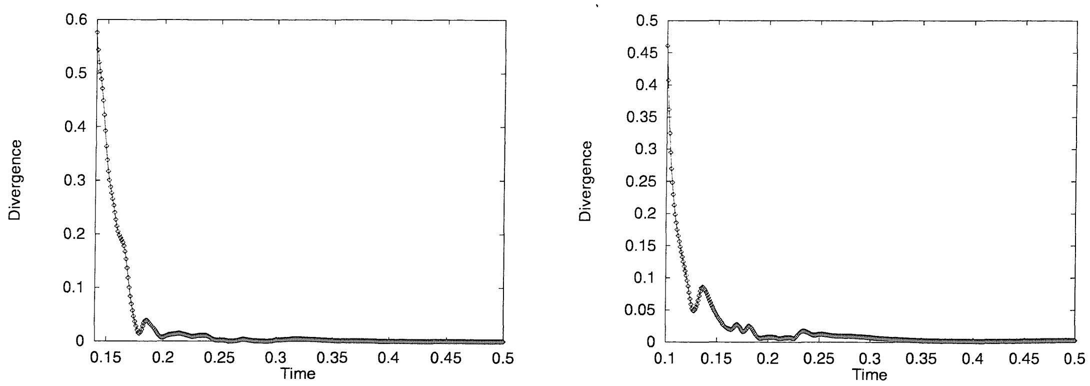
(a) (b)

Figure 3.11: Vorticity along the line $y = \lambda/4$. As the transmitted shock propagates, it deposits vorticity in the fluid behind it. ($M_I = 2$, $At = 0.2$, $a_0 = 0.01$, $L = 0.1$, (a) $t = 2$, (b) $t = 4$)

3.5 Assumptions of the Impulse Model

In light of the disagreement between the model and Euler simulations in the previous section, it is clear we must take a closer look at the underlying assumptions of the impulse model. The model simulates the growth of the shocked density layer as a band of vorticity evolving under its own power in an incompressible fluid. Any vorticity in the surrounding flow is ignored. The effects of compressibility are only accounted for by using post-shock values of parameters in the model. The large difference between interfacial and total circulations shown in figures 3.7-3.10 as well as the vorticity profiles in figure 3.11 reveal the actual flow definitely contains vorticity in the bulk of the fluid. The effect this vorticity may have on the growth rate of the interface is not clear and will be studied in this section. Before we do this, the assumption of incompressibility is examined by calculating the divergence of the fluid velocity in the Euler simulations.

We restrict our attention to the divergence along the interface, defined as the locus of points such that $\psi(x, y) = 0$. To plot the time evolution, an average value of the divergence along the interface, $|\nabla \cdot \mathbf{u}|_{\text{avg}}$, is computed to obtain a single number at each time. The discrete divergence in each cell is calculated using centered differences. To compute $|\nabla \cdot \mathbf{u}|_{\text{avg}}$, a sweep through all the cells in the finest grid level is done to locate the cells which straddle the curve $\psi(x, y) = 0$. Linear interpolation is done to compute values of $|\nabla \cdot \mathbf{u}|$ directly on the curve. These values are summed and the result divided by the number of points



(a) (b)

Figure 3.12: Time evolution of $|\nabla \cdot \mathbf{u}|_{\text{avg}}$ ($M_I = 2$, $a_0 = 0.01$, $L = 0.1$ and (a) $At = 0.2$, (b) $At = 0.8$).

to obtain the average discrete divergence along the interface. It should be noted that the discrete divergence does not converge at the numerically smeared shock. The shock is smeared over a set number of grid cells by the flow solver regardless of the values of Δx and Δy for those cells. The numerical thickness of the shock is self-similar with grid spacing. As the computational grid is refined, the discrete velocity differences (i.e.: $u_{i+1,j} - u_{i-1,j}$) do not change much whereas the discrete spatial differences (i.e.: $x_{i+1,j} - x_{i-1,j}$) decrease to zero.

The evolution of $|\nabla \cdot \mathbf{u}|_{\text{avg}}$ for two problems with parameters $M_I = 2$, $a_0 = 0.01$, and $L = 0.1$ is shown in figure 3.12. Figures 3.7 and 3.8 have shown the impulse model predicts the growth rate accurately for the $At = 0.8$ case but not the $At = 0.2$ case. In figure 3.12 it is seen that the interfacial divergence decays very rapidly for both Atwood ratios. Care was taken to plot only the portion of each curve that converged with increasing grid resolution. At times earlier than those shown in the graphs, the numerically smeared profile of the shock affected the values of $|\nabla \cdot \mathbf{u}|_{\text{avg}}$ calculated. For both cases the divergence drops to $|\nabla \cdot \mathbf{u}|_{\text{avg}} \approx 0.01$ as early as $t \approx 0.2$. Comparing this time with the growth rate plots of figures 3.7 and 3.8, it is seen that compressibility on the interface is restricted to very early times. Mach numbers as high as $M_I = 8$ were tested and $|\nabla \cdot \mathbf{u}|_{\text{avg}}$ was found to decay to very small values well before the interface began the linear growth phase in all cases.

The plots in figure 3.12 track the divergence only on the $\psi = 0$ curve. The divergence was found to be very small throughout most of the flow field with non-negligible values occurring only very near the transmitted and reflected shocks. As the shocks move away from the interface, the divergence in the majority of the flow field dropped to values on the order of $|\nabla \cdot \mathbf{u}| \approx 0.001$. The assumption of incompressibility in the impulse model appears to be quite accurate.

Since the flow is almost entirely incompressible except for a very small region around the shocks, the evolution of the Richtmyer-Meshkov instability should be almost entirely driven by vorticity. Impulse models assume this vorticity to lie entirely in the density layer, but it has already been shown that this is not the case. We now seek to measure the influence of the unaccounted for vorticity on the growth rate of the interface.

Under certain circumstances [48], the most important being that the flow is divergence free, the fluid velocity is given uniquely by the sum of a solenoidal vector component and an irrotational scalar component. The vector component, $\mathbf{u} = [u, v]$, is given by the Biot-Savart law:

$$u - iv = \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\omega(x', y', t)\left[(y' - y) - i(x - x')\right]}{(x - x')^2 + (y - y')^2} dy' dx'. \tag{3.17}$$

Or, rewriting the kernel with the complex variables $z = x + iy$ and $z' = x' + iy'$,

$$u - iv = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\omega(z', t)}{z - z'} dy' dx'. \tag{3.18}$$

At any finite time, the vorticity field of the Richtmyer-Meshkov problem is finite in extent in the x-direction (bounded by the transmitted and reflected shocks) but infinite in the y-direction due to periodicity. To obtain a convergent integral, the integral over the infinite interval in y' is replaced with an infinite sum of finite integrals over one period:

$$u - iv = \frac{1}{2\pi i} \int_{-\infty}^{\infty} \int_{0}^{\lambda} \omega(z', t) \left(\sum_{n=-\infty}^{\infty} \frac{1}{z - z' - \lambda n i} \right) dy' dx'. \tag{3.19}$$

The infinite sum can be rewritten as a cotangent [1]. Algebraic manipulation results in the following, rewritten in terms of x and y:

$$u - iv = \frac{1}{4} \int_{-\infty}^{\infty} \int_{0}^{\lambda} \omega(x', y', t) \frac{\sin\left[\frac{2\pi}{\lambda}(y' - y)\right] - i \sinh\left[\frac{2\pi}{\lambda}(x - x')\right]}{\cosh\left[\frac{2\pi}{\lambda}(x - x')\right] - \cos\left[\frac{2\pi}{\lambda}(y' - y)\right]} dy' dx'. \tag{3.20}$$

In principle, the growth rate could be calculated by subtracting velocities at the peak and trough of the interface, evaluated using (3.20) if the vorticity field was known. Ignoring the scalar irrotational component of the fluid velocity is equivalent to viewing the instability in a frame of reference moving with the mean speed of the interface, V. Impulse models can be thought of as a method to compute the vorticity field without explicitly computing the shock refraction through the density layer.

The two-dimensional trapezoidal rule was used to numerically calculate the integral (3.20) using the vorticity field in the Euler simulations. The vorticity in each grid cell was computed using central differences. Although this may appear to be a circular exercise – using the discrete velocities to calculate vorticity which is then used to calculate velocities through the Biot-Savart law – this was done to determine if the growth rate could be calculated from the Biot-Savart law which holds only if the flow is incompressible. We seek an answer to the question: is it sufficient to describe the Richtmyer-Meshkov instability solely in terms of a vorticity field?

One modification must be made to (3.20), however, to account for the fact that only half a wavelength was simulated in the Euler simulations. The range of integration is restricted to $y' \in [0, \lambda/2]$ and symmetry in the problem is used to account for the vorticity in $y' \in (\lambda/2, \lambda]$. Using $\omega(x, y, t) = -\omega(x, \lambda - y, t)$, the results are:

$$u(x,y) = \frac{1}{4}\int_{-\infty}^{\infty}\int_{0}^{\lambda/2} f_u(x,y,x',y')dy'dx', \tag{3.21a}$$

$$v(x,y) = \frac{1}{4}\int_{-\infty}^{\infty}\int_{0}^{\lambda/2} f_v(x,y,x',y')dy'dx', \tag{3.21b}$$

$$f_u(x,y,x',y') = \omega(x',y',t)\left\{\frac{\sin\left[\frac{2\pi}{\lambda}(y'-y)\right]}{D(x,y,x',y')} - \frac{\sin\left[\frac{2\pi}{\lambda}(\lambda-y'-y)\right]}{D(x,y,x',\lambda-y')}\right\}, \tag{3.21c}$$

$$f_v(x,y,x',y') = \omega(x',y',t)\sinh\left[\frac{2\pi}{\lambda}(x-x')\right]\left\{\frac{1}{D(x,y,x',y')} - \frac{1}{D(x,y,x',\lambda-y')}\right\}, \tag{3.21d}$$

$$D(x,y,x',y') = \cosh\left[\frac{2\pi}{\lambda}(x-x')\right] - \cos\left[\frac{2\pi}{\lambda}(y'-y)\right]. \tag{3.21e}$$

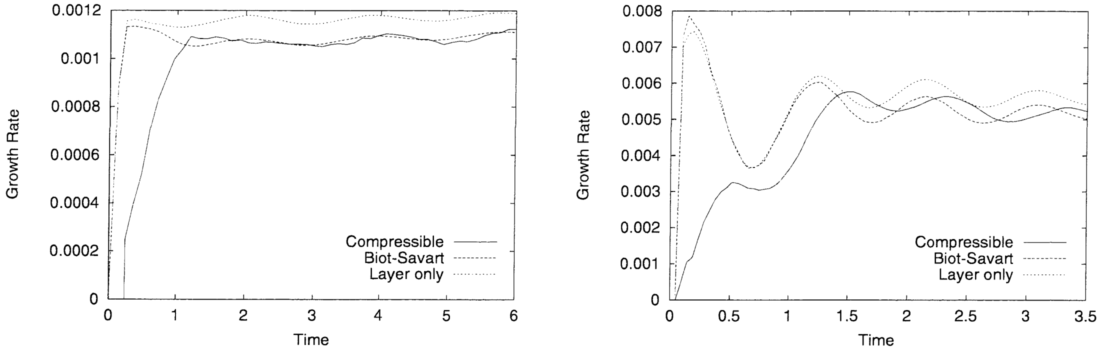
The data on the finest grid at any location (x', y') was used in the calculation of the integrals in the same manner as in the calculation of Γ_{int} in §3.4. Identifying the peak and trough of the interface by (x_p, y_p) and (x_t, y_t), respectively, the incompressible growth rate was computed using the following:

$$\dot{a}_{\text{biot}}(t) = \sum_{l=0}^{l_{\max}} \Delta x_l \Delta y_l \sum_{k=1}^{\text{NGA}_l} \sum_{\substack{i,j \\ \text{flag}_{i,j}=0}} \left[f_u(x_p, y_p, x_{i,j}, y_{i,j}) - f_u(x_t, y_t, x_{i,j}, y_{i,j}) \right], \tag{3.22}$$

where $l_{\max}$ is the number of grid levels in the AMR grid hierarchy (see appendix A), NGA_l is the number of grid patches on level l, and $\text{flag}_{i,j}$ identifies which cells have been flagged for refinement. The innermost summation loops over all i and j in mesh patch k of grid level l. Note that in contrast to the well-known point vortex implementation of the Biot-Savart law, the velocities are not evaluated at vortex nodes so it is not necessary to skip over any vortex nodes in the sum to avoid the singularity in the integrand. The use of Richardson extrapolation is a popular way to attain a high order of formal accuracy in point vortex methods. We have chosen not to implement this since the use of weighting functions in the sum which are functions of the computational indices of the vortex nodes would be cumbersome to implement on the AMR grid. A high enough spatial resolution was used to obtain converged results for $\dot{a}_{\text{biot}}(t)$. The needed resolution was comparable to that needed to compute the growth rate using velocities from the Euler code.

The calculation of the growth rate from the Biot-Savart law allows determination of the contribution of bulk vorticity on the growth rate. This was accomplished simply by placing a restriction on which vortex contributions to consider in the calculation of (3.22) in the same manner as was done for calculation of Γ_{int} in §3.4. To compute the growth rate using only interfacial vorticity, the restriction $|\psi_{i,j}| \leq 0.99$ is added to the innermost sum in (3.22). Any difference between the calculations with and without the $|\psi_{i,j}| \leq 0.99$ restriction is the result of neglect of the vorticity not in the layer.

Figure 3.13 compares the growth rates computed using velocities from the Euler simulation ("Compressible") as was done in figures 3.7-3.10, the incompressible formula (3.22) ("Biot-Savart"), and the restricted Biot-Savart calculation over the region $|\psi(x,y)| \leq 0.99$ ("Layer only") for parameters $M_I = 2$, $a_0 = 0.01$, and $L = 0.1$. At early times there is clear disagreement between the compressible and incompressible results due to the influence of the shocks on the interface. The asymptotic growth rate, however, is calculated quite accurately from the Biot-Savart law for both Atwood ratios. This provides strong evidence that the instability is vorticity-dominated at late times.



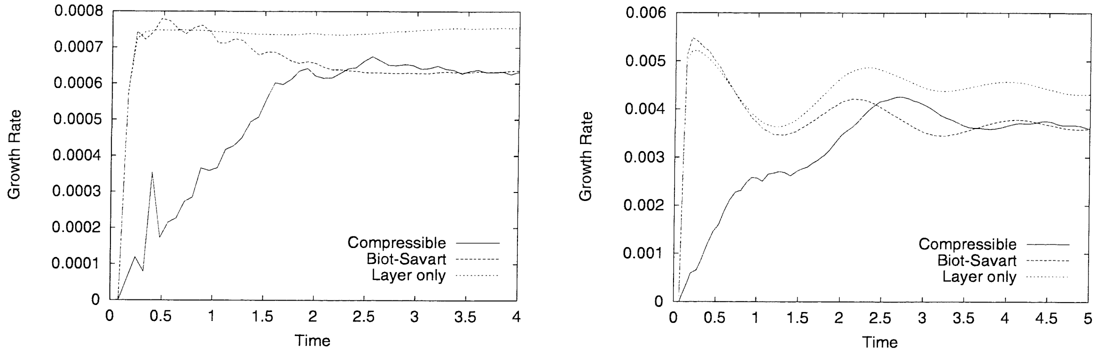
(a) (b)

Figure 3.13: Calculation of growth rate from compressible and incompressible calculations. ($M_I = 2$, $a_0 = 0.01$, $L = 0.1$, (a) $At = 0.2$, (b) $At = 0.8$)

Comparing the two incompressible results, it is seen that the vorticity away from the layer plays a definite role in the determination of the growth rate, although a small one. For both cases, the neglect of the off-layer vorticity by the "Layer only" calculation creates an 8-9% over-prediction. The reduction in growth rate due to inclusion of bulk vorticity is because the largest patch of vorticity away from the layer, and the one closest to the interface, is one of negative vorticity laid down by the transmitted shock.

Temporal oscillations of the growth rate in the $At = 0.8$ case are present in the incompressible calculations but are out of phase with those of the compressible results. These oscillations are due to pressure perturbations in the fluids generated by the corrugated shocks as they stabilize themselves. These pressure waves modify the velocities at the interface both by modifying the vorticity distribution in the layer (see §3.6) through baroclinic interactions and by creating pressure forces. We cannot offer an explanation for the time-lag between oscillations of the compressible and incompressible growth rates and believe it to be due to some secondary feature not accounted for in the incompressible theory. Increasing the resolution of the Euler simulation, however, did not bring the two curves back into phase and we are inclined to believe the phase lag is real.

Although the discussion in this section so far has focused on the case $M_I = 2$, $a_0 = 0.01$, and $L = 0.1$, similar results are obtained for thicker layers and stronger shocks. Increasing the Mach number from $M_I = 2$ to $M_I = 8$ does not alter the fact that the asymptotic value of the growth rate is dictated by the vorticity distribution. The results from the



(a) (b)

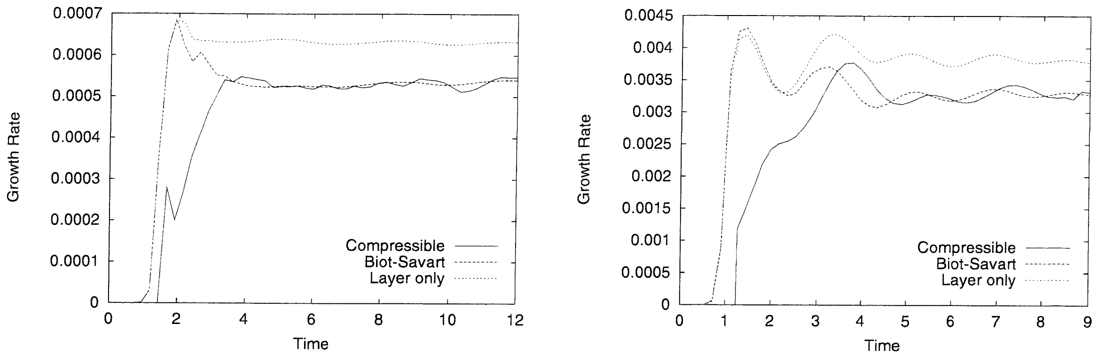
Figure 3.14: Calculation of growth rate from compressible and incompressible calculations. ($M_I = 8$, $a_0 = 0.01$, $L = 0.1$, (a) $At = 0.2$, (b) $At = 0.8$)

$M_I = 8$ simulations are shown in figure 3.14. Neglect of the bulk vorticity, however, causes larger errors. The relative error between asymptotic values of growth rate between the two Biot-Savart calculations is about 20%. The results for a layer of initial thickness $L = 1$ are given in figure 3.15.

The assumption of incompressibility made in impulse models is valid for the late time growth of Richtmyer-Meshkov instabilities even if generated by very strong incident shocks. The asymptotic growth rate of the Richtmyer-Meshkov instability for small amplitude interfaces is determined by the vorticity distribution. However, the vorticity does not lie entirely within the layer (as the model assumes) and neglect of vorticity in the bulk of the fluid leads to incorrect calculations of the growth rate. This neglected vorticity, however, is not responsible for the failure of the impulse model to accurately compute the growth rate for $At = 0.2$ cases, as seen in §3.4. In fact, the influence of the negative vorticity outside the layer decreases the discrepancy. We now turn our attention to the vorticity distribution in the shocked layer before considering the case of large amplitude interfaces.

3.6 Distribution of Vorticity

When the Richtmyer-Meshkov instability occurs in continuously stratified fluids the shocked interface is a region of continuous, bounded vorticity rather than a vortex sheet as in the classical problem. Samtaney and Pullin [53] have conjectured that the self-similar

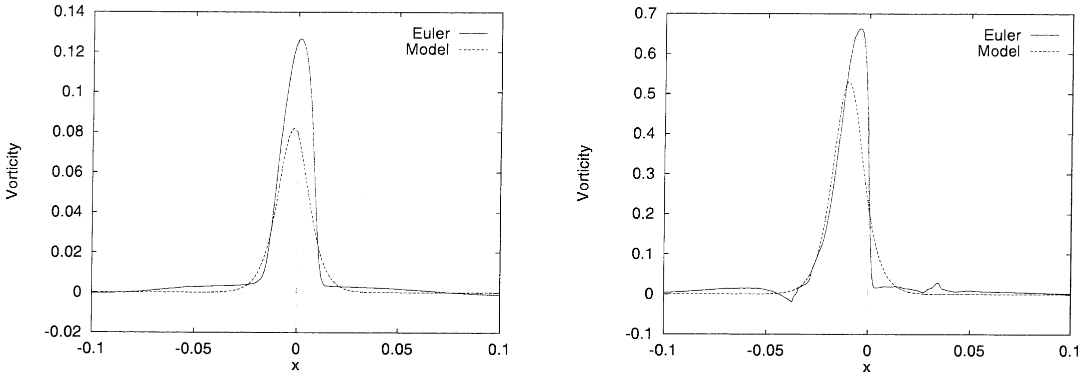


(a) (b)

Figure 3.15: Calculation of growth rate from compressible and incompressible calculations. ($M_I = 8$, $a_0 = 0.01$, $L = 1$, (a) $At = 0.2$, (b) $At = 0.8$)

Richtmyer-Meshkov problem does not converge to a weak solution of the Euler equations as the computational mesh is refined because of the presence of the vortex sheet. In the problem at hand, numerical convergence of pointwise values of vorticity in the layer can be achieved and those values can be compared directly against the Saffman-Meiron model.

The spatial resolution required to obtain convergence of $\omega_{i,j}$ (calculated using central differences) in the layer was found to be significantly more than that needed to obtain convergence of the growth rate and circulation. Numerical viscosity in the Euler simulations caused the shocked layer to decompress and diffuse after the passage of the shock. As the layer spreads out, the magnitude of vorticity in the layer decreases. Pointwise convergence of the vorticity field, then, is not a requirement to obtain accurate values of the growth rate in the Euler simulations. In fact, this must be true for if pointwise convergence was required, accurate calculation of the growth rate for the classical, sharp interface Richtmyer-Meshkov problem would not be possible using shock capturing Euler codes. Accurate calculation of the growth rate depends on computing an "effective vorticity distribution" correctly. It was necessary to use a finest-grid-level resolution of $\lambda/3200$ using the van Leer flux limiter (see §2.3) to get pointwise convergence of vorticity for $L = 0.1$ layers. Adaptive mesh refinement made these calculations feasible. Additional savings were realized by noting that much less resolution was needed in the y direction. The numerical diffusion in any cell depends on, among other factors, the density gradient across a cell face. For the small amplitude cases considered here, the variation in the y-direction is quite small and high



(a) (b)

Figure 3.16: Slices of vorticity along mid line of domain. ($M_I = 2$, $a_0 = 0.01$, $L = 0.1$ and (a) $At = 0.2$, (b) $At = 0.8$)

resolution in the y-direction was not required.

The vorticity distribution predicted from the Saffman-Meiron impulse model can be calculated from $\nabla \times [u_0, v_0]$ using equations (3.10a) and (3.10b):

$$\omega_{\text{model}} = \frac{1}{\rho_0^2}\left(\frac{\partial \rho_0}{\partial x}\frac{\partial p'}{\partial y} - \frac{\partial \rho_0}{\partial y}\frac{\partial p'}{\partial x} + V\bar{\rho}\frac{\partial \rho_0}{\partial y}\right). \tag{3.23}$$

To compare the pointwise vorticity computed from the model and Euler simulations, slices of the vorticity field along $y = \lambda/4$ were compared. These comparisons are shown in figure 3.16 for $M_I = 2$, $a_0 = 0.01$, $L = 0.1$ and both Atwood ratios. The Euler profiles were aligned with the impulse model profiles by shifting the data in the x-direction such that the zero level set of $\psi(x, y)$ passed through $x = 0$ at $y = \lambda/4$. The data from the Euler profiles in the figure was taken at $t = 2.88$ in the $At = 0.8$ case and $t = 2.7$ in the $At = 0.2$ case – moments in the simulations where the Euler growth rate was close to its asymptotic value. In both cases, the impulse model clearly under-predicts the vorticity. The $At = 0.2$ case shows worse agreement than the $At = 0.8$ case. This is consistent with the circulation comparisons in figures 3.7 and 3.8 where the disagreement between interfacial and model circulation was larger for $At = 0.2$ than $At = 0.8$.

Note the location of peak vorticity in figure 3.16 is not at $x = 0$ in the model or the Euler simulation. The coordinate $x = 0$ in these figures corresponds to the intersection of the interface with the line $y = \lambda/4$ and, so, the maximum values of vorticity do not

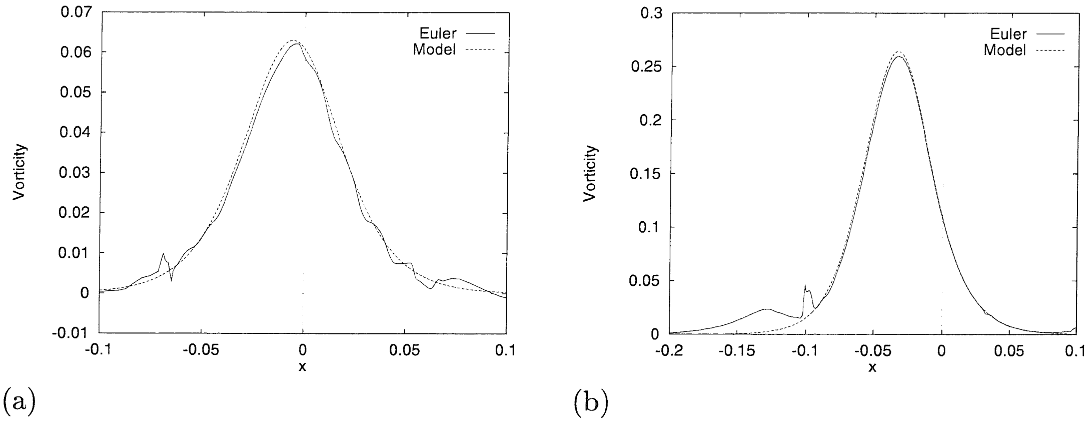


Figure 3.17: Slices of vorticity along mid line of domain. ($M_I = 1.01$, $a_0 = 0.01$, $L = 0.1$ and (a) $At = 0.2$, (b) $At = 0.8$)

occur exactly on the interface curve. A "forward lean" in the Euler profiles can be seen in the figure. The right edge of the layer is steeper than the left. We have noted that this lean increased slightly with time in the Euler simulations. Fearing this to be a numerical artifact, we performed the $At = 0.8$ simulation in both the "laboratory frame" (where the shocked layer moves with mean velocity V) and a frame moving with the interface. The vorticity profiles of both simulations had a lean. Changes in resolution also did not alter the profile. We believe the departure from symmetry may be the result of the pressure perturbations generated by the oscillating shocks. Because the transmitted shock is much stronger than the reflected shock for this value of incident Mach number (see figure 1.9), the waves generated from the transmitted shock may result in more compression occurring on the right side of the layer than the left.

The $M_I = 1.01$ cases corresponding to those in figure 3.16 (with $M_I = 2$) are shown in figure 3.17. The profiles from the Euler simulations are in excellent agreement with those from the model throughout most of the layer. Note there is no forward lean to the Euler profiles as there was in figure 3.16. For $M_I = 1.01$ the strengths of the transmitted and reflected shocks are comparable and we believe this provides some justification to the theory that the lean in the $M_I = 2$ cases is due to anisotropic compression of the density layer. A numerical artifact is seen to the left of each profile, however. This error occurs at the boundary between a coarse mesh patch and a fine mesh patch on the AMR grid. We found that variations in the grid resolution used altered its shape. These artifacts occur only for

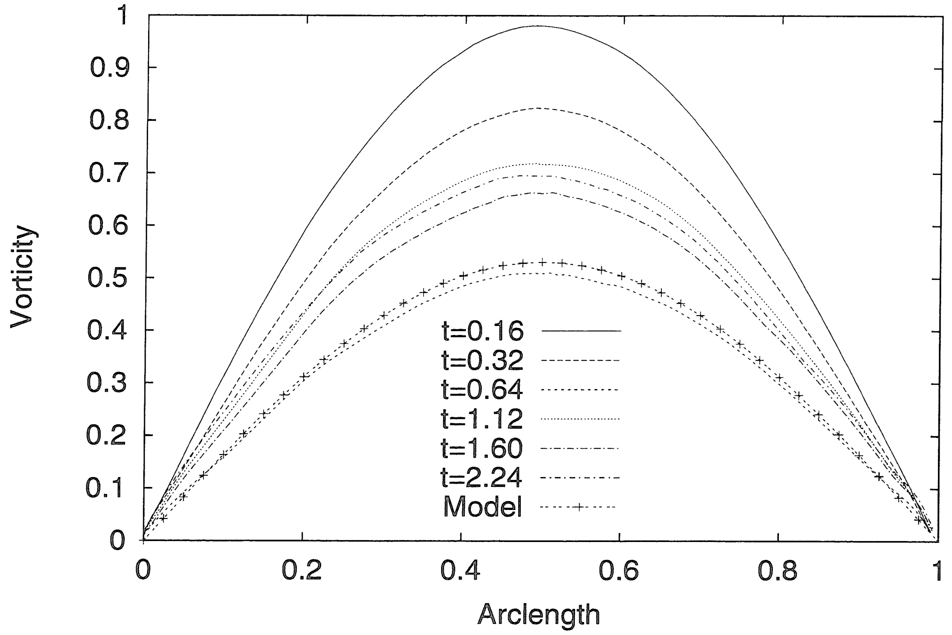


Figure 3.18: Time evolution of vorticity along a curve in the layer ($M_I = 2$, $At = 0.8$, $a_0 = 0.01$, $L = 0.1$)

the $M_I = 1.01$ case and we believe they do not affect the simulation significantly. The region of vorticity behind the $At = 0.8$ layer appears to be related to the anomaly at the coarse-fine grid boundary.

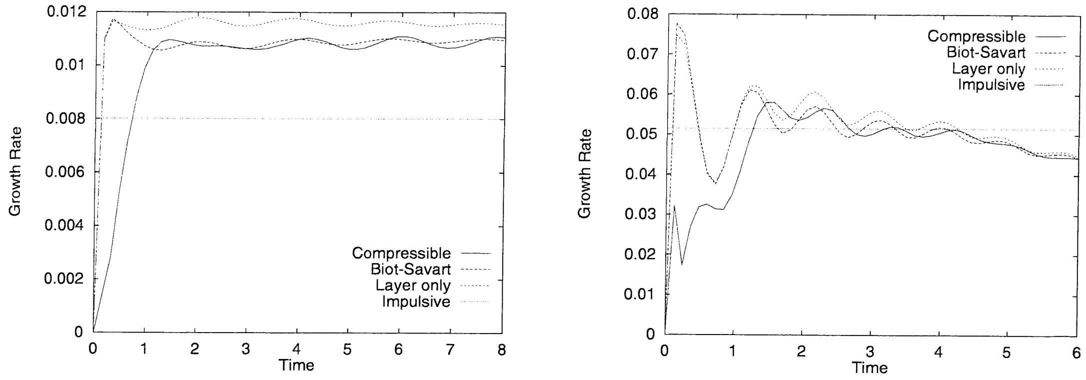
In general the vorticity in the layer changes little with time for the $At = 0.2$ cases. As a result, the interfacial circulation varies little with time (see figures 3.7, and 3.9). The vorticity in the high Atwood ratio case, however, does change in time due to the action of pressure waves originating from the corrugated shocks. These pressure perturbations modify the vorticity distribution through baroclinic interactions. In figure 3.18 the time evolution of vorticity for the $M_I = 2$, $At = 0.8$ case is shown. Each plot is the distribution of vorticity along the curve in the layer which passes through the point of maximum vorticity for each y-coordinate. This curve is not coincident with the interface curve, as defined in §3.2. While the graphs in figure 3.16 plotted vorticity as a function of x for a fixed value of y, the graph in figure 3.18 plots vorticity along a nearly vertical curve lying in the layer. At early times, the vorticity varies significantly with time. At late times, however, the profiles oscillate slightly about an average profile. The curve computed from the impulsive model is shown for comparison. As seen in figure 3.16(b), the impulse model under-predicts the vorticity.

The vorticity distribution in the layer is not computed accurately from the Saffman-Meiron model using post-shock parameters for either low Atwood or high Atwood cases. The success this model enjoys in predicting the asymptotic growth rate for large Atwood ratios appears to be due to a beneficial cancellation between the excess positive vorticity in the layer that the model does not calculate and the negative vorticity in the bulk of the fluid. While these conclusions apply to the Saffman-Meiron model of Richtmyer-Meshkov instability in continuously stratified fluids, we believe they carry over to Richtmyer's model for sharp interfaces as well. The small differences in growth rate between the Richtmyer model and the Saffman-Meiron model for $L = 0.1$ interfaces seen in figures 3.7 and 3.8 indicate that giving the layer a small finite thickness does not change the character of the instability much. We are inclined, therefore, to agree with Fraley [15] that the accurate prediction of the growth rate from the use of post-shock parameters in impulse models is not meaningful.

3.7 Large Amplitude Interfaces

Impulse models, by their very nature, contain no time variable and do not apply to large amplitude Richtmyer-Meshkov instabilities where the growth rate decays with time (see figure 1.4). The creation of vortex roll-up structures (as in figure 1.2) results in much of the kinetic energy of the flow being used to deform the layer rather than increase the peak-to-trough distance, thus, the growth rate does not have an asymptotic value. We now examine the compressibility and vorticity distribution for larger amplitude cases without reference to impulse models.

Growth rates for an interface of moderate amplitude, $a_0 = 0.1$, struck by a $M_I = 2$ shock are shown in figure 3.19. The initial amplitude is ten times as large as the problem shown in figure 3.13. The growth rate estimates from the Saffman-Meiron impulsive model are plotted in figure 3.19 for reference. Again, the Biot-Savart law accurately calculates the growth rate at late times indicating compressibility is only important in the initialization of the instability. The growth rate for the high Atwood ratio case clearly shows decay with time as nonlinearity begins to play a role in the evolution of the interface.



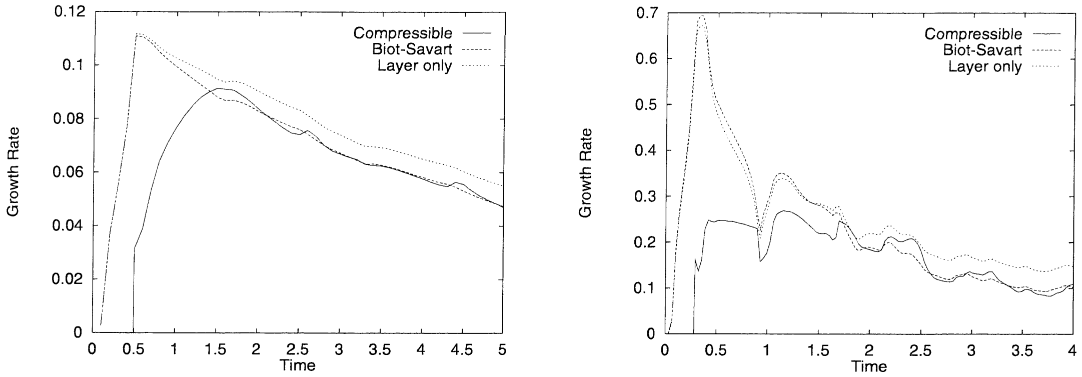
(a) (b)

Figure 3.19: Calculation of growth rate from compressible and incompressible theories. ($M_I = 2$, $a_0 = 0.1$, $L = 0.1$, and (a) $At = 0.2$, (b) $At = 0.8$)

The influence of off-layer vorticity is seen in the discrepancies between the full Biot-Savart calculation and that restricted to vorticity contributions in the shocked layer for the $At = 0.2$ case. The influence is not clear for the higher Atwood ratio case. Close examination of the flow field revealed the existence of regions of positive vorticity near but not within the layer, defined as the set of all points such that $|\psi(x, y)| \leq 0.99$. This unaccounted for positive vorticity cancels the effect of the negative vorticity deposited by the transmitted shock. The excess positive vorticity could be included in the "Layer only" calculation by increasing the cut-off value of ψ.

We now increase the initial amplitude by another order of magnitude to $a_0 = 1$. The growth rates for both Atwood ratios are shown in figure 3.20 where it is seen that the Biot-Savart law captures the decay of the growth rate. Figure 3.21 contains numerical Schlieren images of the two flows where a complex network of shocks and waves is clearly evident in the $At = 0.8$ case. The evolution of the interface remains a vortex dominated process, however, which incompressible theory models well.

The growth rate for problems with interfaces of large amplitude is determined primarily by vorticity just as for small amplitudes. In theory, the Richtmyer-Meshkov problem could be reduced to determining the correct vorticity distribution, which would then be used by incompressible theories to track the evolution of the interface. Determining the vorticity distribution a priori, however, is a daunting task. During the initial period where compressibility is important (identified by the time interval in figure 3.20(b) where the growth



(a) (b)

Figure 3.20: Calculation of growth rate from compressible and incompressible theories. ($M_I = 2$, $a_0 = 1$, $L = 0.1$, and (a) $At = 0.2$, (b) $At = 0.8$)

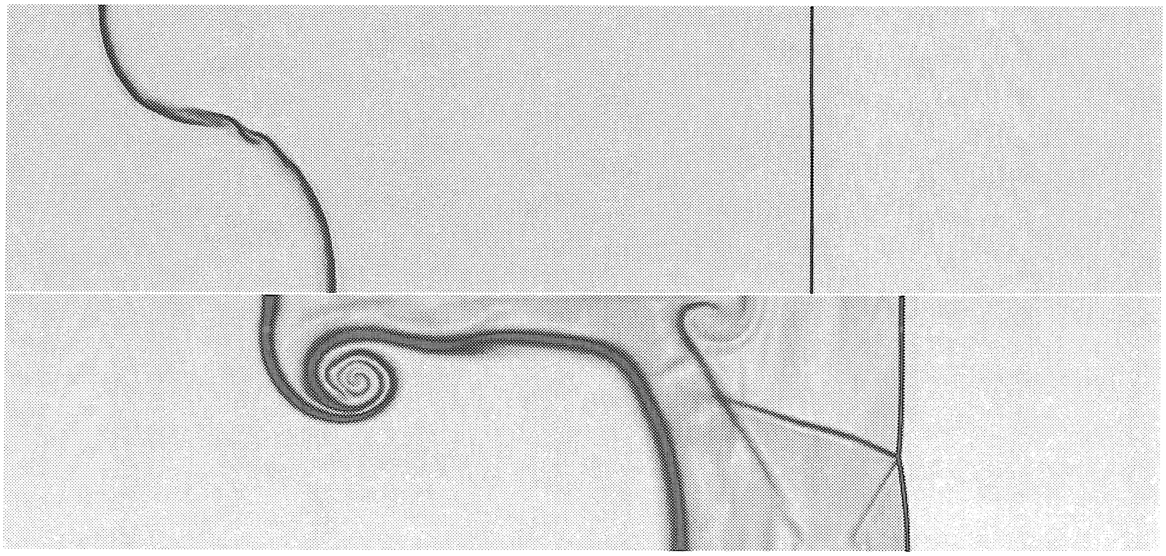


Figure 3.21: Schlieren-type images of the flow fields from simulations presented in figure 3.20. (Top: $At = 0.2$ at $t = 4$, Bottom: $At = 0.8$ at $t = 3.2$)

rate differs with the predictions of the Biot-Savart law), the action of reflected shocks of the triple points drastically alters the vorticity in the layer as shown in figure 3.22. Time evolution of vorticity along the curve passing through points of maximum vorticity (as in figure 3.18) is shown together with numerical Schlieren images of the flow. Secondary shocks sweep across the interface and modify the vorticity through baroclinic interactions. After the shocks have moved away and reflected shocks from triple points no longer intersect the layer, the vorticity distribution will continue to change under action from pressure waves and due to roll up of the interface.

We close by presenting the results of a Mach 8 shock striking an interface of initial amplitude $a_0 = 1$ and Atwood ratio of $At = 0.8$. The growth rate, plotted in figure 3.23, is well predicted by incompressible theory at late times in spite of the complexity of the flow shown in the accompanying Schlieren image. We have demonstrated that the Richtmyer-Meshkov growth rate can be obtained solely from the vorticity distribution regardless of incident shock strength or initial amplitude. However, the correct distribution is not obtained from popular impulsive models which neglect the effect of bulk vorticity on the interface. Furthermore, the interfacial vorticity distribution predicted by the impulsive model is only correct for very weak shocks. The development of a model which computes an accurate vorticity distribution will probably not be trivial due to the complexity of the Richtmyer-Meshkov flow field, especially for interfaces of large amplitude.

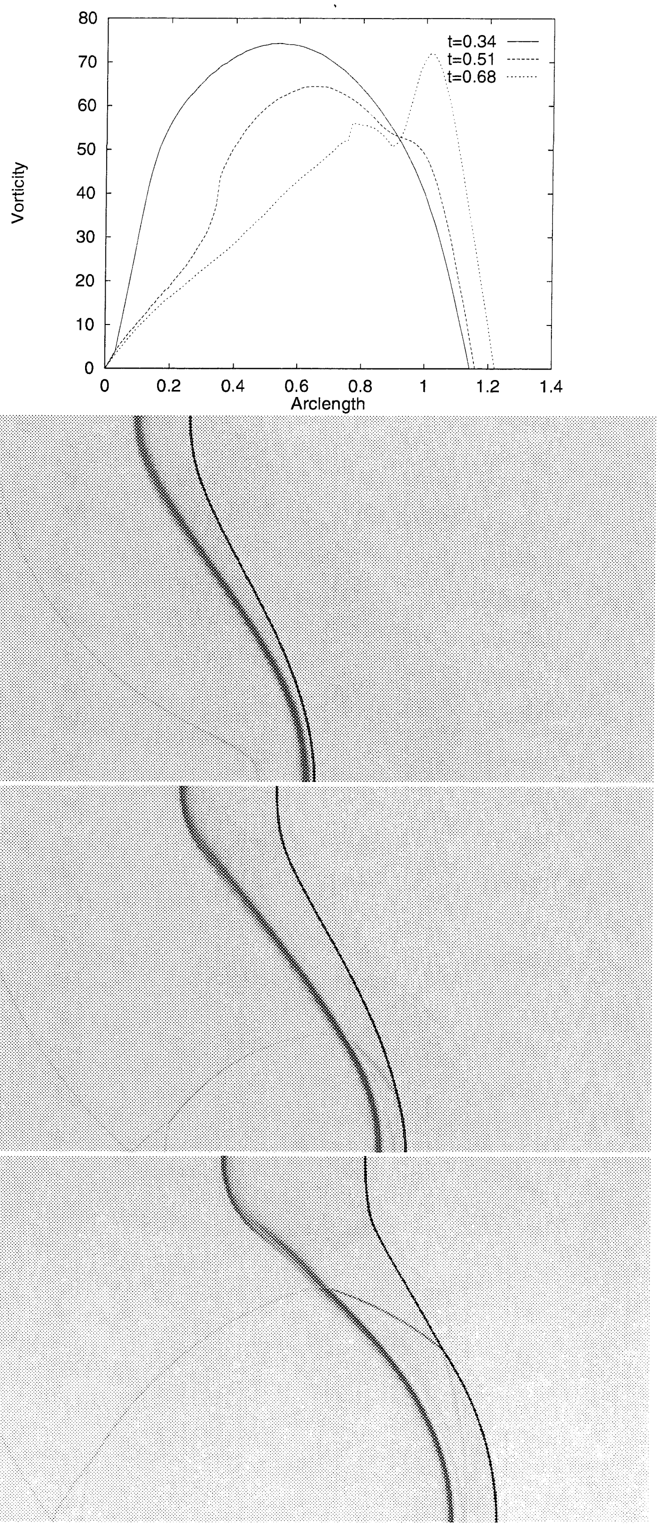


Figure 3.22: The compressible phase of the simulation of figure 3.20(b). Schlieren images are taken at times $t = 0.34$, $t = 0.51$ and $t = 0.68$ corresponding to profiles shown in the graph. The growth rate is plotted in figure 3.20(b).

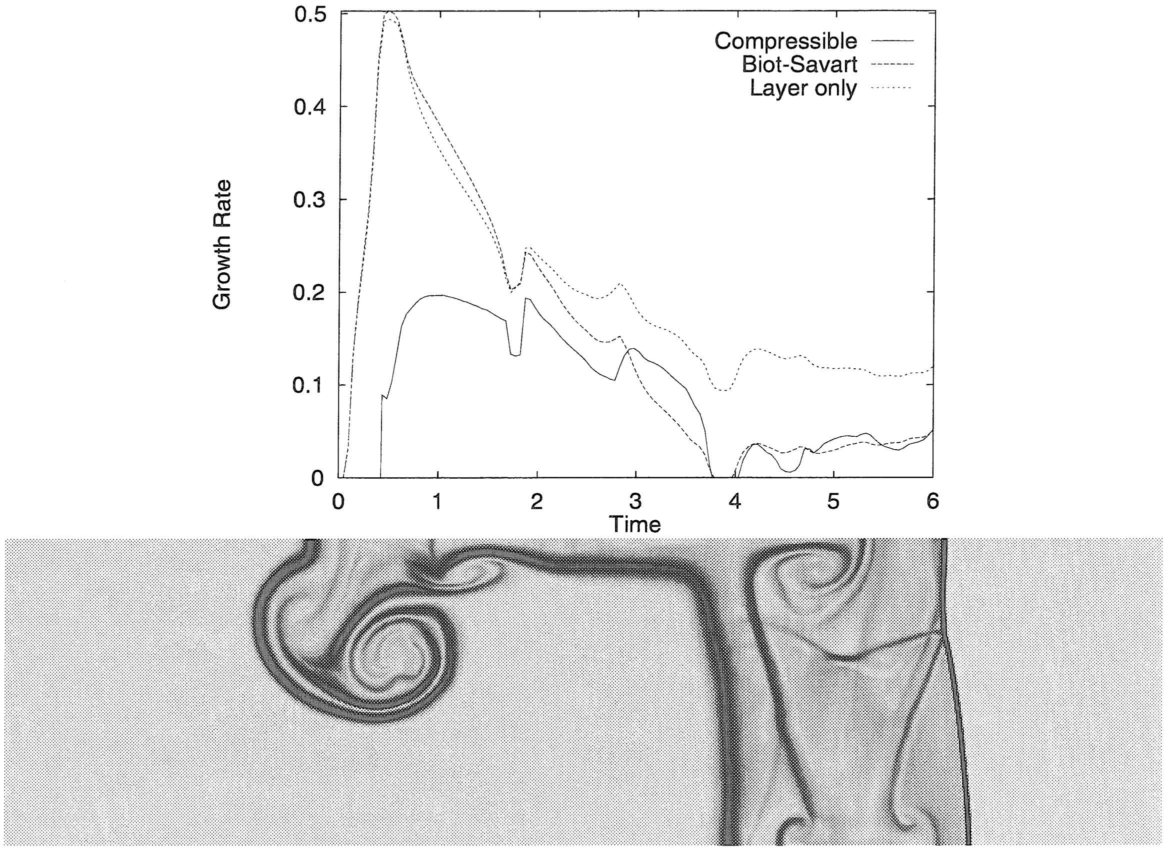


Figure 3.23: Growth rate induced by $M_I = 8$ shock striking interface with $a_0 = 1$, $At = 0.8$, and $L = 0.1$. Schlieren image taken at $t = 4.8$.

Chapter 4

Geometrical Shock Dynamics

Geometrical Shock Dynamics is the name given to a method devised by Whitham in the late 1950s [64, 65] to solve shock propagation problems in more than one space dimension. Although the method is based on assumptions which are difficult to justify, the method has enjoyed remarkable success on a wide variety of shock propagation problems when compared with experiments and solutions of the Euler equations. The original derivation developed the needed equations for propagation into uniform media. The extension to propagation into media with varying sound speed, pressure, and ratio of specific heats was performed by Collins and Chen [13]. Their expression contained errors in some coefficients, however, and was later corrected by Catherasoo and Sturtevant [9]. Formulations of the method to deal with propagation into uniform flow ahead of the shock was done by Whitham [66] and for the case of nonuniform flow by Han and Yin [18] and Cates [8]. Best [3, 4] has shown how the equations of Geometrical Shock Dynamics can be derived as a formal approximation to the Euler equations and has developed a technique for deriving higher order corrections. The method has been implemented numerically using Lagrangian methods [3, 19, 55], method of characteristics [9, 71] and methods designed for the supersonic potential equation [8, 56].

In chapter 3 the impulsive approximation – which does not compute the shock refraction phase or calculate the vorticity distribution outside the density layer – was shown to give incorrect predictions for the growth rate of the interface and circulation in the layer. An alternative approach is to model the instability by explicitly calculating the passage of the shock through the layer and the subsequent propagation of the transmitted shock. The deposition of vorticity both in the layer and in the bulk of the fluid can be computed by calculating the vorticity generation at the shock front as it propagates. It is this idea that is developed further in this chapter using Geometrical Shock Dynamics to compute the motion of the shock. Some work of this nature has been done by Samtaney and Zabusky who analytically computed the vorticity initially deposited on the interface [51] and used this in incompressible vortex models to calculate the growth rate [52]. Their approach, however, is

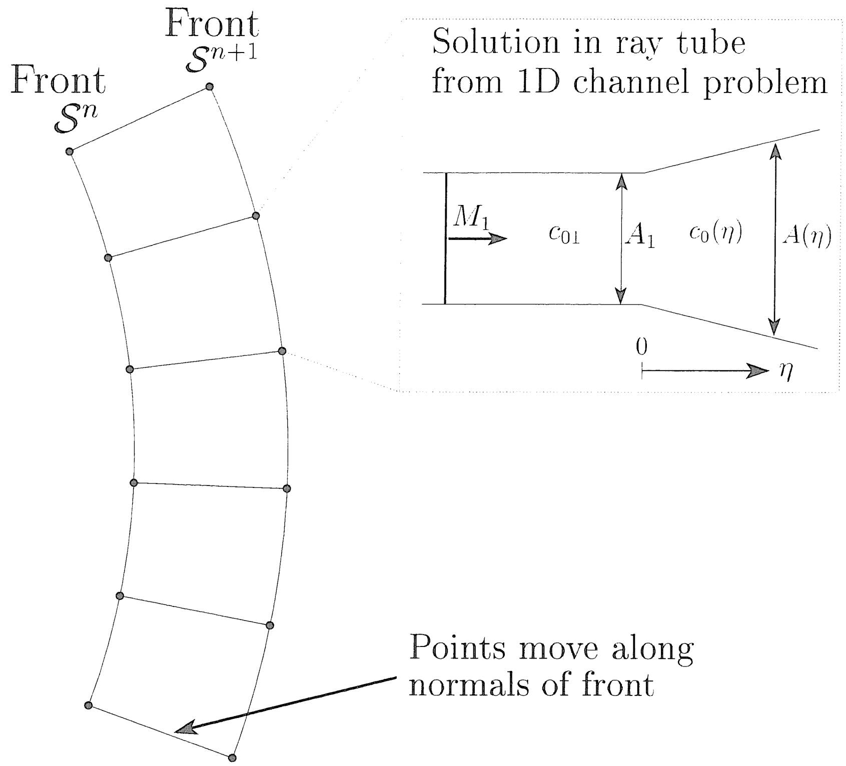


Figure 4.1: The method of Geometrical Shock Dynamics

only applicable to interfaces of zero thickness. Further, only the initial vorticity distribution on the interface is computed while the additional vorticity laid down by the oscillating transmitted and reflected waves, as well as modifications of the interfacial vorticity by secondary interactions between the fronts and the interface, is not considered. We begin by presenting the derivation of Geometrical Shock Dynamics for propagation into media of varying sound speed.

4.1 Equations of Geometrical Shock Dynamics

The method of Geometrical Shock Dynamics is schematically depicted in figure 4.1. The propagation of the shock from one instant to the next is computed by requiring points on the shock to move along normals to the front. The strength of the shock in each ray tube is updated using the solution to the problem of a shock moving into a channel of slowly varying area, A, and sound speed, c_0, as shown in the insert. We briefly outline the solution for the one-dimensional channel problem below. The full derivation is presented in [67].

The coordinate along the channel is η, with the datum $\eta = 0$ located at the start of the nonuniform portion. The one-dimensional Euler equations are averaged over the height of the channel to obtain:

$$\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial \eta} + \rho\frac{\partial u}{\partial \eta} + \rho u\frac{A'(\eta)}{A(\eta)} = 0, \tag{4.1a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial \eta} + \frac{1}{\rho}\frac{\partial p}{\partial \eta} = 0, \tag{4.1b}$$

$$\frac{\partial p}{\partial t} + u\frac{\partial p}{\partial \eta} - c^2\left(\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial \eta}\right) = 0. \tag{4.1c}$$

Before reaching $\eta = 0$ the shock propagates down a section of the channel with constant area A_1 and ambient sound speed c_{01} with a Mach number of M_1. As the shock propagates through the uniform region, the state behind the shock is labeled state 2 and that directly in front is labeled state 1. The solution in state 2 can be calculated from state 1 and M_1 using the Rankine-Hugoniot conditions (hereafter called "shock jump conditions"). At $t = 0$ the shock enters the nonuniform part of the channel where the area is $A(\eta)$ and ambient sound speed is $c_0(\eta)$. The Mach number of the shock as it propagates through this region is M and the state behind is given by ρ, u, and p. The sound speed and channel area are assumed to both vary slowly so linearization of the problem about state 2 can be done. Requiring $A'(\eta)/A \ll 1$ and $c_0'(\eta)/c_0 \ll 1$, the equations become:

$$\frac{\partial \rho}{\partial t} + u_2\frac{\partial \rho}{\partial \eta} + \rho_2\frac{\partial u}{\partial \eta} + \rho_2 u_2\frac{A'(\eta)}{A(\eta)} = 0, \tag{4.2a}$$

$$\frac{\partial u}{\partial t} + u_2\frac{\partial u}{\partial \eta} + \frac{1}{\rho_2}\frac{\partial p}{\partial \eta} = 0, \tag{4.2b}$$

$$\frac{\partial p}{\partial t} + u_2\frac{\partial p}{\partial \eta} - c_2^2\left(\frac{\partial \rho}{\partial t} + u_2\frac{\partial \rho}{\partial \eta}\right) = 0. \tag{4.2c}$$

These equations form a linear hyperbolic system. The equations can be placed into characteristic form and integrated to give

$$(p - p_2) + \rho_2 c_2(u - u_2) = -\frac{\rho_2 c_2^2 u_2}{u_2 + c_2}\frac{A(\eta) - A_1}{A_1} + F(\eta - (u_2 + c_2)t), \tag{4.3a}$$

$$(p - p_2) - \rho_2 c_2(u - u_2) = -\frac{\rho_2 c_2^2 u_2}{u_2 - c_2}\frac{A(\eta) - A_1}{A_1} + G(\eta - (u_2 - c_2)t), \tag{4.3b}$$

$$(p - p_2) - c_2^2(\rho - \rho_2) = H(\eta - u_2 t). \tag{4.3c}$$

The lines $d\eta/dt = u_2 + c_2$ and $d\eta/dt = u_2 - c_2$ are the $\mathcal{C}^+$ and $\mathcal{C}^-$ characteristics, respectively. The functions F, G and H are the constants from integration along characteristic lines. The function F in the equation for the $\mathcal{C}^+$ characteristic can be determined by evaluating equation (4.3a) at any place and time, η and t. For any $\eta < 0$, however, $u = u_2$, $p = p_2$, and $A = A_1$ so $F = 0$. Every $\mathcal{C}^+$ characteristic line originates from the uniform section $\eta < 0$ in $\eta - t$ space so $F(\eta - (u_2 + c_2)t) \equiv 0$.

Both p and u at the shock can be related to the Mach number and upstream quantities. Using Taylor series:

$$p = p_2 + \left.\frac{\partial p(M,\gamma,p_0)}{\partial M}\right|_1 (M - M_1)\,, \tag{4.4a}$$

$$u = u_2 + \left.\frac{\partial u(M,\gamma,c_0)}{\partial M}\right|_1 (M - M_1) + \left.\frac{\partial u(M,\gamma,c_0)}{\partial c_0}\right|_1 (c_0 - c_{01})\,, \tag{4.4b}$$

where $p(M,\gamma,p_0)$ and $u(M,\gamma,c_0)$ are obtained from the shock jump conditions. The expressions for $p - p_2$ and $u - u_2$ are substituted into equation (4.3a) to get an expression for the shock Mach number in the nonuniform region, M:

$$M - M_1 = -\frac{M_1^2 - 1}{M_1 \lambda_1}\left[\frac{1}{A_1}(A - A_1) + \frac{g_1}{c_{01}}(c_0 - c_{01})\right]\,, \tag{4.5a}$$

where

$$\lambda_1 = \left(1 + \frac{2}{\gamma + 1}\frac{1 - \mu_1^2}{\mu_1}\right)\left(1 + 2\mu_1 + \frac{1}{M_1^2}\right)\,, \tag{4.5b}$$

$$g_1 = 1 + \frac{2\mu_1(M_1^2 - 1)}{(\gamma - 1)M_1^2 + 2}, \tag{4.5c}$$

$$\mu_1^2 = \frac{(\gamma - 1)M_1^2 + 2}{2\gamma M_1^2 - (\gamma - 1)}. \tag{4.5d}$$

This completes the solution to the one-dimensional channel problem.

The method of Geometrical Shock Dynamics requires that points on the shock move along normals, effectively breaking the shock front into a series of stream-tubes down which a locally planar shock propagates. Note that although the shock jump conditions require that flow immediately behind the shock be normal, particle paths are not the same as rays. As the distance from the shock increases, the deviation of particle motion from that normal becomes larger. Thus, there is an approximation made here [67]. The assumption is made that the error involved in propagating points along normals is small. The channel problem, however, is not immediately applicable to shock propagation down each stream-tube. In the solution of the channel problem, an equation for the Mach number was obtained only because the constant of integration in equation (4.3a), F, was identically zero. In the two-dimensional problem, there is no uniform region for each shock segment to emerge from so the solution to the channel problem cannot be used to update the solution in each stream-tube.

It is at this point that the second fundamental simplification in the method of Geometrical Shock Dynamics is made: the solution to the channel problem will be used as an approximation. By pretending that each piece of the shock front has emerged from a uniform region, the influence of the flow behind on the shock is completely ignored. The error that is made in doing this is not clear. However, it does allow us to obtain an equation for the Mach number in each tube. The evolution equation for the local Mach number at each point on the shock is the nonlinear analogue of equations (4.5a)-(4.5d):

$$\frac{DM(\beta,t)}{Dt} = -\frac{M^2-1}{M\lambda(M)}\left[\frac{1}{A}\frac{DA}{Dt} + \frac{g(M)}{c_0}\frac{Dc_0}{Dt}\right], \tag{4.6a}$$

$$\lambda(M) = \left(1+\frac{2}{\gamma+1}\frac{1-\mu^2}{\mu}\right)\left(1+2\mu+\frac{1}{M^2}\right), \tag{4.6b}$$

$$g(M) = 1+\frac{2\mu(M^2-1)}{(\gamma-1)M^2+2}, \tag{4.6c}$$

$$\mu^2 = \frac{(\gamma-1)M^2+2}{2\gamma M^2-(\gamma-1)}, \tag{4.6d}$$

where D/Dt is a derivative following a particle on the shock and β is a Lagrangian coordinate along the shock that identifies specific particles. Whitham's original derivation [64] did not consider variable sound speed and he integrated equation (4.6a) to obtain an expression directly relating A and M – the so-called "A-M relation." Equation (4.6a) tells us how the Mach number at each point along the shock varies due to changes in ambient sound speed and local curvature of the shock front. Allowing for variation in downstream pressure and ratio of specific heats results in additional forcing terms in equation (4.6a) as shown by Collins and Chen [13]. Their coefficients of the Dp/Dt and $D\gamma/Dt$ terms contained errors which were later corrected by Catherasoo and Sturtevant [9]. Along with the Mach number relation, which specifies the dynamics, there is a kinematic equation which states each point moves along the normal to the shock:

$$\frac{D\mathbf{x}}{Dt} = c_0(\beta, t)M(\beta, t)\hat{\mathbf{n}}(\beta, t), \tag{4.7}$$

where $\hat{\mathbf{n}}(\beta, t)$ is the unit normal to the front.

Whitham noted that the $A - M$ relation could be derived by using the shock jump conditions, which only hold right at the shock, in the differential equation for the $\mathcal{C}^+$ characteristic of the Euler equations. Thus, equation (4.6a) could have been derived all in one simple step. Whitham refers to this alternative derivation as the "characteristic rule." By comparing the exact $\mathcal{C}^+$ equation with the characteristic rule, Whitham determined that success of Geometrical Shock Dynamics relies on the following quantity being small when evaluated at the shock:

$$\left(\frac{1}{c_0 M} - \frac{1}{u + c}\right)\left(\frac{\partial p}{\partial t} + \rho c \frac{\partial u}{\partial t}\right). \tag{4.8}$$

The first factor is simply a measure of the velocity difference between the shock and disturbances propagating along the $\mathcal{C}^+$ characteristic. Although the characteristic and shock overlap in the weak shock limit, this factor is not small for even moderate Mach numbers, quickly approaching its maximum value of 0.274 (for $\gamma = 1.4$). The accuracy of the method, however, is oftentimes much better than this, and it is because of the smallness of

$$\frac{1}{\partial p/\partial t}\left(\frac{\partial p}{\partial t} + \rho c \frac{\partial u}{\partial t}\right), \tag{4.9}$$

that the method works so well. This term is some measure of the nonuniformity behind the shock. Although expression (4.9) is zero in the channel problem that forms the basis of Geometrical Shock Dynamics, the reason why this quantity should be small for general two-dimensional shock propagation problems is not clear at all. Indeed, Whitham himself states "... no really satisfactory explanation of this was found." [67] Thus, it must always be remembered that this method is based on some approximations whose accuracy cannot be assessed in advance. Comparison of solutions of Geometrical Shock Dynamics with experiment and other theory has shown, however, that Geometrical Shock Dynamics produces accurate solutions to a wide range of shock propagation problems. For some problems involving strong shocks propagating into uniform media Geometrical Shock Dynamics has produced amazingly accurate solutions, prompting some to theorize the effects of the flow behind the shock are not strong enough to significantly modify the shock in those problems.

By neglecting the flow behind the shock, a problem in N-dimensional space is reduced to one in $(N-1)$ dimensions, allowing the method to compute solutions cheaply when implemented numerically. The simple form of (4.6a) can provide quantitative justification to theoretical arguments as well. For example, it is well known that perturbations of a planar shock decay as the shock propagates [42]. Looking at (4.6a) (and ignoring the sound speed forcing term), troughs of the shock are regions where $DA/D\eta$ are negative, leading to an increase in Mach number. Expansive regions of the shock slow down because $DM/Dt < 0$ when $DA/Dt > 0$. With each oscillation, the bulge will spread out along the shock, resulting in a reduction of the amplitude of perturbation [67]. Geometrical Shock Dynamics is only an approximate theory, however, and does not damp sufficiently small perturbations, allowing them to oscillate with fixed amplitude indefinitely instead [62]. An analytic solution showing this behavior is derived in Appendix C.

An alternative form of the equations of Geometrical Shock Dynamics can be obtained on an orthogonal system of coordinates (α, β) where $\alpha(\mathbf{x})$ =constant are shock fronts and β=constant are the rays. Letting M and θ, the angle of ray inclination from horizontal, be the unknowns, the equations of Geometrical Shock Dynamics can be written in the following characteristic form [9]:

$$d\theta \pm d\omega = -\left(\frac{M}{A}\frac{\partial c_0}{\partial \beta} \pm C\frac{Ag(M)}{c_0^2 M}\frac{\partial c_0}{\partial \alpha}\right) d\alpha, \tag{4.10a}$$

$$C = \frac{c_0}{A}\sqrt{\frac{M^2-1}{\lambda(M)}}, \tag{4.10b}$$

$$\omega = \int_1^M \sqrt{\frac{\lambda(m)}{m^2-1}}dm. \tag{4.10c}$$

These hyperbolic equations describe the propagation of waves along the front carrying changes in M and θ. The waves are nonlinear and disturbances carrying an increase in Mach number will steepen and eventually form discontinuities in M and θ with respect to β. Whitham calls these *shock-shocks* [67] and has derived jump conditions for them. These shock-shocks, in fact, signify the formation of a Mach stem along the shock. Similarly, disturbances carrying a decrease in Mach number can form expansion fans (named *shock-expansions* by Catherasoo and Sturtevant [9]). There are no walls in the Richtmyer-Meshkov problem (it is periodic on $y \in [0, 2\pi]$), so these waves are generated solely through gas nonuniformities.

4.2 Numerical Implementation

For most problems of interest a numerical implementation is necessary to integrate the equations. Some researchers have used the method of characteristics [9, 71]. It is also possible to write the equations of Geometrical Shock Dynamics in the same form as the supersonic potential equation, allowing numerical methods designed for that equation to be used for Geometrical Shock Dynamics as well [8, 56]. We used a Lagrangian approach [3, 19, 55] to propagate points along the shock front and numerically integrate the differential equations (4.6a) and (4.7) using the second order Runge-Kutta method commonly known as Heun's method (also known as the modified trapezoidal rule). Most earlier researchers have used an integral form of the A-M relation and numerically inverted it to obtain the Mach number. Periodic boundary conditions along the shock were used and one full wavelength of perturbation was computed. The forcing term Dc_0/Dt in (4.6a) is evaluated using the expression for the density field in front of the shock, equation (1.3).

Lagrangian methods must deal with two issues. One is the clustering of points in compressive regions of the front and dispersion in expansive portions, leading to a nonuniform distribution of points along the front. Loss of resolution in shock-expansions and collision of two points at a shock-shock, forcing the ray tube area between them to zero and causing a singularity in equation (4.6a), are two detrimental effects of the nonuniform spacing. The second issue is how to deal with the formation of shock-shocks. Calculation of geometrical quantities, such as front normals, require differentiation along the shock and care needs to be taken in the event of a discontinuous Mach number distribution along the front. We implemented two different approaches to deal with these numerical issues and describe them below. A comparison of the results obtained with both implementations was usually done in order to ensure our results were not dependent on the numerical details of the solution.

4.2.1 Cubic Spline Approach

The first approach is to fit the points along the shock front to a cubic spline. Computing derivatives along the front is trivial. The arclength along the front at time t is denoted as

$s(t)$ with $s(t) \equiv 0$ at the point located at $y = 0$. The normals to the shock are determined from

$$\hat{\mathbf{n}}(s(t),t) = \left(\frac{\partial y(s(t),t)}{\partial s(t)}, -\frac{\partial x(s(t),t)}{\partial s(t)} \right). \tag{4.11}$$

Equation (4.6a) requires the calculation of $(1/A)DA/Dt$ which must also be computed from the local geometry. Best [3] gives a suitable expression:

$$\frac{1}{A}\frac{DA}{Dt} = c_0(s(t),t)M(s(t),t)\left[\frac{\partial x(s(t),t)}{\partial s(t)}\frac{\partial^2 y(s(t),t)}{\partial s(t)^2} - \frac{\partial y(s(t),t)}{\partial s(t)}\frac{\partial^2 x(s(t),t)}{\partial s(t)^2} \right]. \tag{4.12}$$

None of the expressions require the derivative $\partial M(s(t),t)/\partial s(t)$ so sharp gradients in M with respect to $s(t)$ do not pose a problem. Derivatives along the front are calculated by differentiating the cubic spline interpolating the points.

To maintain a uniform distribution of points along the front the refinement scheme of Henshaw *et al.* [19] was used. The point spacing was checked after every time step and it was required that

$$d \leq \frac{\Delta s_i(t)}{\Delta s_{\text{avg}}} \leq D \qquad \text{for all } i = 2, \ldots, N \tag{4.13}$$

where N is the number of points along the shock, $\Delta s_i(t) = s_i(t) - s_{i-1}(t)$, and $\Delta s_{\text{avg}} = s_N(t)/(N-1)$. The values $d = 1/2$ and $D = 2$ were found to work well. If two points were too close together, one was deleted. To maintain the same number of points on the front, a search was performed along the front to locate the largest gap between two existing points and a new point was inserted halfway between the two existing points using cubic spline interpolation. Similarly, if two points are too far apart, a point was inserted between them and a corresponding point is deleted from the location of minimum $\Delta s_i(t)$. In this way, instabilities arising from violations of a CFL criterion were avoided and regions of shock-expansions were accurately resolved.

It was also necessary to employ the smoothing procedure of Henshaw *et al.* to reduce high frequency errors in $\mathbf{x}$. Every set number of time steps, n_s, values of $\mathbf{x}$ were reassigned according to the following:

$$\frac{1}{2}(\mathbf{x}_{i-1} + \mathbf{x}_{i+1}) \to \mathbf{x}_i. \tag{4.14}$$

The interval, n_s, was chosen on a per-problem basis and was typically in the range of 10-100 time steps.

4.2.2 Fourier Collocation Approach

The Richtmyer-Meshkov problem is periodic on $y \in [0, 2\pi]$, allowing use of Fourier Collocation to obtain spectral accuracy in space. Derivatives along the shock are done using the Fast Fourier Transform (FFT). FFT routines require function values to be given at equally spaced intervals in the independent variable, so the front was parameterized with respect to the Lagrangian variable β rather than arclength. The value of β at each point on the shock is fixed throughout the simulation and initialized as $\beta_i = y_i$ for each point i along the shock at $t = 0$. Note that $y(\beta, t)$ is not periodic so all $\partial y(\beta, t)/\partial \beta$ terms were computed by performing the Fourier transforms on $y(\beta, t) - \beta$ rather than $y(\beta, t)$. Fourier methods suffer from the Gibbs phenomenon if the function is not sufficiently smooth so sharp gradients in Mach number and position along the shock must be resolved over several points. To ensure discontinuities do not form, a linear diffusion term with constant viscosity was added to each equation:

$$\frac{Dx}{Dt} = N_x + \nu \frac{\partial^2 x}{\partial \beta^2}, \tag{4.15a}$$

$$\frac{Dy}{Dt} = N_y + \nu \frac{\partial^2 y}{\partial \beta^2}, \tag{4.15b}$$

$$\frac{DM}{Dt} = N_M + \nu \frac{\partial^2 M}{\partial \beta^2}, \tag{4.15c}$$

where N_x, N_y and N_M are the nonlinear right-hand sides of equations (4.6a) and (4.7). The viscous terms can be treated with exact integration via an integrating factor in Fourier space and, so, have no effect on the formal order of accuracy* or stability of the scheme [7]. Care was taken to use a small enough value of ν in each simulation that the solution did

---

*Here accuracy refers to the error incurred in the numerical solution of equations (4.15a)-(4.15c). The addition of viscous terms will have some effect on the difference between the numerical solution and the true solution to the nonviscous equations of Geometrical Shock Dynamics, (4.6a) and (4.7). For example, a coefficient of viscosity, ν, that is too large can smear shock-shocks out too much and erode extrema of $M(\beta)$.

not depend on its value. Geometrical quantities are obtained from the following:

$$A(\beta,t) = \sqrt{\left(\frac{\partial x}{\partial \beta}\right)^2 + \left(\frac{\partial y}{\partial \beta}\right)^2}, \tag{4.16}$$

$$\frac{DA}{Dt} = \frac{1}{A}\left[\frac{\partial x}{\partial \beta}\frac{\partial}{\partial \beta}\left(\frac{Dx}{Dt}\right) + \frac{\partial y}{\partial \beta}\frac{\partial}{\partial \beta}\left(\frac{Dy}{Dt}\right)\right], \tag{4.17}$$

$$\hat{\mathbf{n}}(\beta,t) = \frac{1}{\|\partial \mathbf{x}/\partial \beta\|}\left(\frac{\partial y(\beta,t)}{\partial \beta}, -\frac{\partial x(\beta,t)}{\partial \beta}\right). \tag{4.18}$$

Both the cubic spline method and Fourier Collocation method have their advantages and disadvantages. The Fourier method is restricted to problems with periodic boundary conditions while the spline method is not. The Fourier method, however, features spectral accuracy along the front. The cubic spline method is much better at maintaining adequate resolution in expansive regions of the front because of its adaptive point refinement scheme. The spline approach, however, requires the user to specify a maximum and minimum point spacing, D and d, as well as a smoothing interval, n_s. The Fourier method only requires a coefficient of viscosity, ν. In the majority of simulations presented in the next section, we have used the Fourier Collocation implementation because of its superior accuracy in the direction along the front. Checks were made against shock fronts computed using the cubic spline method to ensure our results were not dictated by the particular implementation used.

4.3 Results

4.3.1 Qualitative and Quantitative Comparison with Euler Simulations

The use of Geometrical Shock Dynamics to compute shock propagation into media of varying sound speed was popularized by Catherasoo and Sturtevant [9] and Schwendeman [55]. Using a Lagrangian implementation, Schwendeman computed the passage of a Mach 1.22 shock through a cylindrical inhomogeneity to compare with experiments done by Haas and Sturtevant [17]. Later, Cates [8], using a numerical method designed for the supersonic potential equation, re-did those comparisons in more detail. Both found good agreement in the location of shock-shocks in Geometrical Shock Dynamics and corners of the shock in experiments. In figure 4.2 the same problem is considered but Geometrical Shock Dynamics results are compared against Euler simulations instead.* Only the top half of the computational domain is shown in the figure since the bottom half is simply a mirror image. Following Schwendeman, a sound speed distribution of:

$$c_0(r) = \begin{cases} 1 & \text{if } r - R \geq R_I, \\ \frac{1}{2}\left[(1 + c_{02}) + (1 - c_{02})\sin\left(\frac{\pi(r-R)}{2R_I}\right)\right] & \text{if } |r - R| < R_I, \\ c_{02} & \text{if } r - R \leq -R_I, \end{cases} \tag{4.19}$$

was used where r is the distance from the center of the cylinder, $R = 1$ and $R_I = 0.01$. The sound speeds of $c_{02} = 0.53$ and $c_{02} = 2.32$ were chosen to compare with experiments of a shock propagating through a cylinder immersed in air containing freon and helium, respectively. The difference in the ratio of specific heats, γ, of both gases were ignored by Schwendeman and Cates. Both the Euler simulation and Geometrical Shock Dynamics calculations shown in figure 4.2 were done for a uniform ratio of specific heats $\gamma = 1.4$. The $c_{02} = 0.53$ case had an Atwood ratio of $At = 0.56$ across the cylinder interface and the $c_{02} = 2.32$ case, a ratio of $At = -0.69$. The shock front slows down as it propagates through the cylinder in the positive Atwood ratio case and speeds up in the negative Atwood ratio case. The variation in sound speed along the front generates waves on the shock which carry changes in local Mach number and angle of inclination according to equations (4.10a)-(4.10c). These waves propagate along the front and cause the shape of the shock to change during the course of the computation. This can readily be seen in figure 4.3 where successive shock fronts from the Geometrical Shock Dynamics computations shown in figure 4.2 are plotted. The mid-line of the interface, $r = R$, is plotted as a dashed line for reference. The creation of both shock-shocks and shock-expansion regions along the fronts are observed in these plots.

The agreement between shock fronts from the Euler simulations and Geometrical Shock Dynamics computations appears to be quite good for both cases shown in figure 4.2. Close examination of figure 4.2(a) shows that the shock position inside the cylinder computed with Geometrical Shock Dynamics is slightly ahead of that computed from the Euler simulation.

---

[*] In creating the images in figure 4.2 a smaller value of the "exposure" parameter has been used in the numerical Schlieren image routine (described in appendix B), resulting in a darker image than the other Schlieren pictures in this thesis. This was done to create greater contrast between the white shock front and the surrounding flow. This is purely for flow visualization purposes and the darkness of the image should not be interpreted as indicating anything special about the flow.

Yet the location of the corner along the front has been predicted extremely well. For the $c_{02} = 2.32$ case, the four shock intersection seen in the image from the Euler simulation has been replaced by two closely spaced shock-shocks in the Geometrical Shock Dynamics front. These two shock-shocks of opposite families are needed by Geometrical Shock Dynamics to approximate the regular reflection of the true shock which turns the shock without a large increase in Mach number [8].

In these simulations the cubic spline method was used to compute the Geometrical Shock Dynamics shock front rather than the Fourier Collocation method. The shock fronts computed from the Fourier method differed from the ones shown in figure 4.2 in expansive regions due to dispersion of points, increasing the numerical viscosity there. The cubic spline method did not experience this problem because of the point refinement scheme employed. The simulations shown in figure 4.2 are two of the very few occasions when any significant difference between the cubic spline and Fourier methods was observed.

It is encouraging that Geometrical Shock Dynamics works so well for a problem featuring such a highly curved, thin interface. However, the Mach number was $M_I = 1.22$ and we wish to use shocks of arbitrary strengths. As a second test Geometrical Shock Dynamics was applied to the one-dimensional problem of propagation through a thin density layer. The density profile of the layer is given by equation (1.3) with $a_0 = 0$ and $L = 0.01$. A quantitative comparison between the Geometrical Shock Dynamics and Euler simulations was done by comparing pressure profiles. The pressure at the Geometrical Shock Dynamics shock was calculated from the Mach number and the shock jump conditions. The theory of Geometrical Shock Dynamics implicitly assumes uniform flow behind the shock so the pressure profile consists of two constant states, pre-shock and post-shock, separated by a discontinuity at the shock position. The results are seen in figures 4.4 for low Atwood ratios and 4.5 for high Atwood ratios. For low Atwood ratios the agreement in shock location and value of post-shock pressure between Geometrical Shock Dynamics and Euler simulation is excellent. In the higher Atwood ratio problems, however, Geometrical Shock Dynamics has over-predicted the pressure. The over-prediction in pressure is a result of over-prediction in Mach number of the shock. This causes the velocity of the shock, $c_0 M$, in the Geometrical Shock Dynamics calculation to be larger than the velocity of the shock in the Euler simulation, resulting in incorrect shock locations in figure 4.5. Four cases are shown in the figure featuring strong shocks ($M_I = 8$) propagating through thin layers ($L = 0.01$)

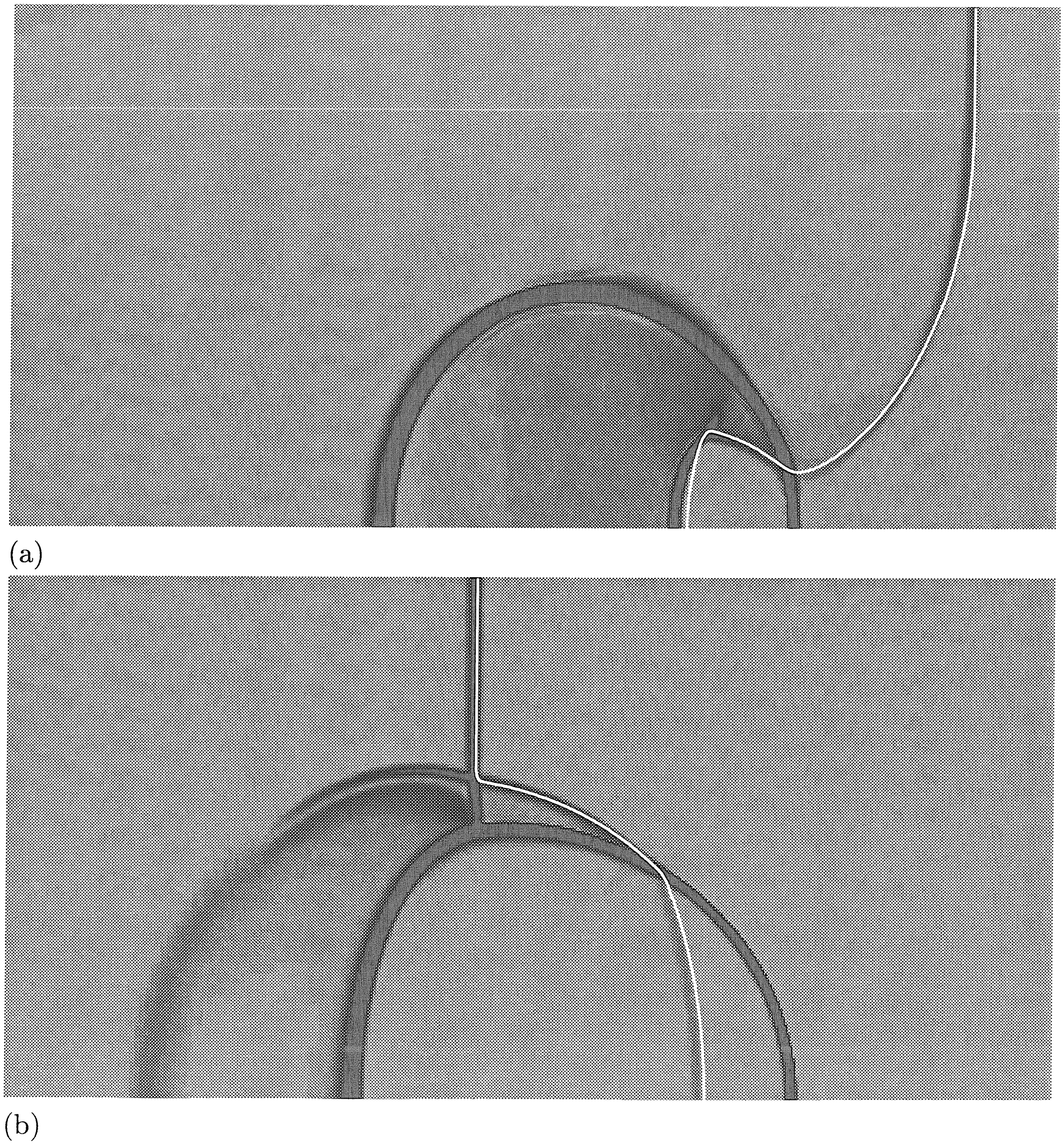


Figure 4.2: Propagation through cylindrical inhomogeneity given by equation (4.19). The shock front computed by Geometrical Shock Dynamics is shown as a solid white line superimposed on a numerical Schlieren image from an Euler simulation. (a) $c_{02} = 0.53$, (b) $c_{02} = 2.32$.

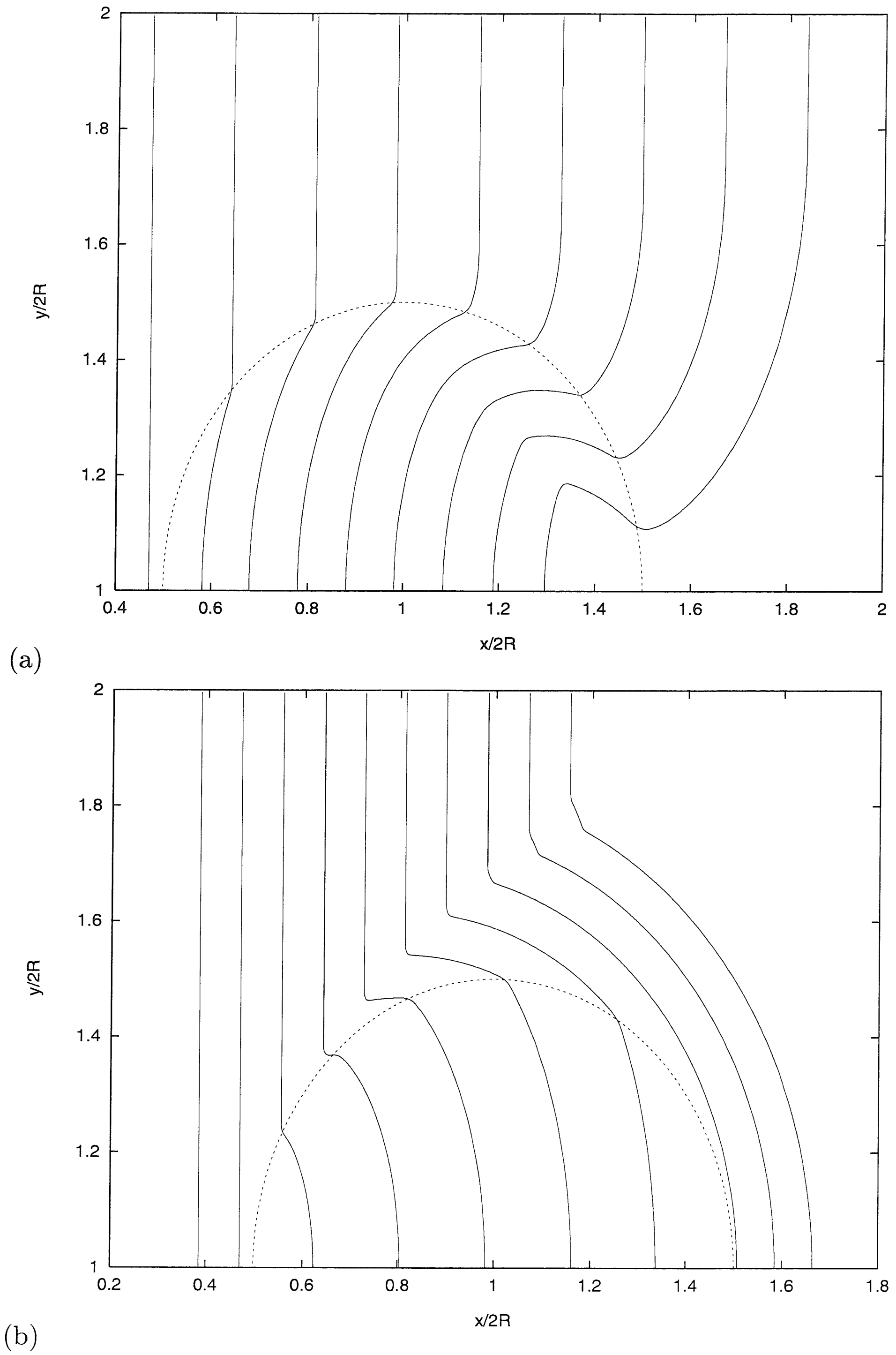


Figure 4.3: Successive shock fronts from the same Geometrical Shock Dynamics computations shown in figure 4.2. The mid-line of the interface, $r = R$, is plotted as a dashed line for reference.

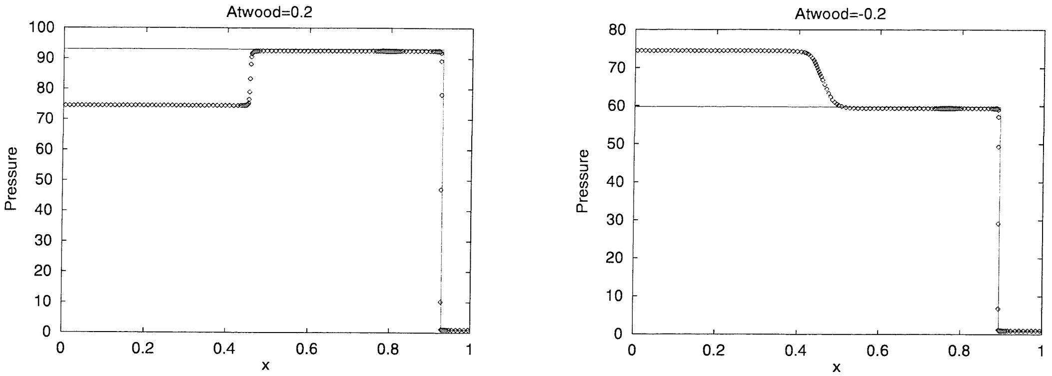


Figure 4.4: Comparison of pressure profiles from Geometrical Shock Dynamics (solid line) and Euler code (diamond symbols) for a one-dimensional shock-layer problem with $At = \pm 0.2$, $M_I = 8$ and $L = 0.01$.

and shocks of more moderate strength ($M_I = 2$) through thick ($L = 1$) layers. Geometrical Shock Dynamics over-predicts the Mach number for all cases, indicating the inaccuracy is not due to the assumption $c_0'(\eta)/c_0(\eta) \ll 1$ made in the derivation of Geometrical Shock Dynamics.

Chisnell [10] studied the shock-layer problem analytically and obtained the same relation between variation in Mach number and ambient sound speed that Whitham did (equation (4.6a) without the DA/Dt term) using an alternative derivation. For high Atwood ratios the expression did, indeed, over-predict the strength of the transmitted shock but, by considering the effect of the flow behind the shock, the estimate could be improved substantially. Chisnell found it was enough to only consider the effect of "doubly-reflected waves" – waves that were originally reflected backwards by the shock as it passed through the layer and then reflected back towards the shock upon interaction with the nonuniform density distribution in the wake of the shock. Cases with negative Atwood ratios were not considered by Chisnell. Because the neglect of nonuniform flow behind the shock results in inaccurate calculation of the strength of the transmitted shock in the one-dimensional problem for high Atwood ratios, only lower ratios, $At = \pm 0.2$, will be considered from here on.

We start by considering a Mach 1.25 shock moving through the Richtmyer-Meshkov density field given by (1.3). In figure 4.6 the results for a layer of thickness $L = 0.01$, an Atwood ratio of $At = 0.2$ and an initial amplitude of $a_0 = 2$ are shown. The simulation has

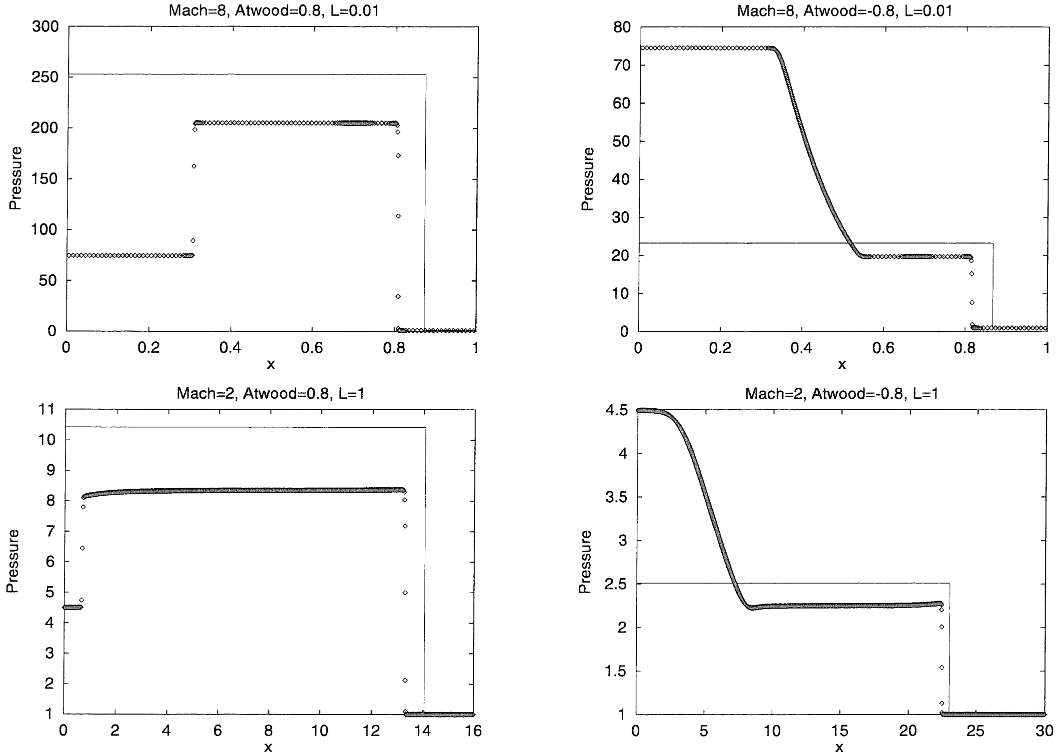


Figure 4.5: Comparison of pressure profiles from Geometrical Shock Dynamics (solid line) and Euler code (diamond symbols) for a one-dimensional shock-layer problem with $At = \pm 0.8$. (top row: $M_I = 8$ and $L = 0.01$, bottom: $M_I = 2$ and $L = 1$)

been run long enough for the shock to travel far beyond the layer. Only the flow field for half a wavelength, $0 \leq y \leq \pi$, is shown. At early times, the shape and position of the two shock fronts agree very well. However, at $t = 4$ there is clearly a discrepancy. The portion of the Geometrical Shock Dynamics shock front along the top of the image ($y = \pi$) leads the shock in the Euler simulation. The Geometrical Shock Dynamics shock front also has a feature not seen in the Euler shock: a small kink located near $y = \pi/2$. The shape of the two shocks are quite different, however, the mean shock location and velocity are in good agreement.

Previous researchers have not had the experimental data needed to assess the accuracy of $M(\beta, t)$ computed by Geometrical Shock Dynamics. Comparisons have been based on features of the shock front such as location of discontinuities (see figure 4.2) and angles of refraction at interfaces. Because Geometrical Shock Dynamics is compared with Euler simulations, pressure profiles can be compared. In figure 4.7 the pressure as a function of x along two fixed values of y is plotted. The data is taken from the simulation shown in figure 4.6 at $t = 2$ when the shock has just completely traversed the layer. The theory of Geometrical Shock Dynamics assumes uniform flow behind the shock so its downstream pressure profile is simply the value of the pressure at the shock front. The pressure at the shock from both simulations is compared and it is seen that the pressure predicted by Geometrical Shock Dynamics is a little larger than the correct value at $y = \pi$ but quite accurate at $y = 0$.

While plotting pressure profiles allows us to visualize the difference between the Geometrical Shock Dynamics shock and that computed from Euler simulations, it is worthwhile to introduce a measure of error in Mach number. Because the transmitted shock is not planar, the Mach number at each point on the front differs from the Mach number of the corresponding one-dimensional shock-contact problem, M_T, obtained by setting $a_0 = 0$ and $L = 0$. The value M_T can be calculated easily (see §1.3) and is useful as a rough "average value" of $M(\beta, t)$ along the shock in the two-dimensional problem. The purpose of performing a two-dimensional calculation like Geometrical Shock Dynamics is to obtain information about the shape of and variation in Mach number along the shock. Therefore, we will examine the error in $\tilde{M}(\beta, t) \equiv M(\beta, t) - M_T$ rather than $M(\beta, t)$. Using the shock jump conditions to relate values of pressure at the shock computed from Euler simulations, p_{Euler}, to Mach number, the following measure of relative error at any point along the shock

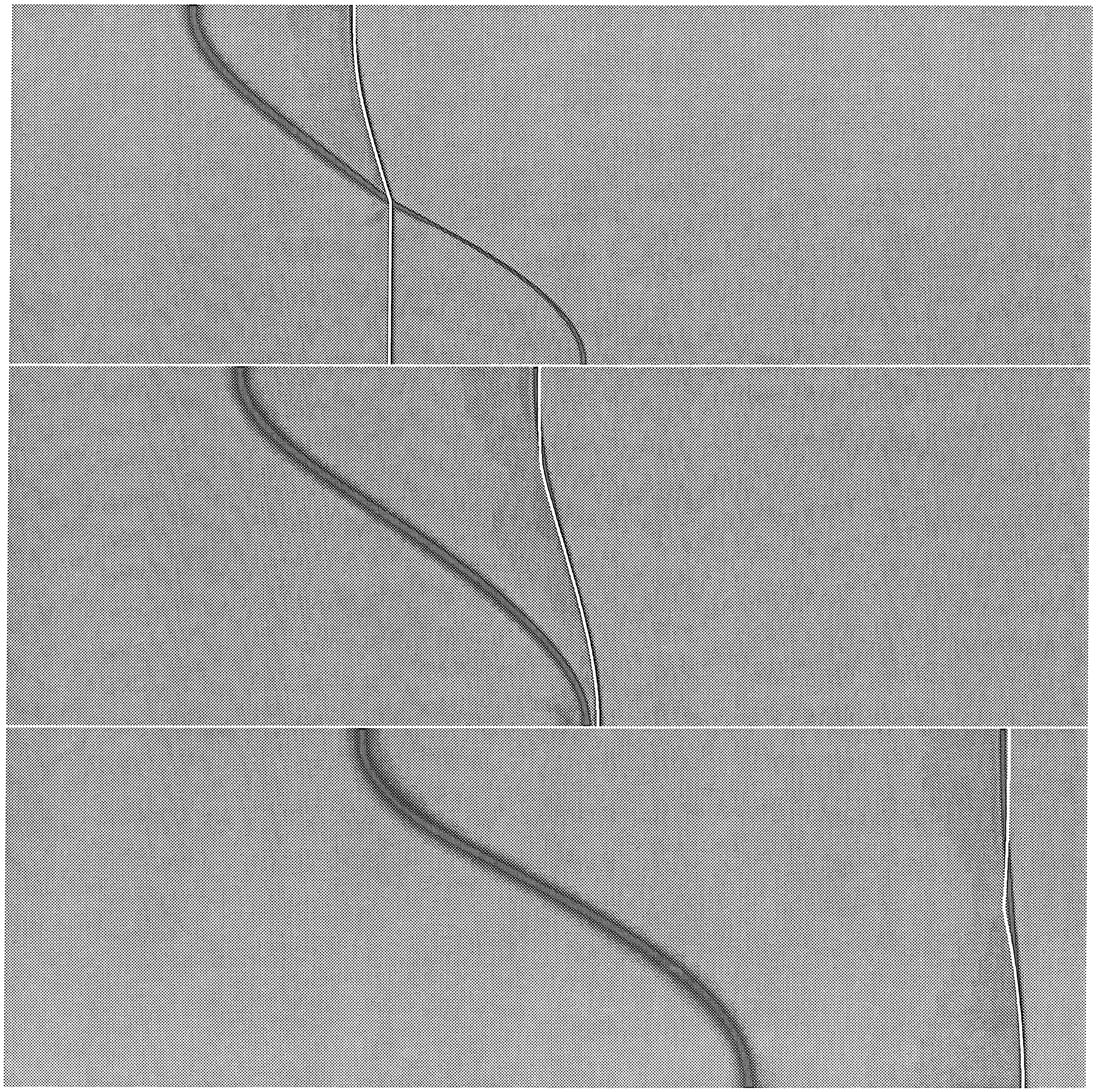
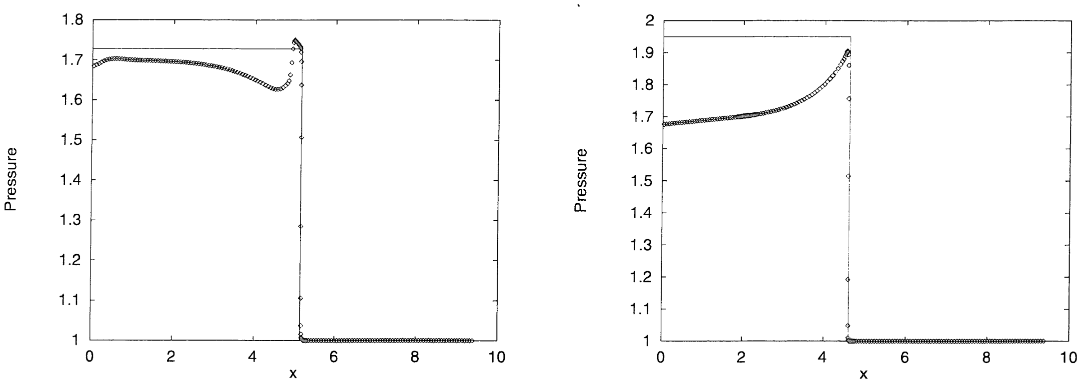


Figure 4.6: Comparison between Geometrical Shock Dynamics (white line) and Euler simulation. ($M_I = 1.25$, $At = 0.2$, $L = 0.01$ and $a_0 = 2$ at $t = 1.2$, $t = 2$, and $t = 4$)



(a) (b)

Figure 4.7: Comparison of pressure profiles from Geometrical Shock Dynamics (solid line) and Euler simulations (diamond symbols) for problem shown in figure 4.6 at $t = 2$: (a) $y = 0$, (b) $y = \pi$.

is obtained:

$$\frac{\left|\tilde{M}_{\mathrm{GSD}} - \tilde{M}_{\mathrm{Euler}}\right|}{\tilde{M}_{\mathrm{Euler}}} = \frac{\left|\sqrt{2\gamma}M_{\mathrm{GSD}} - \sqrt{(\gamma+1)(p_{\mathrm{Euler}}-1)+2\gamma}\right|}{\sqrt{(\gamma+1)(p_{\mathrm{Euler}}-1)+2\gamma} - \sqrt{2\gamma}M_T}, \tag{4.20}$$

where $\tilde{M}_{\mathrm{GSD}} = M_{\mathrm{GSD}} - M_T$, $\tilde{M}_{\mathrm{Euler}} = M_{\mathrm{Euler}} - M_T$, M_{GSD} is the Mach number calculated from Geometrical Shock Dynamics and M_{Euler} is the Mach number from the Euler simulation (calculated from p_{Euler}). We have used the fact that the pressure is equal to one in front of the shock.

Returning to the comparison between Geometrical Shock Dynamics and Euler simulation for $M_I = 1.25$ and $At = 0.2$ (figures 4.6 and 4.7), the peak value of pressure from the Euler data shown in figure 4.7(b) was used as p_{Euler} in equation (4.20). The relative error in $\tilde{M}(\beta, t)$ at $y = \pi$ and $t = 2$ is 26.0%. It must be remembered that this error is much larger than figure 4.7(b) suggests because M_T has been removed in the definition of $\tilde{M}$. There is essentially no error along $y = 0$ for this same instant in time. It is unclear why the error along $y = \pi$ is larger than that at $y = 0$. One possibility is that the error only occurs at a compressive portion of the shock. From figure 4.6 it is seen that the front is concave (compressive) at $y = \pi$ and convex (expansive) at $y = 0$. Another possibility is that the error has developed over time and the error at $y = \pi$ is larger simply because that portion of the shock passed through the layer some time ago allowing more time for the Mach number to deviate from its correct value. To investigate further, the same configuration with a negative Atwood ratio is considered next.

In figure 4.8 the time evolution of the $At = -0.2$ simulation is shown. In the first picture of the figure, the agreement in the shock fronts is not that good near the interface. The fact that the reflected shock is so weak that it has not triggered the refinement criterion does not account for the discrepancy. The shock computed with the Euler code is undergoing irregular refraction and has formed a kink away from the interface. The Geometrical Shock Dynamics front is also undergoing irregular refraction. The transition from regular to irregular refraction arises naturally in Geometrical Shock Dynamics for nonuniform media. Criteria for the transition were obtained by Catherasoo and Sturtevant [9] by examining disturbances propagating along the characteristics of the Geometrical Shock Dynamics equations. This is in contrast to Geometrical Shock Dynamics for uniform media, where regular reflections off solid walls are not allowed [64].

For this combination of problem parameters, the angle between the undisturbed portion of the incident shock and the interface is large enough for disturbances propagating along the $\mathcal{C}^-$ characteristics of the equations of Geometrical Shock Dynamics to have formed a shock-shock on the front to the left of the interface. The disturbances propagating along the $\mathcal{C}^+$ characteristics have created an expansive region on the transmitted portion of the front. For this problem we used the cubic spline implementation of Geometrical Shock Dynamics described in §4.2.1 rather than the Fourier Collocation method. There were differences in the shock fronts computed using the two methods. A close examination of the point distribution along the front in the two calculations revealed that the discrepancy was due to a loss of resolution in the expansive region in the Fourier computation. The point spacing was significantly larger in that portion of the front compared to the rest of the front when computed with the Fourier Collocation approach, effectively increasing the numerical viscosity there. The cubic spline calculation was considered to be the correct one.

As the Geometrical Shock Dynamics computation proceeds from $t = 2.16$ to $t = 2.52$, the shock-shock seen in the first picture of figure 4.8 collides with an equal and opposite one (its "mirror-image") at $y = 0$. The collision causes the shock-shock (and its mirror-image) to propagate back along the front in the direction it came, as shown in the second and third pictures of the figure. The Euler simulation experiences an analogous Mach reflection along the bottom solid wall.

In computing the evolution of the Geometrical Shock Dynamics shock front, we found

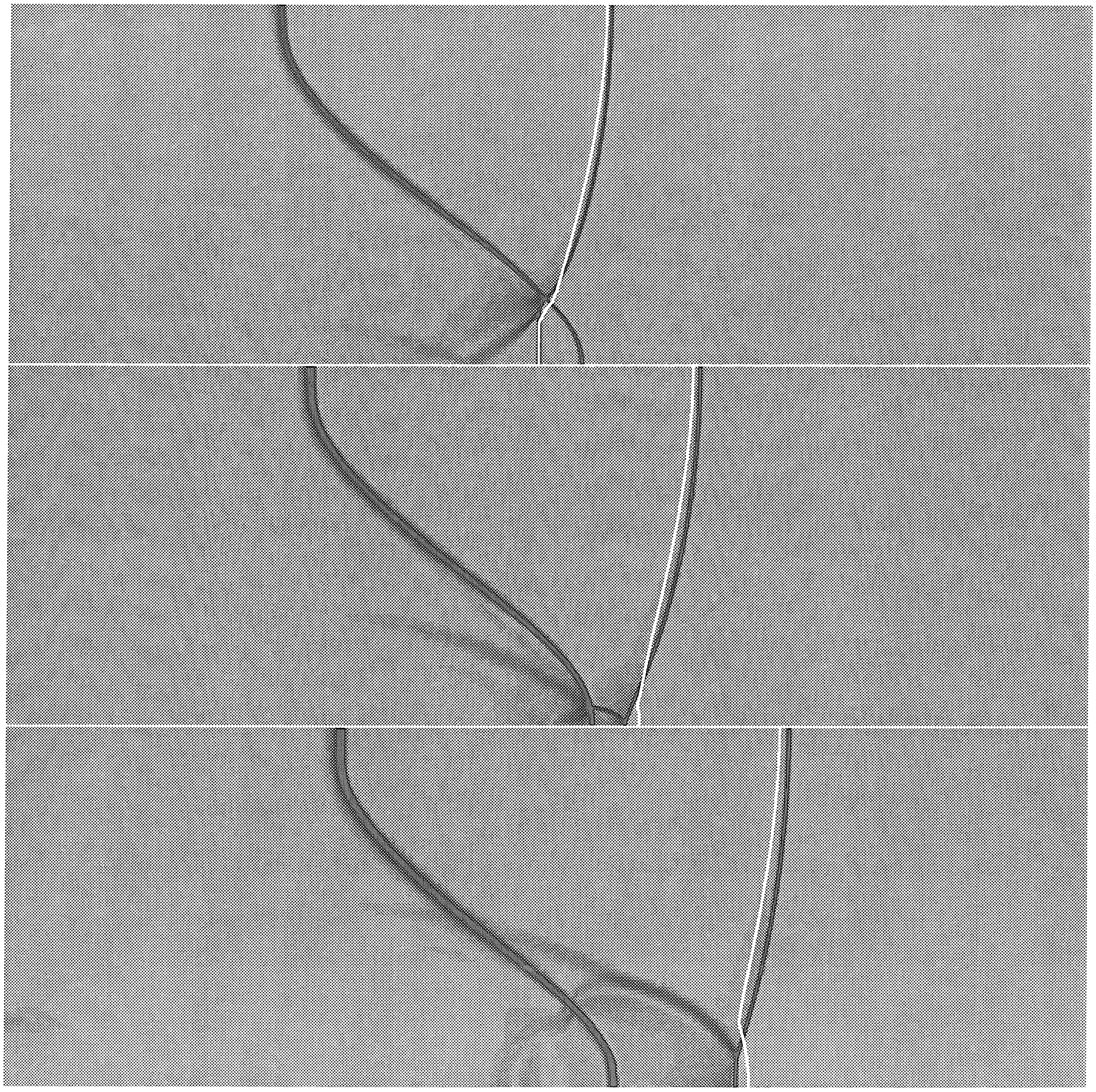


Figure 4.8: Comparison between Geometrical Shock Dynamics (white line) and Euler simulation. ($M_I = 1.25$, $At = -0.2$, $L = 0.01$ and $a_0 = 2$ at $t = 2.16$, $t = 2.52$, and $t = 2.88$)

the Fourier Collocation method was unable to compute past the collision of the shock-shocks unless a relatively large coefficient of viscosity ν was used. The added diffusion can be used to smear the shock-shocks into compression waves. Collision between two continuous compressive disturbances did not pose a problem for our code. It was found, however, that the amount of viscosity needed to smear the shock-shock into compression waves was so large that it altered the entire shock front significantly. The resulting solution was unacceptable. The cubic spline code was able to compute through the shock-shock collision provided the smoothing procedure of Henshaw *et al.*, (4.14), was used. We believe the averaging done in (4.14) applies just enough diffusion at the moment of collision to allow the calculation to continue.

The agreement in shock fronts from the Euler simulation and the Geometrical Shock Dynamics calculation is quite poor at late times, as can be seen in the last picture of figure 4.8. The location of the triple point in the Euler simulation does not agree with the location of the shock-shock on the Geometrical Shock Dynamics front. Geometrical Shock Dynamics has also under-predicted the Mach number at $y = \pi$ which causes the front to lag behind the shock in the Euler simulation. This is in contrast to the $At = 0.2$ case where the theory over-predicted the Mach number along this line. An error in the Mach number is incurred by Geometrical Shock Dynamics at $y = \pi$ regardless of whether the shock front is compressive or expansive there. This suggests that the error at $y = \pi$ is larger than at $y = 0$ after the shock passes through the entire layer simply because there has been more time for errors to accumulate in that portion of the front. If we accept this, it suggests that the nonuniform flow behind the transmitted portion of the shock may be the cause of discrepancy. It is interesting to note that the Mach number at $y = \pi$ computed by Geometrical Shock Dynamics was over-predicted when positive vorticity lay behind the shock and under-predicted for negative vorticity.

We now consider the case of a $M_I = 2$ shock passing through an interface with $a_0 = 2$, $L = 0.01$, and $At = 0.2$. For problems of shock propagation into uniform media, Geometrical Shock Dynamics has generally been considered more accurate for strong shocks than weak ones. However, if nonuniformity behind the transmitted shock really is the reason for the error in $\tilde{M}$ from Geometrical Shock Dynamics calculations, we would see worse agreement in the $M_I = 2$ case than in the $M_I = 1.25$ case. This simulation was performed to compare with the error in the $M_I = 1.25$ case using the same interface parameters. The shock fronts for the $M_I = 2$ simulation are displayed in figure 4.9 and pressure profiles are shown in figure 4.10. Comparing the images in figure 4.9 with those in figure 4.6, we see that increasing the Mach number from $M_I = 1.25$ to $M_I = 2$ has had a detrimental effect. At $t = 2$ in the $M_I = 2$ case the shock fronts computed from Geometrical Shock Dynamics and Euler simulation look completely different. From the pressure profiles in figure 4.10, it is seen that again the Mach number is obtained reasonably well at $y = 0$ right after transmission while the portion of the front which has been in the second gas longer ($y = \pi$) has not. The relative error in $\tilde{M}$ at $y = \pi$ is 52.3%. Comparing with the error at $y = \pi$ in the $M_I = 1.25$ simulation (with $At = 0.2$), we note the relative error is larger: 52.3% compared to 26.0%. The time required for the incident shock to completely traverse the layer is $2a_0/c_0M_I$. The error in $\tilde{M}$ at $y = \pi$, then, is not only larger but it occurs in less time: $\Delta t = 2a_0/2c_0$ compared to $\Delta t = 2a_0/1.25c_0$.

Increasing the incident shock Mach number from $M_I = 1.25$ to $M_I = 2$ has two effects which can explain the increased error in $\tilde{M}$. First, the magnitude of vorticity deposited in the density layer is larger for the $M_I = 2$ case. Second, a comparison of the middle pictures of figures 4.6 and 4.9 clearly shows the post-shock amplitude of the interface is smaller for the stronger shock case. Along $y = \pi$ there is less distance between the transmitted shock and shocked layer for $M_I = 2$. The interface, a region of significant vorticity (clearly nonuniform flow), is closer to the transmitted shock for the $M_I = 2$ case and could, in principle, exert more influence on the evolution of the shock. For larger Mach numbers the agreement between Geometrical Shock Dynamics and Euler simulations is worse and clear differences in shock front shape are observed even before the shock completely passes through the interface.

In figures 4.6 and 4.9 a shock-shock is visible along the Geometrical Shock Dynamics shock fronts while no corresponding Mach stem occurs along the shock in the Euler simulations. For the relatively low Atwood ratio of $At = 0.2$ it is necessary to use stronger incident shocks and larger interface amplitudes to create enough variation in the Mach number along the transmitted shock that a corner will form on the shock. One such simulation was performed to compare the location of the Geometrical Shock Dynamics shock-shock with the corner of the Euler shock. In figure 4.11 the shock front comparison for a problem with incident shock strength $M_I = 4$, large initial amplitude $a_0 = 3$, $At = 0.2$ and $L = 0.01$ is shown. The agreement in shock front shape between Geometrical Shock Dynamics and

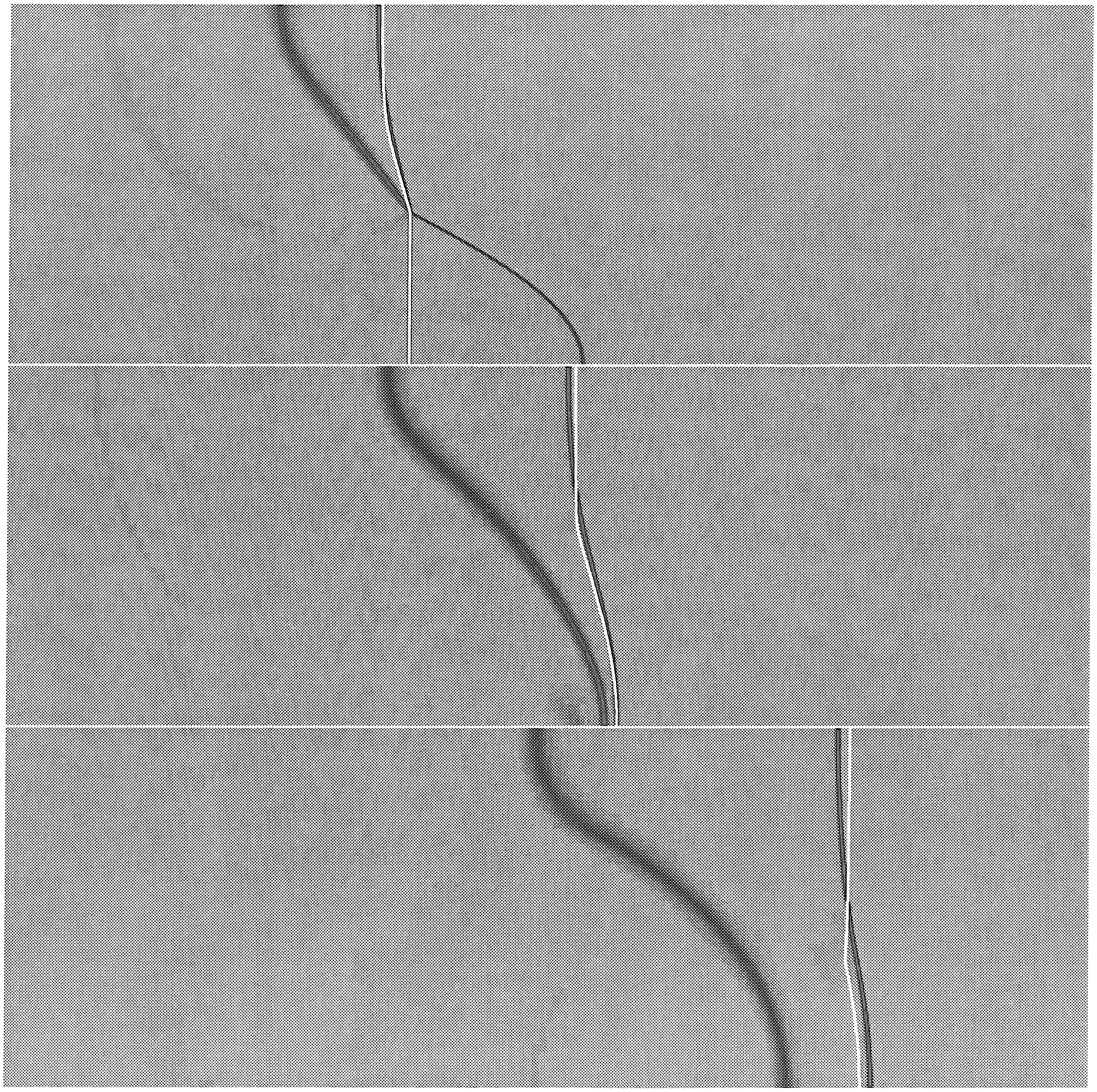


Figure 4.9: Comparison between Geometrical Shock Dynamics (white line) and Euler simulation. ($M_I = 2$, $At = 0.2$, $L = 0.01$ and $a = 2$ at $t = 0.8$, $t = 1.3$, and $t = 2$)

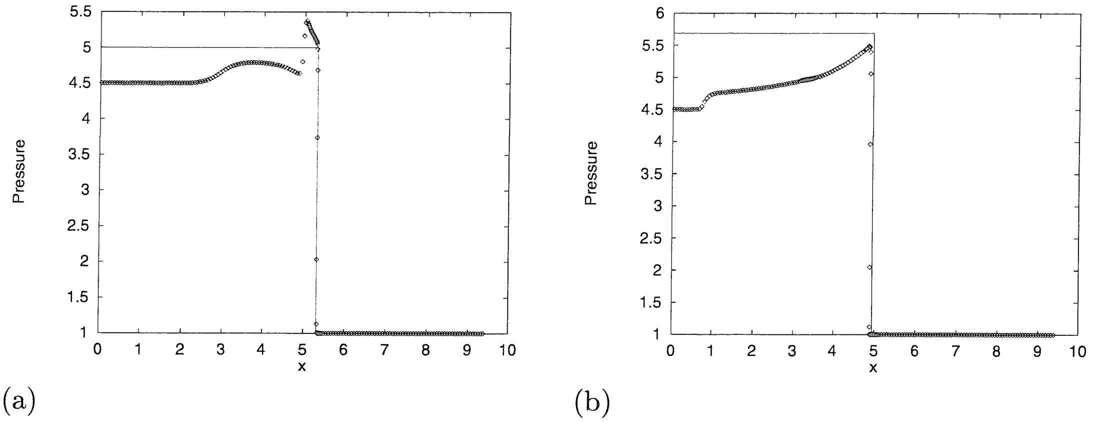


Figure 4.10: Comparison of pressure profiles from Geometrical Shock Dynamics and Euler simulation for simulation shown in figure 4.9 at $t = 1.3$. (a) $y = 0$, (b) $y = \pi$.

Euler simulation is quite poor. The Geometrical Shock Dynamics front forms a shock-shock even before the incident shock is halfway across the interface. At late times, the Geometrical Shock Dynamics shock-shock is not close to the corner seen along the Euler shock.

4.3.2 Calculation of Circulation

In the previous subsection the accuracy of Geometrical Shock Dynamics was evaluated by comparing the shape of transmitted shock fronts and local Mach numbers against data from Euler simulations. However, the motivation for using Geometrical Shock Dynamics was to calculate the vorticity generated by the shock as it passed through the density layer and propagated through the second fluid. In this section we examine the accuracy of Geometrical Shock Dynamics in computing the circulation in the entire flow. This is more convenient than studying pointwise values of vorticity. The vorticity is the driving mechanism of the instability and the circulation is a measure of the mixing. While Samtaney and Zabusky [51] have developed methods to compute the initial circulation deposited on a sharp interface separating two fluids as the shock passes through it, no such work has been done for continuous interfaces. Further, the Samtaney-Zabusky theory provides no information about the time evolution of the circulation as the transmitted and reflected fronts deposit vorticity in the bulk of the fluid as they attempt to stabilize themselves.

At any point in the simulation, Geometrical Shock Dynamics only provides a description of the flow at the shock front. To compute the total circulation in the entire flow field, a

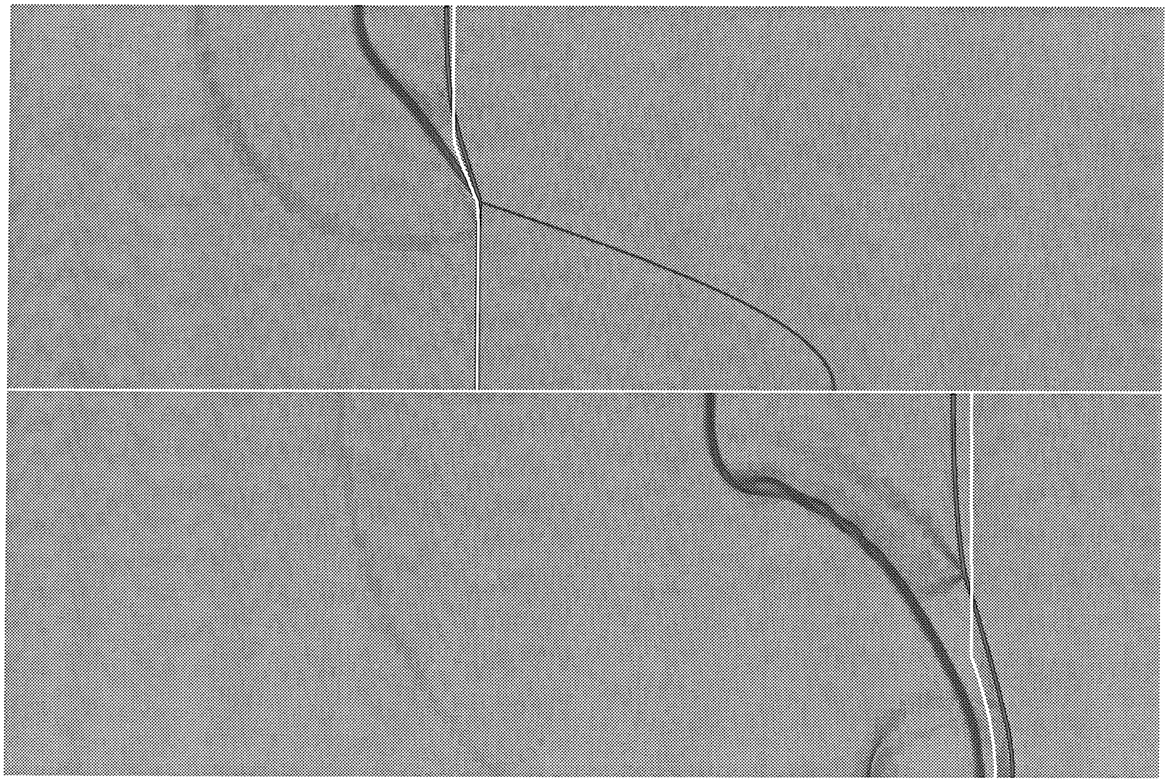


Figure 4.11: Comparison between Geometrical Shock Dynamics (white line) and Euler simulation. ($M_I = 4$, $At = 0.2$, $L = 0.01$ and $a_0 = 3$ at $t = 0.48$ and $t = 1.08$)

strategy was devised. Let $\Gamma(t)$ denote the total circulation over half a period:

$$\Gamma(t) = \int_{-\infty}^{\infty} \int_{0}^{\pi} \omega(x, y, t) dy dx. \tag{4.21}$$

Note while the integral in x is over an infinite interval, at any finite time the vorticity is nonzero only in the finite region between the transmitted shock and reflected wave. Throughout this section only positive Atwood ratios are considered to reduce the number of parameters varied. The reflected wave, then, is always a shock. Two assumptions are made to enable the calculation of $\Gamma(t)$ from Geometrical Shock Dynamics. First, the vorticity in the shocked layer is not significantly modified by secondary effects. That is, the vorticity in the layer is determined solely from the initial baroclinic interaction between the incident shock and the interface. Pressure perturbations bouncing between the transmitted and reflected shocks and the interface do not affect the circulation and triple points (and associated vortex sheets) do not form along the shocks. The small variation in interfacial circulation with time in figures 3.7 and 3.9 indicates this assumption is reasonable. Second, the time evolution of $\Gamma(t)$ (after the initial shock refraction phase) is assumed to be only due to additional vorticity deposited in the fluid away from the interface by the transmitted shock. This additional vorticity is generated by the oscillation of the curved transmitted shock as it propagates. In the Richtmyer-Meshkov problem, the reflected shock also generates vorticity as it propagates. A comparison of the strength of the transmitted and reflected shocks in the one-dimensional shock-contact problem described in §1.3, M_T and M_R respectively, is shown in figure 1.9(a). The values M_T and M_R are good estimates of a rough average value of the local Mach numbers along the corrugated transmitted and reflected shocks in the two-dimensional Richtmyer-Meshkov instability. Figure 1.9 indicates the reflected shock is much weaker than the transmitted shock. This provides some justification for neglecting the vorticity generated by the reflected shock in the circulation calculation.

Figure 4.12 illustrates the strategy in computing $\Gamma(t)$ from Geometrical Shock Dynamics. The circulation at any instant in the computation, $\Gamma(t^{n+1})$, is the total circulation at the previous time step plus the line integral of fluid velocity around the region just swept out by the shock, $\delta\Gamma$. The line integral of velocity around the closed curve is equivalent to the area integral of vorticity over the enclosed region by Stokes' theorem. The strategy is to compute the area integral (4.21) piece by piece in the x-direction. In what follows, the

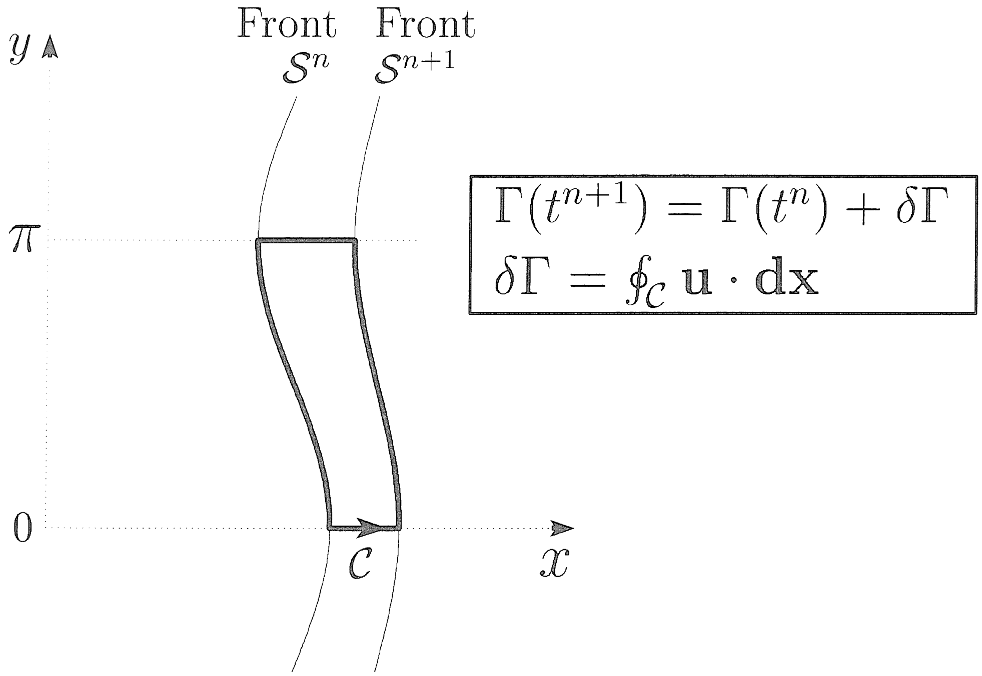


Figure 4.12: Calculation of the circulation in the region using Geometrical Shock Dynamics.

fluid velocity normal to the shock is denoted as u and transverse as v. Directly behind the current shock front $\mathcal{S}^{n+1}$ there is no transverse flow. The line integral $\delta\Gamma$, then, will be the sum of contributions along $y = 0$, $y = \pi$, and along the previous shock front, $\mathcal{S}^n$. These integrals are done numerically using the trapezoidal rule. The integrals along $y = 0$ and $y = \pi$ use only two pieces of data each: the values of u^{n+1} on fronts $\mathcal{S}^{n+1}$ and $\mathcal{S}^n$. Velocities at the current time step along front $\mathcal{S}^n$ are obtained using Taylor series in time. Normal velocities at the shock, u, are always computed from the shock jump conditions.

Expressions for $\partial u/\partial t$, $\partial v/\partial t$, and $\partial^2 v/\partial t^2$ evaluated at a shock front are needed in the Taylor series and can be obtained using the ideas of Best [3, 4]. A local coordinate system is introduced at each point along the front with η the coordinate in the direction of propagation (the normal to the front) and ξ the coordinate along the shock. The two-dimensional Euler equations are transformed to the new (η, ξ) coordinate system through a simple rotation of θ, the angle of inclination of the normal with respect to the x-axis. Algebraic manipulations on the transformed Euler equations, coupled with the geometric relations [4]

$$\frac{\partial \theta}{\partial \xi} = \frac{1}{A}\frac{DA}{D\eta}, \tag{4.22a}$$

$$\frac{\partial \theta}{\partial \eta} = -\frac{1}{M}\frac{\partial M}{\partial \xi}, \tag{4.22b}$$

and the definition of a Lagrangian derivative for a point on the shock, $D/Dt = \partial_t + c_0 M \partial_\eta$, leads to expressions for $\partial u/\partial t$, $\partial v/\partial t$, and $\partial^2 v/\partial t^2$. Details can be found in Best's work. Here we simply state the results:

$$\frac{\partial v}{\partial t} = \frac{c_0 M}{c_0 M - u}\left[\frac{u^2}{M} - \frac{4\gamma M}{(\gamma+1)\rho}\right]\frac{\partial M}{\partial \xi}, \tag{4.23}$$

$$\frac{\partial^2 v}{\partial t^2} = \frac{\partial u}{\partial t}\frac{\partial v}{\partial t} + c_0 M\left[\frac{2u}{M}\frac{\partial u}{\partial t}\frac{\partial M}{\partial \xi} + \frac{1}{\rho^2}\frac{\partial \rho}{\partial t}\frac{\partial p}{\partial \xi}\right], \tag{4.24}$$

$$\frac{\partial u}{\partial t} = \left[\frac{c_0 M}{\rho}\frac{Dp}{Dt} + (c^2 + u(c_0 M - u))\frac{Du}{Dt} + c_0 M c^2 u \frac{A'(\eta)}{A(\eta)}\right] \Big/ \left[c^2 - (c_0 M - u)^2\right], \tag{4.25}$$

where

$$\begin{aligned}\frac{\partial \rho}{\partial t} = \Bigg[c_0 M \frac{Dp}{Dt} + \rho c_0 M (c_0 M - u)\frac{Du}{Dt} + c_0 M \rho u (c_0 M - u)^2 \frac{A'(\eta)}{A(\eta)} \\ - u(c^2 - (c_0 M - u)^2)\frac{D\rho}{Dt}\Bigg] \Big/ \left[(c_0 M - u)(c^2 - (c_0 M - u)^2)\right].\end{aligned} \tag{4.26}$$

The Lagrangian derivatives Dp/Dt, Du/Dt, and $D\rho/Dt$ can be rewritten in terms of DM/Dt and Dc_0/Dt using the shock jump conditions. Best's work only considered shock propagation into uniform media so his expressions did not contain the Dc_0/Dt terms.

$$\frac{Dp}{Dt} = \frac{4\gamma M}{\gamma+1}\frac{DM}{Dt}, \tag{4.27}$$

$$\frac{Du}{Dt} = \frac{2}{(\gamma+1)}\left[c_0\left(1 + \frac{1}{M^2}\right)\frac{DM}{Dt} + \left(M - \frac{1}{M}\right)\frac{Dc_0}{Dt}\right], \tag{4.28}$$

$$\frac{D\rho}{Dt} = \frac{2\gamma(\gamma+1)M}{c_0^2((\gamma-1)M^2+2)}\left[\frac{2}{(\gamma-1)M^2+2}\frac{DM}{Dt} - \frac{M}{c_0}\frac{Dc_0}{Dt}\right]. \tag{4.29}$$

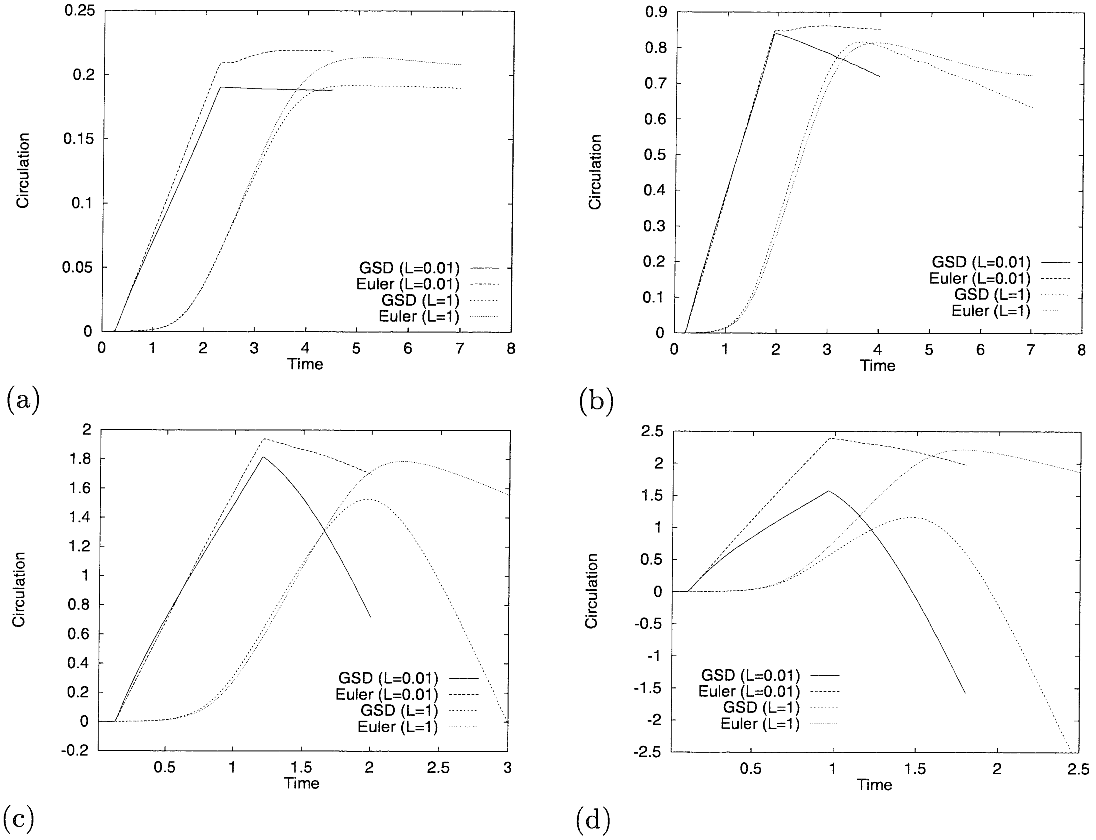
In the Fourier Collocation method derivatives are not computed with respect to ξ but β instead. The needed conversion factor is the local area: $\partial_\xi = (1/A)\partial_\beta$.

An alternative to computing the velocities u and v along the shock front $\mathcal{S}^n$ at t^{n+1} from Taylor series in time is an expansion in space, which would use only information at the current time step. This approach would be more cumbersome for computing the contribution along front $\mathcal{S}^n$. To determine v^{n+1} along $\mathcal{S}^n$, a coordinate rotation would have to be done to account for the fact that the ξ-direction at any point along $\mathcal{S}^{n+1}$ is not the same as the ξ-direction of that same point on $\mathcal{S}^n$. The Taylor series in time approach was chosen and a sufficiently small time step was used to obtain a converged result for $\Gamma(t)$.

In figure 4.13 the results for $\Gamma(t)$ from both Geometrical Shock Dynamics and Euler simulations for $At = 0.2$ and $a_0 = 2$ and a variety of Mach numbers are plotted. In each graph, the results from both the $L = 0.01$ and $L = 1$ cases are shown. The prediction of peak circulation is reasonable for all but the $M_I = 2.5$ case. The relative error in the maximum value for $M_I \leq 2$ is about 10%. In the $M_I = 1.05$ case the circulation changes little with time after the initial vorticity deposition phase. This indicates that the amount of vorticity deposited in the fluids away from the layer is negligible. Note this is true for both the Geometrical Shock Dynamics and Euler simulations. For larger Mach numbers the total circulation in the region decreases after the initial generation of vorticity by the incident shock. This is due to the generation of negative vorticity from the first oscillation of the transmitted shock. While both Geometrical Shock Dynamics and the Euler simulation register the creation of this negative vorticity, the Geometrical Shock Dynamics calculation predicts a sharper decrease in circulation. This is not too surprising in light of the images shown in figures 4.6 and 4.9 showing discrepancies in shock front shape. The over-prediction in Mach number along $y = \pi$ by Geometrical Shock Dynamics results in that portion of the line integral $\delta\Gamma$ having a larger negative contribution than it would had the Mach number been computed accurately. For incident shocks of Mach number $M_I \geq 2.5$, the agreement between Geometrical Shock Dynamics and the Euler simulation in $\Gamma(t)$ is very poor.

The spectacular agreement in the maximum value of Γ for the $M_I = 1.25$ case is somewhat surprising in light of the error along $y = \pi$ shown in figure 4.7(b). It must be remembered that this error was only at one point on the front. The error in Mach number computed from Geometrical Shock Dynamics becomes important later in the simulation and results in incorrect time evolution of $\Gamma(t)$.

Increasing the thickness from $L = 0.01$ to $L = 1$ causes larger errors in the agreement in $\Gamma(t)$. In particular, the Geometrical Shock Dynamics circulation attains its maximum value before the Euler simulation does. In the $M_I = 2$ and $M_I = 2.5$ cases, the relative error in the maximum value of $\Gamma(t)$ is larger for $L = 1$ than $L = 0.01$. There are two explanations



(a) (b) (c) (d)

Figure 4.13: $\Gamma(t)$ computed from Geometrical Shock Dynamics (GSD) and Euler simulation. ($a_0 = 2$, $At = 0.2$, (a) $M_I = 1.05$, (b) $M_I = 1.25$, (c) $= M_I = 2$, (d) $M_I = 2.5$)

for the increased discrepancy. First, the time needed for the shock to completely traverse the layer is nearly twice as long for the $L = 1$ case. Since the discrepancy in front shape between Geometrical Shock Dynamics and Euler simulations increases with time, the values of $\delta\Gamma$ computed at later times may be quite far off from the correct values. Because the Geometrical Shock Dynamics calculation of $\Gamma(t)$ achieves its maximum value before the Euler simulation does, Geometrical Shock Dynamics must be calculating negative values for $\delta\Gamma$ before the shock has completely passed through the layer.

A second explanation is that for thick layers the gap between the region of nonuniform flow (shocked density layer) and the transmitted shock is not as large. The assumption implicit in Geometrical Shock Dynamics of uniform flow behind the shock is very clearly incorrect for thick layers. The vorticity is not concentrated in a thin band, as it is for $L = 0.01$, but it is spread out over a larger area. The swirling flow in the wake of the shock could have a strong effect on the shock simply due to its close proximity.

In figure 4.14 the results for a smaller amplitude interface, $a_0 = 1$, with $L = 0.01$ are shown. As for $a_0 = 2$, the agreement in $\Gamma(t)$ worsens as the incident shock strength is increased. In the $M_I = 2$ case, the maximum value of $\Gamma(t)$ calculated from Geometrical Shock Dynamics is larger than the correct value. This is in contrast to the $a_0 = 2$ case shown in figure 4.13. We were unable to determine why Geometrical Shock Dynamics sometimes over-predicts the maximum of $\Gamma(t)$ and sometimes under-predicts it. By varying the initial amplitude from $a_0 = 1$ to $a_0 = 2$, the $a_0 = 1.5$ case was found to have almost no error in the calculation of maximum vorticity. The time evolution, however, was incorrect. We believe the very small error in initial circulation for this case, as well as the $M_I = 1.25$, $a_0 = 2$ and $M_I = 1.05$, $a_0 = 1$ cases is simply fortuitous and does not indicate cases where Geometrical Shock Dynamics is more accurate in computing the propagation of the shock. While the error in initial circulation is within 10% for $M_I \leq 2$, the time evolution is rarely correct. Thus, there is little merit in using Geometrical Shock Dynamics to calculate a vorticity distribution in the Richtmyer-Meshkov problem.

4.4 Conclusions

Our original intent in applying Geometrical Shock Dynamics to the Richtmyer-Meshkov problem was to develop an easy and quick way to compute the vorticity distribution in the

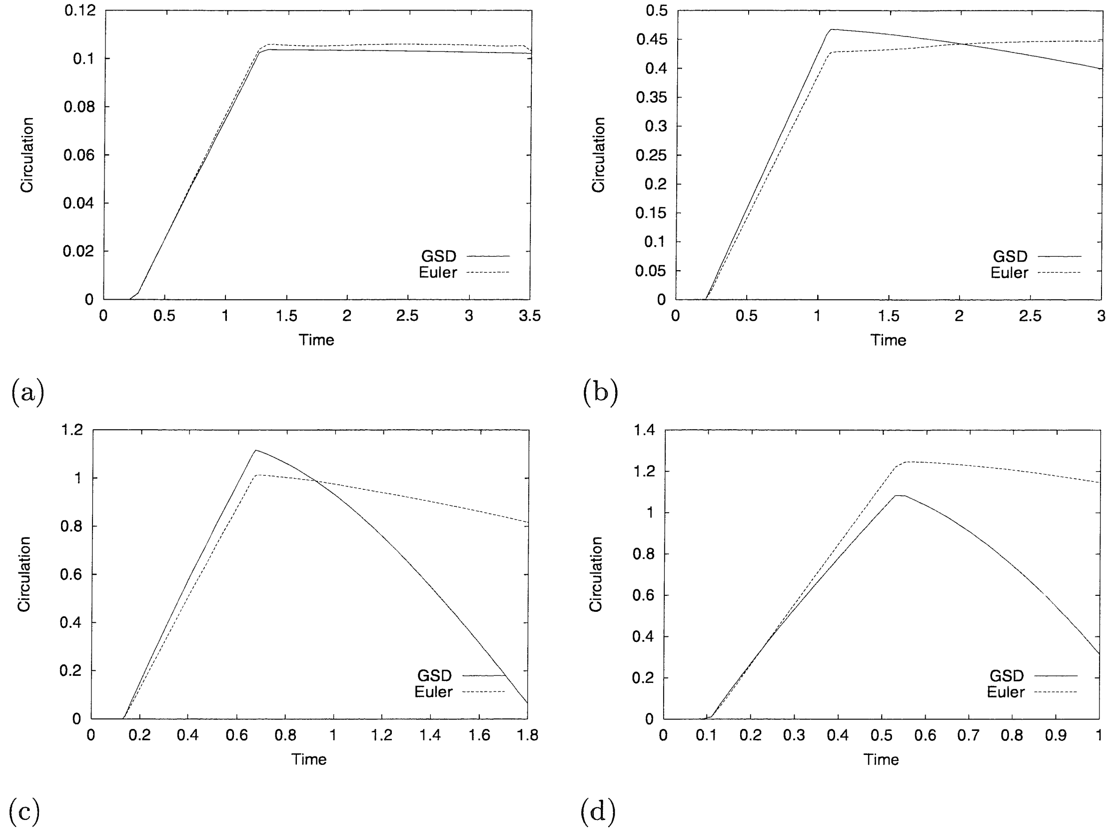


Figure 4.14: $\Gamma(t)$ computed from Geometrical Shock Dynamics (GSD) and Euler simulation. ($a_0 = 1$, $At = 0.2$, (a) $M_I = 1.05$, (b) $M_I = 1.25$, (c) $M_I = 2$, (d) $M_I = 2.5$)

entire flow – particularly for strong shocks. We felt that since the theory of Geometrical Shock Dynamics has been successful in approximating the solution to a wide variety of shock propagation problems and is generally believed to be more accurate for strong shocks, it could be used to accurately compute both the initial interaction between the interface and the incident shock and the deposition of vorticity by the transmitted shock. It was found, however, that the method does not accurately compute the evolution of the transmitted shock. Errors in the local Mach numbers and shape of the shocks are more pronounced for problems involving strong incident shocks. A method for computing circulation in the entire flow from Geometrical Shock Dynamics was developed and implemented. The resulting calculations provided reasonably accurate estimates of the initial circulation deposited in the density layer for Mach numbers less than or equal to 2. The subsequent deposition of vorticity in the flow away from the layer, however, was not calculated accurately as evidenced by the errors in circulation calculations at later times. We conclude that Geometrical Shock Dynamics is not an accurate method to compute the vorticity field in the Richtmyer-Meshkov problem. We believe the discrepancies between Geometrical Shock Dynamics and Euler simulations indicate that the vorticity generated in the diffuse Richtmyer-Meshkov interface – ignored in Geometrical Shock Dynamics – plays an important role in the evolution of the transmitted shock.

Previous work applying Geometrical Shock Dynamics to shock propagation through density stratified fluids concluded that the method was quite accurate, based on comparisons of features of the shock front with experiments. The comparisons shown in figure 4.2 are a well known example of the ability of Geometrical Shock Dynamics to correctly calculate the location of regular and Mach reflections along the shock. Our comparisons, however, were done against Euler simulations, allowing us to perform comparisons of the Mach numbers. Referring to figure 4.2(a) it was found that the slight discrepancy between the shock front computed with Geometrical Shock Dynamics and with the Euler code along the bottom was due to over-prediction of the shock strength. Had this comparison been made against an experimental Schlieren image rather than a numerical image, one would probably have classified the agreement in shock fronts as "very good." Because Geometrical Shock Dynamics was compared against Euler simulations, results could be synchronized exactly, resulting in more accurate comparisons. In addition, the comparisons were done at late enough times in the simulations that errors in the Mach number were clearly visible as incorrect shock front positions. We feel these are some reasons why we have been able to identify the inaccuracy of Geometrical Shock Dynamics for problems involving propagation through media of varying sound speed while other researchers have not.

In searching the literature we have found work by Yin [71] who used Geometrical Shock Dynamics to compute shock-vortex problems. Yin used a formulation of Geometrical Shock Dynamics by Han and Yin [18] to deal with nonuniform flow ahead of the shock and then used a numerical method of characteristics to compute the shape of a shock as it passed through a region containing one or more vortices. Shock fronts were compared with experimental results. In one case, a $M = 1.27$ shock interacted with a counter-rotating vortex pair in vertical alignment. Quoting Yin:

> We can see from comparison between calculation and experiments that shock dynamic description is basically successful. But, detailed comparison shows that the deformation of shock surface calculated by shock dynamics is less than the experimental results. Particularly, in the interaction with counter-rotating vortex pair, shock wave deforms less after passing through the cores and becomes quickly flat. We think that it is because influence of vortex behind shock wave on shock surface is not considered in shock dynamics.

Our results show a similar behavior. One can imagine the shocked interface in the Richtmyer-Meshkov problem as being modeled by an array of vortices located at $y = \lambda/4 + k\lambda/2$, $(k = \dots, -2, -1, 0, 1, 2, \dots)$ of alternating sign. Our simulations show that shock fronts computed from Geometrical Shock Dynamics attempt to straighten themselves more quickly than the shocks computed in the Euler simulations. Often, a shock-shock is created in Geometrical Shock Dynamics before the Euler shock forms an analogous Mach stem. We believe the similarity in Yin's results and ours indicate that the neglect of vorticity behind the transmitted shock in the Richtmyer-Meshkov problem is responsible for the poor agreement in front shape.

In an attempt to consider the influence of swirling flow behind the shock, we turned to the work of Best [3, 4] who developed a method for deriving higher order corrections to Geometrical Shock Dynamics. Using algebraic manipulation, Best showed how equation (4.6a) could be derived as a formal approximation to the two-dimensional Euler equations and how additional differential equations could be added to it to account for nonuniformity behind the shock. The extension of his method for propagation into media of varying sound speed is trivial. While Best's Mach number equation retained the term implicitly truncated from Whitham's A-M relation, equation (4.9), he found he needed to explicitly truncate a term to close the infinite set of differential equations he derived. Thus, Best's theory accounts for some, but not all, of the effect of nonuniform flow on the shock. The method was implemented numerically with the zeroth order equation (the Mach number equation) and one higher order correction equation. Testing it on one-dimensional shock-layer problems, such as those shown in figure 4.4 and 4.5, it was found the method did not compute the Mach number of the transmitted shock accurately. For this reason, we did not apply it to the two-dimensional problem. It would appear, at least in the one-dimensional problem, that some beneficial cancellation of waves bouncing between the shock and the layer occurs and that it is better to consider none of their effects rather than some.

We have found Geometrical Shock Dynamics to be unsuitable to compute the propagation of the transmitted shock in the Richtmyer-Meshkov problem. We believe that the shortcomings of Geometrical Shock Dynamics result from omitting the effects of vorticity behind the shock. Although we have not provided concrete proof of this, we believe the results of Yin [71] provide independent evidence. For problems involving stronger shocks, the influence of vorticity on the shock increases and Geometrical Shock Dynamics is unsuitable for such problems even for early times.

Chapter 5

Conclusions

We have attempted to gain more insight into the problem of Richtmyer-Meshkov instability occurring in continuously stratified fluids by comparing results obtained from simplified models against those from simulations of the full nonlinear Euler equations. Analytic results have been derived [15, 62, 68] for the case of small amplitude interfaces of zero thickness but exact formulas for the asymptotic growth rate exist only in the weak shock limit. Approximate formulas and techniques exist for computing the growth rate induced by shocks of arbitrary intensity; however, one could argue that such results do little to advance our understanding of the physics of the instability [31]. In addition, the usefulness of such formulas to problems of technological interest, where the interfacial amplitudes are not necessarily small compared to the wavelength of perturbations, is somewhat limited. Models are a popular way to try to explain and predict features of the Richtmyer-Meshkov instability in a simple way by focusing on the most important effects in the flow and neglecting the rest.

Disagreement between experimentally measured growth rates and those obtained from theory and numerical simulation have yet to be resolved adequately [47]. Recent work [63] has provided some evidence that the influence of membranes used in experiments is significant, prompting some researchers to perform experiments not requiring membranes [5, 6, 34]. While there exists a considerable body of work devoted to the theory of Richtmyer-Meshkov instabilities of a sharp interface, relatively little work has been done to study cases where the interface is diffuse [6, 30, 36, 49]. This work was an attempt to further the theory of Richtmyer-Meshkov instabilities occurring in continuously stratified fluids.

In chapter 3 the accuracy of an impulse model due to Saffman and Meiron [49] was studied. In contrast to the popular Richtmyer impulse model, no restrictions are placed on the shape or thickness of the interface. The model was implemented numerically and results were compared against simulations of the full nonlinear Euler equations. It was shown that the impulse approximation, known to be accurate for sharp sinusoidal interfaces of infinitesimal amplitude [15], calculates growth rates and late time circulations which are in excellent agreement with Euler simulations of problems involving interfaces of finite amplitude and thickness in the weak shock limit. The model was extended to stronger shock cases with the use of post-shock Atwood ratios, amplitudes and layer thicknesses. Comparisons of growth rates calculated from the model and Euler simulations showed trends similar to those observed by Yang *et al.* [69] between the Richtmyer model and simulation of the linearized Euler equations. The agreement in growth rate for high Atwood ratio cases is quite good while the model under-predicts the growth for lower Atwood ratio cases. Comparison between circulation calculations and vorticity distributions, however, have shown that the impulse model is actually not accurate for either Atwood ratio case, contrary to popular belief. The influence of vorticity away from the layer, not accounted for in the model, on the growth rate was shown to be non-negligible for problems involving strong shocks. These results indicate that the accurate growth rate predictions from the model in the high Atwood ratio case are most likely due to a fortuitous cancellation between vorticity in the layer and vorticity in the bulk of the fluids. It was shown that compressibility effects are only important at early times, after which the growth rate of the interface is calculated accurately by the Biot-Savart law with the correct vorticity distribution. This is true even for problems with very strong incident shocks and large amplitude interfaces, where the resulting flow is quite complex.

The theory of Geometrical Shock Dynamics was implemented numerically to calculate the shock refraction phase and subsequent evolution of the transmitted shock. The method propagates the front cheaply by neglecting the influence of the flow behind, reducing the dimension of the problem by one. A method for computing total circulation in the region from Geometrical Shock Dynamics was proposed and implemented. For incident shocks of moderate strength the predictions of maximum circulation were within 10% of results from Euler simulations. However, Geometrical Shock Dynamics does not calculate the motion of the shock at late times accurately and the negative vorticity deposited in the bulk of the fluid by the oscillation of the transmitted shock is too large. Detailed comparison of shock fronts and local Mach numbers computed with Geometrical Shock Dynamics and Euler simulations have shown disagreement at late times. For stronger shocks, the agreement worsens and the calculation of the initial circulation is no longer accurate. This is counter to the popular belief that Geometrical Shock Dynamics is more accurate for strong shocks.

We believe that we have been able to identify the inaccuracy of the method by running our simulations out to long enough times that errors in local Mach number, which we have been able to detect by directly comparing with pointwise pressures obtained from Euler equations, are clearly visible as discrepancies in shock front shape. We believe the errors resulting from Geometrical Shock Dynamics are due to the neglect of the influence of vorticity in the wake of the shock on the evolution of the front. Work by Yin [71] on shock-vortex interactions using Geometrical Shock Dynamics has shown similar discrepancies in the shock front when compared to experiments. While Geometrical Shock Dynamics appears to compute the shock refraction phase adequately for moderate strength shocks, in general it is not an accurate way to compute the vorticity distribution of the flow.

The impulse approximation, analyzed in chapter 3, attempts to predict the asymptotic growth rate of the interface and circulation without explicitly calculating the shock refraction phase. It does not track the subsequent evolution of the transmitted and reflected shock fronts thus ignoring the vorticity left in their wake as they propagate. Conversely, Geometrical Shock Dynamics **does** calculate the shock refraction phase but then ignores any effect the shocked, vorticity-laden layer might have on the transmitted shock front. Since neither model really simulates the instability accurately, it appears the shocks and the shocked density layer affect the evolution of each other. This does not bode well for models that attempt to analyze and simplify the Richtmyer-Meshkov instability by treating the shocks and interfaces separately.

Most numerical methods for the compressible Euler equations place a restriction on the time step allowed to ensure that signals in the flow do not propagate more than one cell length in a single time step (the CFL criterion). For the Euler equations, the signal speed is the sum of the local flow velocity and sound speed. In the Richtmyer-Meshkov problem, the relative incompressibility of the flow between the transmitted and reflected shocks suggests that the late time evolution of the instability could be more efficiently computed using numerical methods specifically designed for incompressible flow. The time step in a compressible simulation would be restricted primarily by the sound speed, resulting in a very stiff time integration. Ideally, one could generate an initial vorticity field that closely approximates the true distribution at the end of the compressible phase and use this as an initial condition for an incompressible simulation. This idea has been used by Pham and Meiron [36] to study the late time growth of a thick, multimode density layer.

They used the Saffman-Meiron impulse model to generate the initial flow caused by a weak incident shock. In principle, a similar approach could be done to compute the resulting flow initiated by a stronger shock. However, the impulse approximation does not compute the correct vorticity distribution. A model that captures more of the interaction between the shocks and layer is needed to compute the correct vorticity distribution both in the layer and in the bulk of the fluid, especially for interfaces of large amplitude. Analytic models for large amplitude interfaces are likely to be quite involved, however, due to the complex interactions between the transmitted and reflected shocks and the interface through pressure perturbations and reflected shocks from triple points. It may be possible to model the instability through vortex dynamics, however, determining the vorticity distribution will certainly require an approach more sophisticated than the popular impulse models. It may be that the only feasible way to compute the vorticity distribution is to perform a high resolution compressible simulation to calculate the initial phase of the instability. The compressible calculation could be stopped at an appropriate time and the solution fed into an incompressible code. Such an approach would be computationally less expensive than using a compressible flow algorithm to perform the entire simulation. This approach may be cumbersome, however, since work would need to be done on developing an accurate procedure for projecting the compressible flow solution onto a divergence free space.

The real challenge in the field of Richtmyer-Meshkov instability is to develop a predictive capability that will allow scientists and engineers to design inertial confinement fusion capsules that minimize the gross mixing of materials that occurs in these experiments. In these situations, the Richtmyer-Meshkov instability is generated with strong shocks, very high Atwood ratios and multimode interfaces. The deuterium-tritium fuel pellet is compressed by means of an imploding spherical shock. Typically, these problems experience multiple shocks as the initial shock rebounds from the center and propagates outward through the mixing region. These additional shocks greatly enhance the mixing process, resulting in an inefficient burn of fuel. A Cartesian-geometry simulation demonstrating the re-shocking phase is shown in figure 5.1. A $M_I = 8$ shock initiates an instability across a multimode interface of Atwood ratio $At = 0.8$ and thickness $L = 0.1$. The transmitted shock rebounds off the right boundary which has been coded as a reflecting wall. As this reflected shock passes through the already unstable interface, the vorticity in the region is increased significantly, leading to gross mixing at late times.

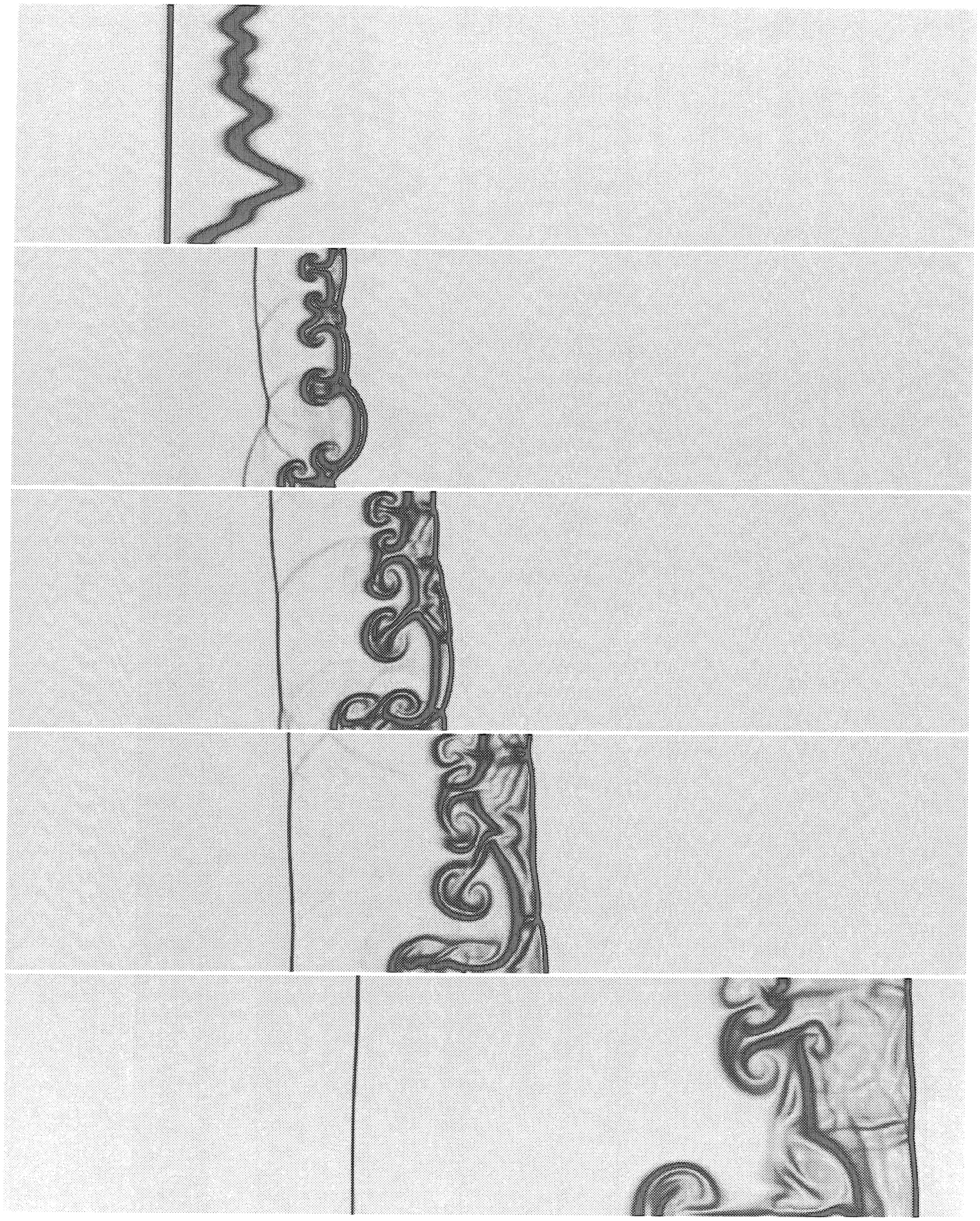


Figure 5.1: Instability of a multimode interface with reflecting end wall. (Continued on next page. $M_I = 8$, $At = 0.8$, $L = 0.1$)

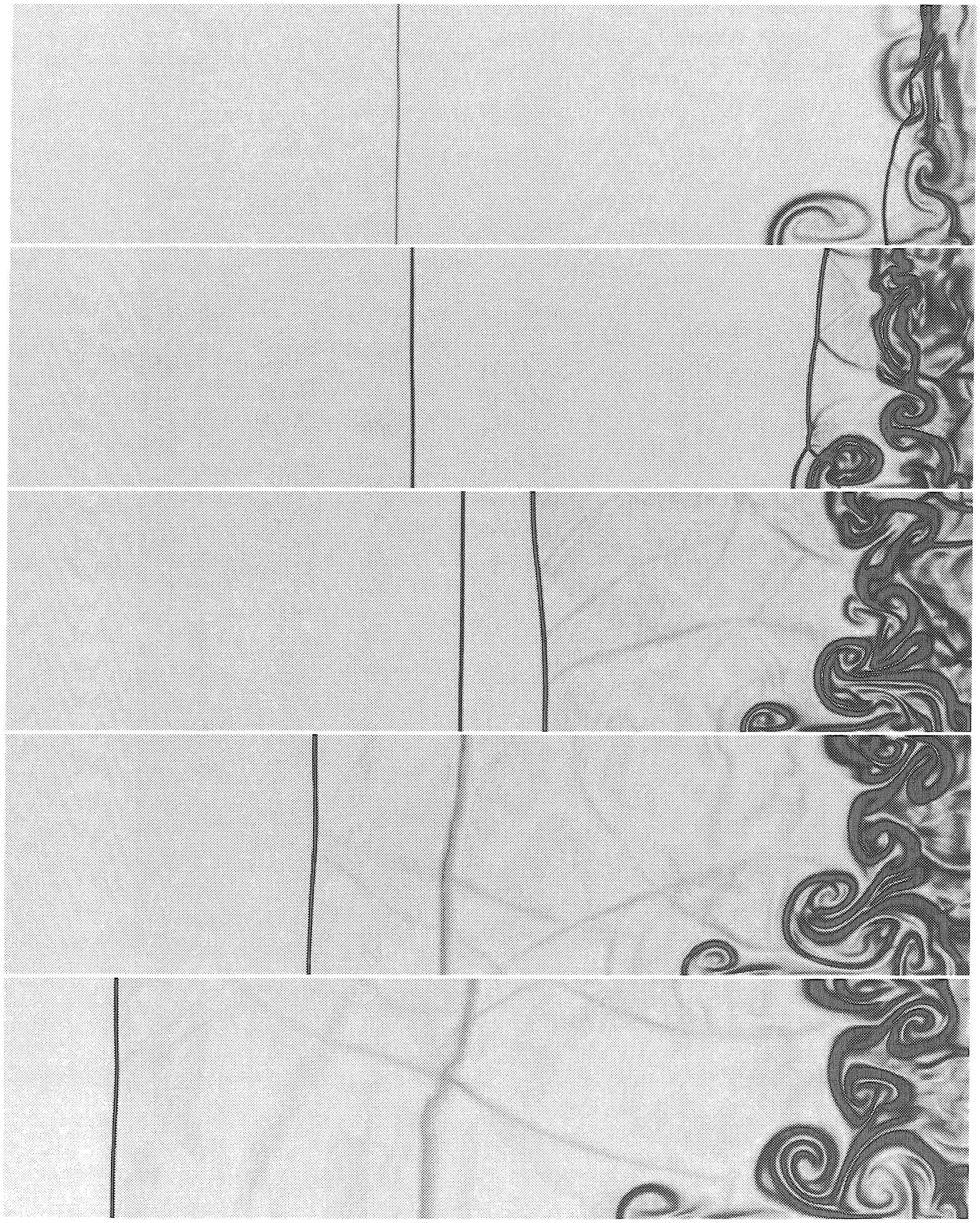


Figure 5.1 continued.

Appendix A

Adaptive Mesh Refinement Algorithm

Euler simulations in this thesis have been performed with the new computational fluid dynamics software package *Amrita*. The computational module of the software, *Amr_sol*, features adaptive mesh refinement. The specific algorithm used is that due to Quirk [39]. For completeness, a brief description of the algorithm is presented here. The following is an excerpt from the documentation provided with *Amrita* and is reprinted with permission of the author. A few small typographical changes have been made to the original text solely for compatibility with the rest of this thesis. It should be noted this documentation was written before the public release of *Amrita* version 1.34 which could perform computations on computers with parallel architectures.

The module *Amr_sol* is based upon a general purpose Adaptive Mesh Refinement (AMR) algorithm for integrating systems of hyperbolic partial differential equations. This algorithm attempts to reduce the costs of a simulation by matching the local resolution of the computational grid to the local requirements of the solution being sought. For example, in simulations of gas dynamic flows, a fine mesh would be used only in the vicinity of shock waves and other flow discontinuities, leaving a coarse mesh elsewhere. Although the savings which accrue from this technique are entirely problem dependent, they can be every bit as attractive as those gained from using expensive parallel computers (savings of more than five hundred-fold have been obtained for simulations of detonation phenomena [39]).

The AMR algorithm employs a hierarchical grid system. In the following, the term 'mesh' refers to a single topologically rectangular patch of cells and the term 'grid' refers to a collection of such patches. At the bottom of the hierarchy a set of coarse mesh patches delineates the computational domain. These patches form the grid G_0 and they are restricted such that there is continuity of grid lines between neighboring patches. This domain may be refined locally by embedding finer mesh patches into the coarse grid G_0. These embedded patches form the next grid in the hierarchy, G_1. Each embedded patch is effectively formed by subdividing the coarse cells of the patches that it overlaps. The choice for the refinement ratio is arbitrary, but it must be the same for all the embedded patches. Thus, by construction, the grid G_1 also has continuity of grid lines. This process of adding grid tiers to effect local refinement may be repeated as often as desired; see figure A.1.

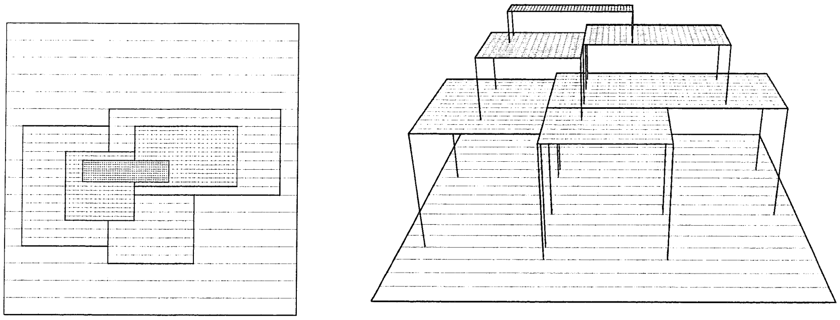


Figure A.1: *Amr_sol* employs a hierarchical grid system.

From stability considerations, many numerical schemes have a restriction on the size of time step that may be used to integrate a system of equations. The finer the mesh, the smaller the allowable time step. Consequently, the AMR algorithm refines in time as well as space. More but smaller time steps are taken on fine grids than on coarse grids in a fashion which ensures that the rate at which waves move relative to the mesh (the Courant number) is comparable for all grid levels. This avoids the undesirable situation where coarse grids are integrated at very small Courant numbers given the time step set by the finest grid's stability constraints: some schemes (e.g., Lax-Wendroff) give poor accuracy for small Courant numbers.

The field solution on each grid is retained even in regions of grid overlap and so all grid levels in the hierarchy coexist. The order of integration is always from coarse to fine since it is necessary to interpolate a coarse grid solution in both time and space to provide boundary conditions for its overlying fine grid. The various integrations at the different grid levels are recursively interleaved to minimize the span over which any temporal interpolation need take place. Periodically, for consistency purposes, it is necessary to project a fine grid solution on to its underlying coarse grid. Figure A.2 shows the sequence of integration steps and back projections for a three level grid $\{G_0, G_1, G_2\}$ with refinement ratios of 2 and 4.

GRID INTEGRATION	TIME STEP	PROJECTION	ADAPTION
G_0	Δt		
G_1	$\Delta t/2$		
$4 \times G_2$	$4 \times \Delta t/8$		
		$G_2 \rightarrow G_1$	
			G_2
G_1	$\Delta t/2$		
$4 \times G_2$	$4 \times \Delta t/8$		
		$G_2 \rightarrow G_1$	
		$G_1 \rightarrow G_0$	
			G_2
			G_1

Figure A.2: Grid operations are recursively interleaved (to be read from top to bottom).

The integration of an individual grid is extremely simple in concept. Each mesh is surrounded by borders of dummy cells. Prior to integrating a grid, the dummy cells for every mesh patch in the grid are primed with data which is consistent with the various boundary conditions that have to be met. Each mesh patch is then integrated independently by an application dependent, black-box integrator that never actually sees a mesh boundary. In principle any cell-centered scheme developed for a single topologically rectangular mesh could form the basis for the integration process.

In general, it is necessary to adapt the computational grid to the changes in the evolving flow solution and so the grid structure is dynamic in nature. Monitor functions based on the local solution are used to determine automatically where refinement needs to take place to resolve small scale phenomena. For a simple example, figure A.3 shows several snapshots taken from the simulation of a shock wave diffracting around a corner. Each snapshot shows the outlines of the mesh patches which go to make the finest grid. This grid clearly conforms to the main features of the flow, namely the diffracted shock front and the vortex located at the apex of the corner. Although the changes in grid structure shown here are dramatic, many adaptions have taken place between each frame (the mesh patches appear small, but each patch actually contains several hundred cells).

A large number of small grid movements occurs because the adaption process dovetails with the integrations process; see figure A.2. Observe that the adaption always proceeds from fine to coarse so as to ensure that there is never a drop of more than one grid level at

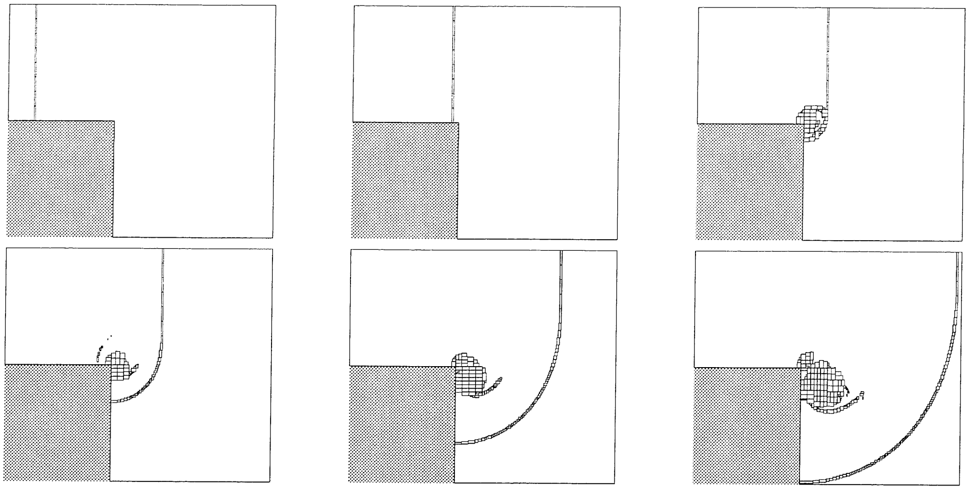


Figure A.3: *Amr_sol* employs a dynamic grid system.

the edge of a fine grid to the underlying coarse grid. A grid adaption essentially produces a new set of mesh patches which must be primed with data from the old set of patches before the integration process can proceed. Where a new patch partially overlaps an old patch of the same grid level, for the region of overlap, data may be simply shoveled from the old patch to the new patch. In regions of no such overlap, the required field solution is found by interpolation from the underlying coarse grid solution.

In a typical application the finest grid will contain several hundred mesh patches. Consequently the mesh patch is a sufficiently fine unit of data for efficient parallelism. The parallel version of *Amr_sol*[*] [39] is implemented using a Single Program Multiple Data (SPMD) model. Each processing node executes the basic serial algorithm in isolation from all other nodes, except that at a few key points messages are sent between the nodes to supply information that an individual node deems to be missing, that is off-processor. For example, during the integration of a grid, the only point at which a processor needs to know about other processors is during the priming of the dummy cells. Whereas in a serial computation all data fetches are from memory, for a parallel computation some are from memory and some necessitate receiving a message from another processor. Each time the grid adapts, the algorithm generates a schedule of tasks that have to be performed so as to prime correctly the dummy cells of a given grid. If running in parallel, this schedule is parsed to produce a schedule of those tasks that necessitate off-processor fetches. At which

---

[*] This is not yet available for public consumption.

point, individual processors can exchange subsets of their fetch schedules, as appropriate, so that every node can construct a schedule of messages that it must send out at some later date. Thus, the priming process is carried out in two phases. First, all the local data fetches are performed as for the serial case. Second, each node sends out the data that has been requested of it. The node then waits for those data items it has requested. For each incoming message it can readily determine from its own schedules what to do with the off-processor data, and so the order in which messages arrive is unimportant. The adaption process and the back projection of the field solution between grid levels also necessitate sizeable amounts of communication; these are handled in a similar fashion to the priming of the dummy cells.

The problem of load balancing the AMR algorithm rests on determining the best distribution of the new patches amongst the processing nodes before the new field solution is interpolated from the old field solution. Currently, this is done using heuristic procedures which bear strong similarities to classical 'bin packing'.

Appendix B

Numerical Flow Visualization

The images of the flow field computed from Euler simulations have been created with the `SchlierenImage` standard library routine in the computational fluid dynamic software package *Amrita*. As the name implies, the procedure creates an image of the computational domain designed to resemble Schlieren images obtained from experiments. Experimental Schlieren images visualize high density gradients as dark regions. The `SchlierenImage` subroutine creates a greyscale image with the following function:

$$\begin{aligned} \mathcal{G}_{i,j} &= \mathcal{E} e^{-\mathcal{A}\mathcal{W}_{i,j}} \\ \mathcal{W}_{i,j} &= \frac{||\nabla\rho||_{i,j} - ||\nabla\rho||_{\min}}{||\nabla\rho||_{\max} - ||\nabla\rho||_{\min}} \end{aligned}$$

where $\mathcal{E}$ and $\mathcal{A}$ are parameters set by the user and $||\nabla\rho||_{\min}$ and $||\nabla\rho||_{\max}$ are the minimum and maximum values of $||\nabla\rho||_{i,j}$ in the computational domain. The value of $||\nabla\rho||_{i,j}$ is calculated with central differences. Each cell is assigned a specific shade of grey $\mathcal{G}_{i,j}$ between 0 and 1 where $\mathcal{G}_{i,j} = 1$ is pure white and $\mathcal{G}_{i,j} = 0$ is pure black.

Parameter $\mathcal{E}$ is called the "exposure" and, like its name suggests, controls the darkness of the image. The default value of $\mathcal{E} = 0.8$ was used for most images in this thesis except those in chapter 4 where a value of $\mathcal{E} = 0.5$ was used to produce a darker image and create greater contrast between the white shock computed from Geometrical Shock Dynamics and the flow field calculated from Euler simulations. Parameter $\mathcal{A}$ is the "amplification factor" and defaults to $\mathcal{A} = 15$. As $\mathcal{A}$ is increased, weaker and weaker density gradients become darker and more visible in the image. It should be noted that since shocks are smeared out over several grid cells, increasing the value of $\mathcal{A}$ thickens these features in the image.

Appendix C

Normal Mode Solutions of Geometrical Shock Dynamics

Geometrical Shock Dynamics is an approximate theory and does not give exact solutions to the Euler equations. One example concerns the stability of shocks. Roberts [42] showed that perturbations of a shock front decay with time. The following analytic solution to the equations of Geometrical Shock Dynamics shows the theory predicts small perturbations of a front to oscillate with fixed amplitude indefinitely.

We consider propagation into uniform media and normalize the sound speed to 1. The problem is periodic in y with period 2π. The equations of Geometrical Shock Dynamics are:

$$\frac{Dx}{Dt} = Mn_x, \tag{C.1a}$$

$$\frac{Dy}{Dt} = Mn_y, \tag{C.1b}$$

$$\frac{DM}{Dt} = -\frac{M^2-1}{M\lambda(M)}\frac{1}{A}\frac{DA}{Dt}, \tag{C.1c}$$

where n_x and n_y are the x and y components of the unit normal to the front. Introducing a Lagrangian parameter along the shock, β, and small parameter, ϵ, we linearize about the solution of a planar shock of Mach number M_0 propagating through uniform media:

$$x(\beta,t) = M_0 t + \epsilon\tilde{x}(\beta)e^{\sigma t}, \tag{C.2a}$$

$$y(\beta,t) = \beta + \epsilon\tilde{y}(\beta)e^{\sigma t}, \tag{C.2b}$$

$$M(\beta,t) = M_0 + \epsilon\tilde{M}(\beta)e^{\sigma t}. \tag{C.2c}$$

The functions $\tilde{x}$, $\tilde{y}$, and $\tilde{M}$ and the growth rate σ are to be determined.

Normals to the front and differential area elements are determined from the local geometry. Expressions for these quantities are given by equations (4.16)-(4.18). Substituting the form of the solutions, equations (C.2a)-(C.2c), into the geometrical relations, equations (4.16)-(4.18), and keeping only first order terms in ϵ results in

$$n_x = 1, \tag{C.3a}$$

$$n_y = -\epsilon \tilde{x}' e^{\sigma t}, \tag{C.3b}$$

$$\frac{1}{A}\frac{DA}{Dt} = \sigma \epsilon \tilde{y}' e^{\sigma t}, \tag{C.3c}$$

where $\tilde{x}'$ and $\tilde{y}'$ denote differentiation with respect to β.

Using (C.3a) and (C.3b), together with (C.2a)-(C.2c) simplifies (C.1a) and (C.1b) to:

$$\sigma \tilde{x} = \tilde{M}, \tag{C.4a}$$

$$\sigma \tilde{y} = -M_0 \tilde{x}'. \tag{C.4b}$$

We must simplify equation (C.1c). Because $4 < \lambda(M) < 1+2/\gamma+\sqrt{2\gamma/(\gamma-1)} \approx 5.0743$ (for $\gamma = 1.4$) we approximate $\lambda(M) \approx \lambda(M_0) \equiv \lambda_0$. Substituting (C.2c) and (C.3c) into (C.1c) leads to:

$$\tilde{M} = \frac{1 - M_0^2}{\lambda_0 M_0} \tilde{y}'.$$

An equation solely in terms of $\tilde{M}$ can be obtained using (C.4a) and (C.4b) to eliminate $\tilde{y}'$:

$$\tilde{M} = \frac{C^2}{\sigma^2} \tilde{M}'', \tag{C.5}$$

where $C^2 = (M_0^2 - 1)/\lambda_0$. The solution of (C.4a), (C.4b), and (C.5) is:

$$\tilde{x}(\beta) = \frac{1}{\sigma}\left[A_1 \exp\left(\frac{\sigma\beta}{C}\right) + A_2 \exp\left(-\frac{\sigma\beta}{C}\right)\right], \tag{C.6a}$$

$$\tilde{y}(\beta) = -\frac{M_o}{\sigma C}\left[A_1 \exp\left(\frac{\sigma\beta}{C}\right) - A_2 \exp\left(-\frac{\sigma\beta}{C}\right)\right], \tag{C.6b}$$

$$\tilde{M}(\beta) = A_1 \exp\left(\frac{\sigma\beta}{C}\right) - A_2 \exp\left(-\frac{\sigma\beta}{C}\right), \tag{C.6c}$$

where A_1 and A_2 are arbitrary constants. The problem is periodic in y with period 2π so both $\tilde{x}(\beta)$ and $\tilde{M}(\beta)$ are periodic. Let $\sigma/C = ik$ where k is any integer. C is purely real ($C^2 > 0$) so σ, the growth rate in equations (C.2a)-(C.2c), is pure imaginary. The full solution is:

$$x(\beta,t) = M_0 t + \frac{\epsilon}{kC}\left[(A_1 - A_2)\sin(k\beta)\cos(kCt) + (A_1 + A_2)\cos(k\beta)\sin(kCt)\right], \tag{C.7a}$$

$$y(\beta,t) = \beta - \frac{\epsilon M_0}{kC^2}\left[(A_1 + A_2)\sin(k\beta)\cos(kCt) + (A_1 - A_2)\cos(k\beta)\sin(kCt)\right], \tag{C.7b}$$

$$M(\beta,t) = M_0 + \epsilon\left[(A_1 + A_2)\cos(k\beta)\cos(kCt) - (A_1 - A_2)\sin(k\beta)\sin(kCt)\right]. \tag{C.7c}$$

Suppose the initial condition of a Geometrical Shock Dynamics problem is a sinusoidal shock of small amplitude and uniform Mach number. This choice selects the particular solution with constants $A_1 = -A_2 = \frac{1}{2}kCA_0$, where ϵA_0 is the small initial amplitude of the front. The resulting solution at any time t is:

$$x(\beta,t) = M_0 t + \epsilon A_0 \cos(kCt)\sin(k\beta), \tag{C.8a}$$

$$y(\beta,t) = \beta - \epsilon\frac{A_0 M_0}{C}\sin(kCt)\cos(k\beta), \tag{C.8b}$$

$$M(\beta,t) = M_0 - \epsilon A_0 kC\sin(kCt)\sin(k\beta). \tag{C.8c}$$

We see Geometrical Shock Dynamics predicts periodic, small amplitude perturbations of a shock front to oscillate without decay. This is in contrast to the solution of the Euler equations with the same initial condition. For the case of a first order perturbation introduced only in the immediate neighborhood of the shock front (the case most comparable to Geometrical Shock Dynamics, which assumes uniform flow behind the shock), Roberts [42] found that the amplitude of the front behaved as $t^{-3/2}$ times an oscillating function. The discrepancy between the solutions to Geometrical Shock Dynamics and the Euler equations exists regardless of the strength of the shocks involved. This behavior is an example of where the theory of Geometrical Shock Dynamics sometimes does not provide accurate predictions.

It should be noted that the above solution of the Geometrical Shock Dynamics equations, stating that plane shocks are only neutrally-stable, is only valid for small amplitude perturbations of the front. For perturbations of finite size, Geometrical Shock Dynamics predicts reduction of the amplitude and can actually be used to provide a quantitative description of the well-known result that plane shocks are super-stable structures [67].

Appendix D

Source code for Amrita Simulations

Euler simulations in this thesis were performed with version 1.31 of the new computational fluid dynamics software package Amrita . This version of Amrita contained an interpreter named *Amrita* and one computational module named *Amr_sol.* Our Amrita code consisted of one main script file and several subroutines written in the language Amrita (note the typographical convention use to distinguish between *Amrita* the interpreter, Amrita the language, and Amrita the complete software package) which were interpreted by *Amrita*, as well as a few Fortran subroutines which were compiled as shared object files and dynamically linked with the module *Amr_sol.* The files used in our simulations are given below. An example of a main script is given below*.

```
######################## Set Up Armita Environment #########################

autopath +lib
ArraySizes NGD=10000, NEX=20000, NGIxJ=600000

############################### Problem Parameters #########################

SetupProblem Ms=4, At=0.8, a=1, L=0.1, tend=1.25
SetupComputation levelset=yes, fluxlimiter=minmod, levels=2, total_dumps=20
SetupGeometry
SetupDiagnostics areacirculation=yes, levelset_interface=single-mode
module amr_sol
SetParameters

############################# Prepare for Run ###############################

postscript on
BuildGrid
FinalChecks
ICandBC shockIC=smear
set odir  = a$prob::amplitude/L$prob::thickness/At$prob::Atwood/Ms$prob::Mach
set odir .= /res$comp::unit_pts\x$comp::rI\x$comp::levels
solver code/$comp::solver
```

---

*This particular script was used to drive the simulation shown in figure 1.2.

```
PrintBeginMessage
#
if ("$comp::restart" eq "yes") then
  GetRestart odir=$odir, dump=$comp::start_dump
  time -> tout
else
  set tout = 0
endif
set comp::first_dump #= $comp::start_dump + 1
#
TakeSnapshots odir=ps/$odir, suffix=$comp::start_dump.ps

################################## Do the Run ##################################

do dump=$comp::first_dump,$comp::last_dump
  do subdump=1,$comp::subdumps
    set tout += $comp::tsubdump
    march to t=$tout with cfl=$comp::cfl
    RMdiagnostics odir=diagnostics/$odir
  end do
  TakeSnapshots odir=ps/$odir, suffix=$dump.ps
  flowout io/$odir/dump$dump
end do

############################## Finishing Tasks ###############################

PrintEndMessage
InformUser action=email, address=mrm@ama.caltech.edu
```

D.1 Amrita Subroutines

The following subroutines are written in the language Amrita and are interpreted by *Amrita* during the course of the simulation. For brevity we have chosen not to include non-essential subroutines such as `PrintBeginMessage` and `InformUser`. These routines can be written by the user to suit their own individual needs. Conversely, it is not necessary to include all the following Amrita routines to get a working code. Many of the subroutines are used to compute diagnostics or pointwise quantities of the flow. The user can pick and choose which subroutines are of interest in a specific simulation. For example, the simulations performed in chapter 4 did not include subroutines such as `BiotGrowthRate` and `PrintMaxVorticity`. The routines are now listed in no particular order.

```
#
# This procedure sets up parameters of the mathematical problem
#
proc SetupProblem {
    Ms [1:?]
    At [-1:1]
    a
    L [0:?]
    tend [0:?]
    profile [0:4]  = 0                  # 1-4 multimode and 0 is single mode
    gamma [1:?]    = 1.4
    pressure [1:?] = 1                  # ambient pressure
} -> prob::
#
  set prob::Mach          = $prob::Ms
  set prob::Atwood        = $prob::At
  set prob::amplitude     = $prob::a
  set prob::thickness     = $prob::L
#
end proc
```

```
#
# This procedure sets up specifics of the computation
#
proc SetupComputation {
    levelset {yes|no}                              = no
    fluxlimiter {minmod|superbee|vanleer}          = minmod
    restart {yes|no}                               = no
    total_dumps [0:?]                              = 5
    start_dump [0:?]                               = 0
    last_dump [1:?]                                = NULL
    unit_pts [50:?]                                = 100
    levels [0:?]                                   = NULL
    variable_type {dimensional|non-dimensional}    = dimensional
    cfl [0:1]                                      = 0.5
    subdumps [0:?]                                 = 10
    amr {on|off}                                   = on
    rI [0:?]                                       = 2
    ICsubfactor [1:100]                            = 10
} -> comp::
#
   if ("$levelset" eq "no") then
      set comp::solver = roe_fl-$fluxlimiter
      EulerEquations gamma=$prob::gamma
```

```
   else
      set comp::solver = roe_lvl_fl-$fluxlimiter
      EulerEquationsPassiveScalar gamma=$prob::gamma
   endif
#
   if ("$comp::last_dump" eq "NULL") then
      set comp::last_dump = $comp::total_dumps
   endif
#
end proc
```

```
#
proc CreateAuxiliaryEulerFunctions {
}
#
omega ::= (V[+i]-V[-i])/(X[+i]-X[-i]) - (U[+j]-U[-j])/(Y[+j]-Y[-j])
divu  ::= (U[+i]-U[-i])/(X[+i]-X[-i]) + (V[+j]-V[-j])/(Y[+j]-Y[-j])
#
end proc
```

```
#
# This procedure sets up characteristics of the grid which will be used to
#  construct it
#
proc SetupGeometry {
    grid_type {auto-grid|user-grid} = auto-grid
    domain_unit_lengths             = NULL
    Xdl                             = NULL
    Xs                              = NULL
    moving_grid {yes|no}            = no
    unit [0:?]                      = PI
} -> geom::
#
end proc
```

```
#
# This procedure determines what diagnostics will be computed in this run.
#  It also makes sure that diagnostics chosen are compatable with the
#  problem and computational configuration.
#
proc SetupDiagnostics {
    linecirculation {yes|no}                       = yes
    areacirculation {yes|no}                       = no
```

```
     density_interface {single-mode|multi-mode|no}  = single-mode
     levelset_interface {single-mode|multi-mode|no} = no
     mixing_width {single-mode|multi-mode|no}       = no
     divergence {yes|no}                            = no
} -> diag::
#
# First set diagnostic package to use depending on whether we have
#  levelset in our problem.  This is important because the diagnostic
#  routines need to know how many unknowns there are in the solution
#  vector (just Euler=4, Euler with levelset=5)
#
   if ("$comp::levelset" eq "no") then
      set diag::package = diagnostics
   else
      set diag::package = diagnostics-level
   end if
#
# First make sure levelset set up if they want diagnostics from it!
#
   if (("$comp::levelset" eq "no") && \
       ("$diag::areacirculation" eq "yes")) then
      DisplayMsg msg=Error in SetupDiagnostics
      error Cannot use areacirculation unless passive scalar is implemented
   end if
   if (("$comp::levelset" eq "no") && \
       ("$diag::levelset_interface" ne "no")) then
      DisplayMsg msg=Error in SetupDiagnostics
      error Must implement passive scalar!
   end if
#
# Now make sure that they are using multimode diagnostic routines if
#   interface is not single mode (i.e.:  profiles 1-4)
#
   if (("$diag::density_interface" eq "single-mode") && \
       ($prob::profile gt 0)) then
      DisplayMsg msg=Error in SetupDiagnostics
      error Cannot specify density_interface=$diag::density_interface \
            for multimode interfaces!
   end if
   if (("$diag::levelset_interface" eq "single-mode") && \
       ($prob::profile gt 0)) then
      DisplayMsg msg=Error in SetupDiagnostics
      error If interface is multimode!
   end if
   if (("$diag::mixing_width" eq "single-mode") && \
       ($prob::profile gt 0)) then
      DisplayMsg msg=Error in SetupDiagnostics
```

```
      error Cannot specify mixing_width=$diag::mixing_width for \
           multimode interfaces!
   end if
#
end proc
```

```
#
# This procedure is used to define all the parameters in a common block
#   that various subroutines of the Richtmyer-Meshkov run will need
#
proc SetParameters
    set variables = $comp::variable_type
#
# First copy some values from main program into local variables
#
    foreach parameter (Ms, At, gamma, pressure, L, tend)
       set longname   = prob::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (total_dumps, unit_pts, rI, levels, subdumps, amr)
       set longname   = comp::$parameter
       set $parameter = $$longname
    end foreach
    set unit = $geom::unit
#
# Define some parameters about space and time discretizations
#
    set tdump       #= $tend/$total_dumps
    set tsubdump    #= $tdump/$subdumps
    set lowest_res  #= $unit/$unit_pts
    set highest_res #= $lowest_res/$rI**$levels
    set halfcell    #= $highest_res/2
#
# Parameters of the one-dimensional problem
#
    set rho1        #= (1 - $At)/2       # preshock left density
    set rho2        #= (1 + $At)/2       # preshock right density
    export Ms,rho1,rho2,pressure,gamma
    if ($At < 0) then
      call ../code/oneD::sf1d
    else
      call ../code/oneD::fs1d
    endif
    import Ui,Ut,Ur,V,rhoL,rhoR          # rhoL,rhoR are postshocked
#
# Set up characterisitc quantities
```

```
#
    if ("$variables" eq "non-dimensional") then
      set rhoc        #= $rhoL + $rhoR
      set uc           = $V
      set pc          #= $rhoc*$V**2
      set lc          #= PI
      set tc          #= $lc/$uc
      set lowest_res  /= $lc
      set highest_res /= $lc
      set halfcell    /= $lc
      set rho1        /= $rhoc
      set rho2        /= $rhoc
      set rhoL        /= $rhoc
      set rhoR        /= $rhoc
      set Ui          /= $uc
      set Ut          /= $uc
      set Ur          /= $uc
      set V           /= $uc
      set pressure    /= $pc
    else
      set rhoc         = 1
      set uc           = 1
      set pc           = 1
      set lc           = 1
      set tc           = 1
    endif

#
# Send new quanities back to main program
#
    foreach parameter (rho1, rho2, rhoL, rhoR, Ui, Ut, Ur, V)
       set longname  = oneD::$parameter
       set $longname = $$parameter
    end foreach
    foreach parameter (rhoc, uc, pc, lc, tc)
       set longname  = scaling::$parameter
       set $longname = $$parameter
    end foreach
    foreach parameter (tdump, tsubdump, lowest_res, highest_res, halfcell)
       set longname  = comp::$parameter
       set $longname = $$parameter
    end foreach
    set prob::pressure = $pressure
#
end proc
```

```
#
# Build the computational grid
#
proc BuildGrid {
    space                          = two-dimensional
}
    set length = $geom::grid_type
#
# First copy some values from main program into local variables
#
    foreach parameter (At, a, L, tend)
       set longname   = prob::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (domain_unit_lengths, Xdl, Xs, unit, moving_grid)
       set longname   = geom::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (unit_pts, lowest_res)
       set longname   = comp::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (Ui, Ut, Ur, V)
       set longname   = oneD::$parameter
       set $parameter = $$longname
    end foreach
    set lc = $scaling::lc
    set a           /= $lc
    set L           /= $lc
    set unit        #= $unit/$lc         # if unit=pi we can't do /= shorthand
#
# Set up geometry of problem
#
    if ($At > 0) then
       set arg         #= 2e-6/$At - 1
    else
       set arg         #= -2e-6/$At - 1
    endif
    set arctanh_arg #= atanh($arg)
    set gap         #= -$L*($arctanh_arg)
#
    switch on $length
      case user-grid:
          set domain_length #= $domain_unit_lengths*$unit
          if ("$Xdl" ne "NULL") then
            if ($Xdl < $domain_length) then
              echo WAR (BuildGrid):  Both factor and Xdl are specified
```

```
              echo Hope this is okay:  domain_length=$domain_length, Xdl=$Xdl
            else
              error ERR (BuildGrid):  domain_length=$domain_length, Xdl=$Xdl
            endif
          else
            set Xdl #= $domain_length/2
          endif
          if ("$Xs" ne "NULL") then
            if ($Xs < $domain_length) then
              echo WAR (BuildGrid):  Both factor and Xs are specified
              echo Hope this is okay:  domain_length=$domain_length, Xs=$Xs
            else
              error ERR (BuildGrid):  domain_length=$domain_length, Xs=$Xs
            endif
          else
            set Xs  #= $Xdl - $a - $gap
            if ($Xs < 0) then
              echo ERR (BuildGrid):  Xs=$Xs < 0   (shock left of grid)
              error Either specify Xs or choose bigger domain_unit_lengths
            endif
          endif
#
      case auto-grid:
          if ("$moving_grid" eq "yes") then
             set Ui          -= $V
             set Ur          -= $V
             set Ut          -= $V
          endif
          set t_impact    #= $a/$Ui + $gap/(3*$Ui/4 + $Ut/4)
          set t_remain    #= $tend - $t_impact
          set x_trans     #= $t_remain*$Ut
          set x_reflect   #= $t_remain*$Ur
          if ($Ur > 0) then                      # refl shock moves forward
            set Xs        #= 10*$lowest_res
            set Xdl       #= $Xs + $gap + $a
            set x_total   #= $x_trans + $Xdl + $a
          else
            set x_reflect #= $t_remain*$Ur       # refl. shock moves backward
            set x_total   #= $x_trans - $x_reflect + $a
          endif
#
          set domain_unit_lengths     = 1
          set domain_length          #= $domain_unit_lengths*$unit
          while ($domain_length < $x_total)
             set domain_unit_lengths += 1
             set domain_length       += $unit
          end while
```

```
#
          if ($Ur < 0) then
            set extra        #= $domain_length - $x_total
            set Xdl          #= $domain_length - $extra/2 - $x_trans - $a
            set Xs           #= $Xdl - $a - $gap
          endif
#
          if ($Xs < 0) then
            set more_factor   = 1
            set more_length   = $unit
            while ($more_length < abs($Xs))
              set more_factor += 1
              set more_length += $unit
            end while
            set Xs            += $more_length
            set Xdl           += $more_length
          endif
#
      default:
          error ERR (BuildGrid):  length set to $length!
    end switch
#
# Build the grid
#
    set patches #= $domain_unit_lengths*$unit_pts/25
    switch on $space
      case two-dimensional:
        set remain  #= $unit_pts % 4
        if ("$remain" eq "0") then # four patch for height
          set qtr_pts #= $unit_pts/4
          set hr1     #= $qtr_pts + 1
          set hr2     #= $hr1 + $qtr_pts
          set hr3     #= $hr2 + $qtr_pts
          def Domain
             lscale $lowest_res
             do col=1,$patches
               set i #= ($col-1)*25+1
               patch <$i,   1,w25,h$qtr_pts>
               patch <$i,$hr1,w25,h$qtr_pts>
               patch <$i,$hr2,w25,h$qtr_pts>
               patch <$i,$hr3,w25,h$qtr_pts>
             end do
          end def
        else
          set half_pts #= $unit_pts/2      # two patch for height
          set hr1     #= $half_pts + 1
          def Domain
```

```
              lscale $lowest_res
              do col=1,$patches
                set i #= ($col-1)*25+1
                patch <$i,   1,w25,h$half_pts>
                patch <$i,$hr1,w25,h$half_pts>
              end do
           end def
         endif
       case one-dimensional:
         def Domain
            lscale $lowest_res
            do col=1,$patches
              set i #= ($col-1)*25+1
              patch <$i,1,w25,h2>
            end do
         end def
       default:
         error ERR (BuildGrid):  space of $space not understood in def Domain
    end switch
#
# Send new quanities back to main program
#
    foreach parameter (Xs, Xdl)
       set longname   = geom::$parameter
       set $longname  = $$parameter
    end foreach
    foreach parameter (a, L)
       set longname   = prob::$parameter
       set $longname  = $$parameter
    end foreach
    set geom::unit    = $unit
#
end proc
```

```
#
# Set up ICs for Richtmyer-Meshkov
#
proc ICandBC {
    shockIC {sharp|smear} = sharp
}
#
# First copy some values from main program into local variables
#
    foreach parameter (Ms, At, a, L, pressure, gamma, profile)
       set longname   = prob::$parameter
       set $parameter = $$longname
```

```
    end foreach
    foreach parameter (cfl, solver, highest_res, amr, levels, rI, ICsubfactor)
       set longname   = comp::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (rho1, rho2, V)
       set longname   = oneD::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (lc, rhoc)
       set longname   = scaling::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (Xdl, Xs, moving_grid)
       set longname   = geom::$parameter
       set $parameter = $$longname
    end foreach
#
# Write down initial conditions of interior flow
#
    if ("$moving_grid" eq "no") then
      set flow = 0
    else
      set flow = -$V
    endif
    if ("$comp::levelset" eq "yes") then
       W'quiescent ::= <RHO=$rho1,U=$flow,V=0,P=$pressure,PSI=1>
    else
       W'quiescent ::= <RHO=$rho1,U=$flow,V=0,P=$pressure>
    endif
#
    DEN          ::= fn(../code/ic::pham_rho,x[],x[+i],y[],y[+j], \
                       $ICsubfactor,$Xdl,$At,$a,$L,$lc,$profile) / $rhoc
    if ("$comp::levelset" eq "yes") then
       scalarfcn ::= 2*DEN[]/($rho1-$rho2) + 1 - 2*$rho1/($rho1-$rho2)
       W'RMfield   ::= <RHO=DEN[],U=$flow,V=0,P=$pressure,PSI=scalarfcn[]>
    else
       W'RMfield  ::= <RHO=DEN[],U=$flow,V=0,P=$pressure>
    endif
    switch on $shockIC
       case sharp:
          ShockWave Ms=$Ms, state1=quiescent, state2=post_shock
          def SolutionField
            setfield W'RMfield    X[]>=$Xs
            setfield W'post_shock X[]<$Xs
          end def
       case smear:
```

```
          set widthcells = 10               # smear shock over this many cells
          if ("$comp::levelset" eq "yes") then
             set routine = SmearedShockPassiveScalar
          else
             set routine = SmearedShock
          endif
          $routine Ms=$Ms, Xs=$Xs, state1=quiescent, \
            state2=smeared_shock, cfl=$cfl, solver=$solver, \
            dxmin=$highest_res, width=$widthcells
          def SolutionField
             setfield W’RMfield       X[]>($Xs+$widthcells*$highest_res/2)
             setfield W’smeared_shock X[]<=($Xs+$widthcells*$highest_res/2)
          end def
       default:
    end switch
#
# Set up Boundary Conditions
#
    def BoundaryConditions
      Nbdy domain: reflect
      Ebdy domain: extrapolate
      Wbdy domain: extrapolate
      Sbdy domain: reflect
   end def
#
# Set up amr and write initial conditions into the grids
#
    if ("$Ms" le "1.5") then
      set tol     #= $rho1 < $rho2 ? 0.0025*$rho1 : 0.0025*$rho2
    else
      set tol     #= $rho1 < $rho2 ? 0.1*$rho1 : 0.1*$rho2
    endif
#
   def MeshAdaption
      adaption  $amr
      lmax      $levels
      rI        $rI
   end def
   def RefinementCriteria
      DensityGradient tolerance=$tol
   end def
#
   do refine=1,$levels
     makefield
     adapt
   end do
   makefield
```

```
   autoscale
#
   if ("$shockIC" eq "smear") then
     execute 'rm' smear.*
     freeinterp
   end if
end proc
```

```
#
# Produce a smeared shock profile that will propagate without startup error
# When finished, W'$state2 contains the new profile
#
proc SmearedShock {
   Ms
   Xs                             # where smeared shock profile will be centered
   state1 = quiescent             # preshock state
   state2 = smeared_shock         # on output will contain new smeared profile
   cfl
   solver
   dxmin                          # smallest dx in domain
   width                          # smear over this many cells
}
   ShockWave Ms=$Ms, state1=$state1, state2=$state2
   set gamma  #= sym(GAMMA'$state1)
   set rho1   #= sym(RHO'$state1)
   set rho2   #= sym(RHO'$state2)
   set p1     #= sym(P'$state1)
   set p2     #= sym(P'$state2)
   set u1     #= sym(U'$state1)
   set u2     #= sym(U'$state2)
#
   splurge to compute_smear_shock

############################ compute_smear_shock #########################
set shock_radius #= $width/2
set a01     #= sqrt($gamma*$p1/$rho1)
set Us      #= $Ms*\$a01 + $u1
set u2      #= $u1 + \$a01*2*($Ms*$Ms-1)/($Ms*($gamma+1))
set a2      #= \$a01*sqrt( 2*$gamma*$Ms*$Ms-($gamma-1) )*sqrt( \\
                ($gamma-1)*$Ms*$Ms+2 ) / ($Ms*($gamma+1))
#
set gamma = $gamma
EulerEquations space=two-dimensional, gamma=$gamma
module amr_sol
#
set Cminus #= \$u2 - \$a2                    # speed of C- characteristic
```

```
set udiff  #= \$Us - \$u2                  # speed diff between shock and C-
set tf     #= 50/\$udiff                  # timestep until C- startup error is
                                          #   50 units upstream
if (\$Cminus < 0) then
  set Xs   #= -(\$Cminus)*\$tf + 20       # start shock far enough inside domain
else                                    #  that errors won't go off left edge
  set Xs    = 15
endif
set Xsf    #= \$Us*\$tf + \$Xs        # final location of shock
set Xtotal #= \$Xsf + 3*\$shock_radius    # we don't want shock to hit the end
                                          #  so we add a bit extra space
#
set patches    = 1                      # define a big enough grid
set length    #= \$patches*25
while (\$length < \$Xtotal)
  set patches #= \$patches+1
  set length  #= \$patches*25
end while
def Domain
   lscale  1
   do col=1,\$patches
     set i #= (\$col-1)*25+1
     patch <\$i,1,w25,h1>
   end do
end def
#
W'quiescent ::= <RHO=$rho1,U=$u1,V=0,P=$p1>
ShockWave Ms=$Ms, state1=quiescent, state2=post_shock
def SolutionField
   setfield W'quiescent
   setfield W'post_shock X[]<\$Xs
end def
#
def BoundaryConditions
   Nbdy domain: reflect
   Ebdy domain: extrapolate
   Wbdy domain: prescribe    W'post_shock
   Sbdy domain: reflect
end def
#
makefield
solver code/$solver
march to t=\$tf with cfl=$cfl
#
set XsL #= \$Xsf - \$shock_radius
set XsR #= \$Xsf + \$shock_radius
#
```

```
den ::= X[]<\$XsL ? (RHO'post_shock) : ( X[]>\$XsR ? (RHO'quiescent) : RHO[] )
vex ::= X[]<\$XsL ?   (U'post_shock) : ( X[]>\$XsR ?   (U'quiescent) :   U[] )
pre ::= X[]<\$XsL ?   (P'post_shock) : ( X[]>\$XsR ?   (P'quiescent) :   P[] )
#
foreach field (den, vex, pre)
printfile smear.\$field
along y=0.5 print X[]-\$Xsf,\$field[]
end foreach
printfile
############################ compute_smear_shock ##########################

   splurge off
   parse token compute_smear_shock &
   smearrho   ::= interp(p=0, smear.den,(X[]-$Xs)/$dxmin)
   smearu     ::= interp(p=0, smear.vex,(X[]-$Xs)/$dxmin)
   smearp     ::= interp(p=0, smear.pre,(X[]-$Xs)/$dxmin)
   W'$state2  ::= <RHO=smearrho[],U=smearu[],V=0,P=smearp[]>
#
end proc
```

```
#
# Because this is a restarted run, we need to do a few extra things
#
proc GetRestart {
    odir
    dump
}
#
# Make sure we don't wipe out diagnostics from previous dumps
#
    foreach data (circ, amp, dadt, delta, divu)
      set file = diagnostics/$odir/$data\.dat
      if (-e "$file") then
        execute 'cp' $file $file\.$dump
      endif
    end foreach
#
    flowin io/$odir/dump$dump
#
end proc
```

```
#
# Procedure to output important diagnostics about the flow to files based
#  on options specified by the user in main script
```

```
#
proc RMdiagnostics {
    odir                                         = ./diagnostics
}
#
# First copy some values from main program into local variables
#
    foreach parameter (density_interface, levelset_interface, mixing_width, divergenc
       set longname   = diag::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (rhoL, rhoR, rho1, rho2)
       set longname   = oneD::$parameter
       set $parameter = $$longname
    end foreach
    foreach parameter (levelset, halfcell, highest_res)
       set longname   = comp::$parameter
       set $parameter = $$longname
    end foreach
#
    set rho_avg   #= ($rhoL+$rhoR)/2
    extent -> xo,yo,width,height
    set bot #= $yo + $halfcell
    set top #= $yo + $height - $halfcell
    set lft #= $xo + $halfcell
    set rgt #= $xo + $width - $halfcell
#
# Go to appropriate routine to compute needed diagnostics
#
    time -> time
    Circulation odir=$odir, time=$time, ytop=$top, ybot=$bot, \
        xlft=$lft, xrgt=$rgt
#
    if ("$density_interface" ne "no") InterfaceDiagnostics odir=$odir, \
       time=$time, rho_avg=$rho_avg, \
       ybot=$bot, ytop=$top, highest_res=$highest_res
#
    if ("$levelset_interface" ne "no") InterfaceDiagnosticsLevelset \
       odir=$odir, time=$time, ybot=$bot, ytop=$top, highest_res=$highest_res
#
    if ("$divergence" eq "yes") CompressibilityDiagnostic odir=$odir, time=$time
#
    if ("$mixing_width" ne "no") then
       MixingWidth odir=$odir, time=$time, interface_type=$mixing_width, \
         ytop=$top, ybot=$bot, xlft=$lft, xrgt=$rgt, rhoL=$rhoL, rhoR=$rhoR, \
         rho1=$rho1, rho2=$rho2, highest_res=$highest_res
    endif
```

```
#
end proc
```

```
#
# Procedure will compute circulation in a region.
#
proc Circulation {
    odir
    time
    lineintegral = $diag::linecirculation
    areaintegral = $diag::areacirculation
    ytop
    ybot
    xlft
    xrgt
}
#
# Line integral over entire computational domain
#
   if ("$lineintegral" eq "yes") then
      foreach side (top, bot)
         set linetype   = y                        # {x|y}
         set coordinate = $$y'side
         set Lbound     = $xlft
         set Hbound     = $xrgt
         set normalize  = n                        # {y|n}
         set field      = u                        # {rho|u|v|mx|my}
         export linetype, coordinate, Lbound, Hbound, normalize, field
         call ../code/$diag::package\::lineintegral
         import integral
         set $side      = $integral
      end foreach
      foreach side (lft, rgt)
         set linetype   = x                        # {x|y}
         set coordinate = $$x'side
         set Lbound     = $ybot
         set Hbound     = $ytop
         set normalize  = n                        # {y|n}
         set field      = v                        # {rho|u|v|mx|my}
         export linetype, coordinate, Lbound, Hbound, normalize, field
         call ../code/$diag::package\::lineintegral
         import integral
         set $side      = $integral
      end foreach
      set circ #= $bot + $rgt - $top - $lft
#
```

```
# Output circulation to datafile
#
      printfile >> $odir/circ.dat
      set time #= sprintf("%12.7g",$time)
      set circ #= sprintf("%12.7g",$circ)
      splurge on (col=1)
         $time $circ
      splurge off
      printfile
   endif                             # end of lineintegral
#
# Area integral just over density layer
#
   if ("$areaintegral" eq "yes") then

# This is a fix for a bug in Amrita version 1.31.  You need to do an
#  grid operation to get the flag_refined_cells call to work right.

           plotfile junk.ps
           plot grids
           plotfile
           execute 'rm' junk.ps

# end of bug fix

      set parameter = 0.99           # the layer is |psi| < this
      export parameter
      call ../code/$diag::package\::area_circulation
      import circulation
      printfile >> $odir/areacirc.dat
      set circulation #= sprintf("%12.7g",$circulation)
      splurge on (col=1)
         $time $circulation
      splurge off
   endif
#
# All's well that ends well
#
end proc
```

```
#
# Procedure handles output of amplitude and growth rate of interface for
#  either single-mode or multi-mode interfaces.  The interface is defined
#  as the zero level set of the passive scalar function psi.
#
proc InterfaceDiagnosticsLevelset {
```

```
     odir
     time
     levelset_interface = $diag::levelset_interface
     a                  = $prob::a
     ybot
     ytop
     highest_res
}
     set time #= sprintf("%12.7g",$time)
     set interface_type = $levelset_interface
     export ybot,ytop,a,highest_res,interface_type
     call ../code/$diag::package\::interface_by_levelset
     import amp,dadt
     printfile >> $odir/psiamp.dat
     set amp  #= sprintf("%12.7g",$amp)
     splurge on (col=1)
        $time $amp
     splurge off
     printfile >> $odir/psidadt.dat
     set dadt #= sprintf("%12.7g",$dadt)
     splurge on (col=1)
        $time $dadt
     splurge off
     printfile
#
end proc
```

```
# compute the average of the divergence along the interface (psi=0)

proc Divergence {
     odir
}

   set field = divu
   export field
   call ../code/avg_along_interface::avg_along_interface
   import average
   time -> time
   printfile >>$odir/divu.dat
   set time #= sprintf("%12.7g",$time)
   set avg  #= sprintf("%12.7g",$average)
   splurge on (col=1)
      $time $avg
   splurge off
   printfile
```

```
end proc
```

```
# Calculates the instaneous growth rate using velocities computed from
#  Biot-Savart point-vortex approximation

proc BiotGrowthRate {
    odir
    region {entire|layer} = entire
}

   extent -> xo,yo,width,height
   set top #= $yo + $height - 0.00001
   along y=$top maximum X[],Y[]    -> xtemp,ytop
   along y=$yo  maximum X[],1/Y[] -> xtemp,yinv
   set ybot #= 1/$yinv
   export ybot,ytop

# This is a fix for a bug in Amrita version 1.31.  You need to do a
#  grid operation to get the flag_refined_cells call to work right.

           plotfile junk.ps
           plot grids
           plotfile
           execute 'rm' junk.ps

# end of bug fix

   call ../code/biot_$region\::biot_savart
   import xbot,xtop,ubot,utop,ubot_biot,vbot_biot,utop_biot,vtop_biot
   set dadt #= ($ubot-($utop))/2
   set dadt_biot #= ($ubot_biot-($utop_biot))/2

   printfile >>$odir/biot.$region\.dat
   time -> time
   set time #= sprintf("%12.7g",$time)
   set dadt_biot #= sprintf("%12.7g",$dadt_biot)
   splurge on (col=1)
      $time $dadt_biot
   splurge off
   printfile

end proc
```

```
#
```

```
# Subroutine acts as driver used for fortran routine that prints out
#  the maximum vorticity in the layer y-value by y-value.
#
proc PrintMaxVorticity {
    odir = layer
    file = maxvort.dat
    code  {fortran|amrita} = fortran
}
#
# Locate which cell on the bottom contains the curve
#
    extent -> xo,ybot,width,height
    set ytop #= $ybot + $height - 1e-6
    along y=$ytop maximum X[],Y[] -> x,ytop
    along y=$ybot maximum X[],1/Y[] -> x,yinv
    set ybot #= 1/$yinv
#
    if ("$code" eq "amrita") then
       error (PrintMaxVorticity) No logic for code=amrita !
    else
       chan (23,f,w) $odir/$file
       export ybot,ytop
       call ../code/max_vorticity::max_vorticity_curve
    endif
#
end proc
```

D.2 Fortran Subroutines

Amrita acts as a programmable interface between the user and the computational module *Amr_sol*. While the language Amrita contains syntax for performing some mathematical calculations, Amrita scripts and subroutines are interpreted rather than compiled so they are executed relatively slowly. Subroutines that perform non-trivial calculations are best coded as Fortran or C subroutines which are compiled separately and then dynamically linked with the module *Amr_sol*. The most important of these 'plug-ins' is the flow solver. We do not list our solver here since it differs from the standard flux-limited Roe solver included in the standard *Amrita* package only slightly. In particular, a fifth field must be added to accommodate the passive scalar function $\psi(x, y)$ introduced in §3.2 and a choice between the minmod, van Leer, and Superbee flux limiters was allowed. Listings of the Fortran routines follow, in no particular order. Subroutines are bundled in 'packages.' These packages are each compiled separately to produce one shared object file per package. These packages are allowed to call program units located in the *Amr_sol* module object file, such as subroutine `FLAG_REFINED_CELLS` and function `Xcg`. As in §D.1, not all the following subroutines are required to perform an Euler simulation.

Package `oneD`

```
      subroutine fs1d
C This program computes the solution to the 1D f/s shock-contact problem
C using bisection
C Solution computed by using shock jump conditions
C
C Notation used is the following:
C Before impact:
C                  ||               ||
C      (1)         ||-->   (L)      ||     (R)
C                  ||               ||
C              incident shk.     contact
C
C After impact:
C             ||                ||             ||
C      (1)    ||->    (2)       ||->   (3)     ||-->   (R)
C             ||                ||             ||
C        refl. shock        contact         trans. shock
C
      IMPLICIT NONE
      AMRDBL Mi,Mt,Mr
      AMRDBL Ui,Ut,Ur
      AMRDBL p0,p1,p23
      AMRDBL u0,u1,u23
      AMRDBL r0L,r0R,r1,r2,r3
      AMRDBL a0L,a0R,a1,a2,a3
      AMRDBL gamma
      AMRDBL Pressure,Velocity,Density,SoundSpeed

C Get data
      u0 = 0.0d0
      CALL AMR_GET_TOKEN('DBL::Ms',Mi)
      CALL AMR_GET_TOKEN('DBL::rho1',r0L)
      CALL AMR_GET_TOKEN('DBL::rho2',r0R)
      CALL AMR_GET_TOKEN('DBL::pressure',p0)
      CALL AMR_GET_TOKEN('DBL::gamma',gamma)

      a0L = dsqrt(gamma*p0/r0L)
```

```
      a0R = dsqrt(gamma*p0/r0R)
      a1 = SoundSpeed(a0L,Mi,gamma)
      call FSgetMachs (Mi,Mt,Mr,gamma,a0R,a0L,a1)

      p1 = Pressure(p0,Mi,gamma)
      p23 = Pressure(p0,Mt,gamma)
      u1 = Velocity(u0,a0L,Mi,gamma)
      u23 = Velocity(u0,a0R,Mt,gamma)
      r1 = Density(r0L,Mi,gamma)
      r2 = Density(r1,Mr,gamma)
      r3 = Density(r0R,Mt,gamma)
      a2 = SoundSpeed(a1,Mr,gamma)
      a3 = SoundSpeed(a0R,Mt,gamma)
      Ui = Mi*a0L
      Ut = Mt*a0R
      Ur = u1 - Mr*a1

      CALL AMR_SET_TOKEN('DBL::Ui',Ui)
      CALL AMR_SET_TOKEN('DBL::Ut',Ut)
      CALL AMR_SET_TOKEN('DBL::Ur',Ur)
      CALL AMR_SET_TOKEN('DBL::V',u23)
      CALL AMR_SET_TOKEN('DBL::rhoL',r2)
      CALL AMR_SET_TOKEN('DBL::rhoR',r3)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine FSgetMachs (Mi,Mt,Mr,gamma,a0R,a0L,a1)
C This subroutine uses bisection method to find Mr.  Mt is also computed
      IMPLICIT NONE
      AMRDBL Mi,Mt,Mr,gamma,a0R,a0L,a1
      AMRDBL Mleft,Mright,Lresidual,Rresidual,res
      AMRDBL FSMachTransmitted,FSresidual,tolerance

      tolerance = 1.0d-7
      Mleft = 1.0d0
      Mt = FSMachTransmitted(Mleft,Mi,gamma)
      Lresidual = FSresidual(Mi,Mt,Mleft,gamma,a0R,a0L,a1)
      Mright = 2.0d0*Mi
      Mt = FSMachTransmitted(Mright,Mi,gamma)
      Rresidual = FSresidual(Mi,Mt,Mright,gamma,a0R,a0L,a1)

      do 10 while (dabs(Mleft-Mright).ge.tolerance)
        Mr = (Mleft+Mright)/2.0d0
```

```
           Mt = FSMachTransmitted(Mr,Mi,gamma)
           res = FSresidual(Mi,Mt,Mr,gamma,a0R,a0L,a1)
           if (res*Lresidual .lt. 0.0d0) then
             Mright = Mr
             Rresidual = res
           else
             Mleft = Mr
             Lresidual = res
           ENDif
 10      continue

         Mr = (Mleft+Mright)/2.0d0
         Mt = FSMachTransmitted(Mr,Mi,gamma)

         RETURN
         END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

         AMRDBL function FSMachTransmitted(Mr,Mi,gamma)
         IMPLICIT NONE
         AMRDBL Mr,Mi,gamma
         AMRDBL temp

         temp = ( 1.0d0+2.0d0*gamma*(Mr*Mr-1.0d0)/(gamma+1.0d0) ) *
     &    ( 1.0d0+2.0d0*gamma*(Mi*Mi-1.0d0)/(gamma+1.0d0) )
         FSMachTransmitted = dsqrt( 1.0d0 + (gamma+1.0d0)*(temp-1.0d0) /
     &    (2.0d0*gamma) )

         RETURN
         END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

         AMRDBL function FSresidual (Mi,Mt,Mr,gamma,a0R,a0L,a1)
         IMPLICIT NONE
         AMRDBL Mi,Mt,Mr,gamma,a0R,a0L,a1

         FSresidual = (Mi*Mi-1.0d0)*a0L/Mi - (Mr*Mr-1.0d0)*a1/Mr -
     &    (Mt*Mt-1.0d0)*a0R/Mt

         RETURN
         END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE SF1D
C This program computes the solution to the 1D s/f shock-contact problem
C using bisection
C Solution computed just like shocktube problem
C
C Notation used is the following:
C Before impact:
C                  ||               ||
C      (1)         ||-->   (L)      ||     (R)
C                  ||               ||
C              incident shk.     contact
C
C After impact:
C            |         |            ||         ||
C      (1)   |-> (2)   |->   (3)    ||->  (4)  ||-->   (R)
C            |         |            ||         ||
C            rarefaction          contact   trans.shock
C           LE        TE
C
      IMPLICIT NONE
      AMRDBL Mi,Mt
      AMRDBL Ui,Ut,UrLE,UrTE
      AMRDBL p0,p1,p34
      AMRDBL u0,u1,u34
      AMRDBL r0L,r0R,r1,r3,r4
      AMRDBL a0L,a0R,a1,a3,a4
      AMRDBL gamma
      AMRDBL Pressure,Velocity,Density,SoundSpeed
C Get data
      u0 = 0.0d0
      CALL AMR_GET_TOKEN('DBL::Ms',Mi)
      CALL AMR_GET_TOKEN('DBL::rho1',r0L)
      CALL AMR_GET_TOKEN('DBL::rho2',r0R)
      CALL AMR_GET_TOKEN('DBL::pressure',p0)
      CALL AMR_GET_TOKEN('DBL::gamma',gamma)
C
      a0L = sqrt(gamma*p0/r0L)
      a0R = sqrt(gamma*p0/r0R)
      a1  = SoundSpeed(a0L,Mi,gamma)
      u1  = Velocity(u0,a0L,Mi,gamma)
      p1  = Pressure(p0,Mi,gamma)
      r1  = Density(r0L,Mi,gamma)
      call getShockStrength(p34,u34,gamma,r1,u1,a1,p1,p0,a0R)
      Mt  = sqrt(1.0d0 + 0.5d0*(gamma+1.0d0)*(p34-p0)/(p0*gamma))
```

```
      r4 = Density(r0R,Mt,gamma)
      a4 = sqrt(gamma*p34/r4)
      r3 = r1*(p34/p1)**(1.0d0/gamma)
      a3 = sqrt(gamma*p34/r3)

      Ui   = Mi*a0L
      Ut   = Mt*a0R
      UrLE = u1 - a1
      UrTE = u34 - a3
C
      CALL AMR_SET_TOKEN('DBL::Ui',Ui)
      CALL AMR_SET_TOKEN('DBL::Ut',Ut)
      CALL AMR_SET_TOKEN('DBL::Ur',UrLE)
      CALL AMR_SET_TOKEN('DBL::V',u34)
      CALL AMR_SET_TOKEN('DBL::rhoL',r3)
      CALL AMR_SET_TOKEN('DBL::rhoR',r4)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine getShockStrength (p34,u34,gamma,r1,u1,a1,p1,p0,a0R)
C This subroutine uses bisection to find p34 and u34
      IMPLICIT NONE
      AMRDBL p34,u34,gamma,r1,u1,a1,p1,p0,a0R
      AMRDBL pL,VL,Lresidual
      AMRDBL pR,VR,Rresidual
      AMRDBL tolerance,SFresidual,sfV,res

      tolerance = 1.0d-7
      pL        = p0
      VL        = sfV(pL,u1,gamma,r1,p1,a1)
      Lresidual = SFresidual(VL,pL,p0,a0R,gamma)
      pR        = p1
      VR        = sfV(PR,u1,gamma,r1,p1,a1)
      Rresidual = SFresidual(VR,pR,p0,a0R,gamma)
C
      do 10 while (abs(pL-pR).ge.tolerance)
         p34 = 0.5d0*(pL+pR)
         u34 = sfV(p34,u1,gamma,r1,p1,a1)
         res = SFresidual(u34,p34,p0,a0R,gamma)
         if (res*Lresidual .lt. 0.0d0) then
            pR        = p34
            Rresidual = res
         else
```

```
            pL        = p34
            Lresidual = res
          ENDif
 10   continue
      p34 = (pL+pR)/2.0d0
      u34 = sfV(p34,u1,gamma,r1,p1,a1)
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function sfV(p,u1,gamma,r1,p1,a1)
      IMPLICIT NONE
      AMRDBL p,u1,gamma,r1,p1,a1
      AMRDBL temp

      temp = - 2.0d0*sqrt( gamma*p**(1.0d0-1.0d0/gamma)
     &     *p1**(1.0d0/gamma)/r1 ) + 2.0d0*a1
      sfV  = u1 + temp / (gamma-1.0d0)
C
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function SFresidual (V,p,p0,a0R,gamma)
      IMPLICIT NONE
      AMRDBL V,p,p0,a0R,gamma
      AMRDBL Z

      Z          = p/p0 - 1.0d0
      SFresidual = Z/( gamma*sqrt(1.0d0+0.5d0*(gamma+1.0d0)*Z/gamma) )
     &         - V/a0R
C
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function Pressure (p,M,gamma)
      IMPLICIT NONE
      AMRDBL p,M,gamma

      Pressure = p*( 1.0d0 + 2.0d0*gamma*(M*M-1.0d0)/(gamma+1.0d0) )
```

```
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function Velocity (u,a,M,gamma)
      IMPLICIT NONE
      AMRDBL u,a,M,gamma

      Velocity = u + a*2.0d0*(M*M-1.0d0)/((gamma+1.0d0)*M)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function Density (rho,M,gamma)
      IMPLICIT NONE
      AMRDBL rho,M,gamma

      Density = rho*(gamma+1.0d0)*M*M/((gamma-1.0d0)*M*M+2.0d0)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL function SoundSpeed (a,M,gamma)
      IMPLICIT NONE
      AMRDBL a,M,gamma
      AMRDBL temp

      temp = (2.0d0*gamma*M*M-(gamma-1.0d0))*((gamma-1.0d0)*M*M+2.0d0)
      SoundSpeed = dsqrt(temp)*a/((gamma+1.0d0)*M)

      RETURN
      END
```

Package ic

```
      AMRDBL FUNCTION PHAM_RHO(W)
      IMPLICIT NONE
C This fortran function is used to construct the initial density field
```

```
C  for the Richtmyer-Meshkov instability in a layer with continous density
C  gradient.  The profiles this subroutine generates are those used by Pham
C  in his CIT Applied Math thesis.
      AMRDBL W(0:*)
      AMRDBL X0,X1,Y0,Y1,SUBFACTOR,XDL,AT,AMP,L,LC,PROFILE
      AMRDBL RHO1,RHO2,XOFFSET,YOFFSET,DENSITY,XSUB,YSUB,INTERFACE
      AMRDBL BASEPROFILE
      AMRINT I,J,FACTOR,IPROFILE

      X0         = W(1)
      X1         = W(2)
      Y0         = W(3)
      Y1         = W(4)
      SUBFACTOR  = W(5)
      XDL        = W(6)
      AT         = W(7)
      AMP        = W(8)
      L          = W(9)
      LC         = W(10)
      PROFILE    = W(11)

      RHO1     = (1.0D0 - AT)/2.0D0
      RHO2     = (1.0D0 + AT)/2.0D0
      XOFFSET  = (X1-X0)/SUBFACTOR
      YOFFSET  = (Y1-Y0)/SUBFACTOR
      FACTOR   = INT(SUBFACTOR)
      IPROFILE = INT(PROFILE)

      DENSITY  = 0.0 D0
      DO J=1,FACTOR
         YSUB      = Y0 + DBLE(2*J-1)*YOFFSET/2.0D0
         YSUB      = YSUB*LC                   /* Need dimensional ysub in
                                                  trig functions */
         INTERFACE = XDL + AMP*BASEPROFILE(IPROFILE,YSUB)
         DO I=1,FACTOR
            XSUB  = X0 + DBLE(2*I-1)*XOFFSET/2.0D0
            IF (L .EQ. 0.0D0) THEN
               IF (XSUB.LT.INTERFACE) THEN
                  DENSITY = DENSITY + RHO1
               ELSE IF (XSUB.GT.INTERFACE) THEN
                  DENSITY = DENSITY + RHO2
               ELSE
                  DENSITY = DENSITY + 0.5 D0*(RHO1+RHO2)
               ENDIF
            ELSE
               DENSITY = DENSITY + 0.5 D0*( 1.0 D0 + AT*TANH(
   X              (XSUB-INTERFACE)/L ) )
```

```
            ENDIF
         END DO
      END DO
      DENSITY = DENSITY / (SUBFACTOR)**2
      PHAM_RHO = DENSITY
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      AMRDBL FUNCTION BASEPROFILE (IPROFILE,Y)
      IMPLICIT NONE
C This function actually contains the data for the multimode profiles
C  used by Pham
      AMRDBL ZETA,Y,R1(60,4),TEMP,KDBL
      PARAMETER(ZETA=1.0D-2)
      AMRINT IPROFILE,K,PROFILE
C
      IF (IPROFILE.EQ.0) THEN
         BASEPROFILE = COS(Y)
      ELSE
         DATA ((R1(K,PROFILE),PROFILE=1,4), K=1,60) /
     &   0.2751921,0.3298150,0.9529670,0.6640730,0.8888904,0.7173490,
     &   0.1626060,0.9854710,0.6348671,0.8679210,0.6224670,0.8386250,
     &   0.4398165,0.8192670,0.6426410,0.3251820,0.0007059,0.8600640,
     &   0.3859950,0.3385220,0.1755233,0.6563750,0.7847900,0.6320470,
     &   0.9761031,0.2583430,0.9588820,0.4574180,0.5681258,0.3082260,
     &   0.3315980,0.1176570,0.2711180,0.6972540,0.6831200,0.7242860,
     &   0.5584524,0.5975760,0.2575900,0.9374660,0.0573481,0.6791700,
     &   0.5677850,0.7264860,0.5488701,0.6054280,0.9451590,0.5130380,
     &   0.9679297,0.9568540,0.0394958,0.3754490,0.4032842,0.2474830,
     &   0.1942990,0.4602190,0.0975372,0.5564400,0.7869740,0.3619470,
     &   0.4471384,0.3467640,0.6032460,0.1429350,0.0023837,0.2345760,
     &   0.8757100,0.8710980,0.4674150,0.4922850,0.0836847,0.3497710,
     &   0.7002206,0.0352285,0.1166330,0.5868370,0.7126347,0.5981560,
     &   0.1046270,0.8136450,0.6703379,0.6970640,0.2422350,0.0863009,
     &   0.8928567,0.0868583,0.6474060,0.4931420,0.8535640,0.1167580,
     &   0.1775080,0.6293650,0.1796786,0.4968310,0.9111100,0.8493050,
     &   0.6522657,0.3333960,0.9809520,0.4097850,0.2062365,0.1098270,
     &   0.4699440,0.0956409,0.9303484,0.8252460,0.9024920,0.8438360,
     &   0.0672049,0.0583677,0.8460980,0.5748300,0.0132558,0.7709130,
     &   0.6588570,0.1527100,0.3187970,0.1898910,0.4803390,0.1316690,
     &   0.6879703,0.8672080,0.0156801,0.5854020,0.9787641,0.5904350,
     &   0.6873580,0.8615760,0.2030125,0.8597640,0.7575580,0.9781390,
     &   0.5263962,0.6907290,0.4087590,0.5421870,0.2684416,0.9638210,
     &   0.3229550,0.3386810,0.9025217,0.2730060,0.9637420,0.3884610,
```

```
     &   0.0558552,0.2479040,0.2458840,0.8707220,0.5095074,0.3096030,
     &   0.5987440,0.4023370,0.1983894,0.6606930,0.5912930,0.5534480,
     &   0.2112587,0.1071700,0.5788170,0.8501400,0.7907189,0.5666290,
     &   0.2982190,0.6232370,0.3332194,0.8906300,0.8560400,0.1824640,
     &   0.3890564,0.7383150,0.2092770,0.5732810,0.6623718,0.5111380,
     &   0.4455750,0.4359340,0.0111537,0.6301070,0.7486700,0.1972440,
     &   0.4472364,0.3847200,0.6011690,0.0872308,0.1363005,0.3698950,
     &   0.5457880,0.2972370,0.3978726,0.0359175,0.8713210,0.4610430,
     &   0.7053255,0.9693060,0.9175900,0.9333250,0.6858780,0.1373090,
     &   0.1912720,0.0213102,0.1205952,0.3030890,0.2301880,0.4353170,
     &   0.9443885,0.2414900,0.0332621,0.8974640,0.2460152,0.6730200,
     &   0.6332500,0.7872860,0.2680789,0.6489650,0.0594251,0.7442220,
     &   0.4070292,0.7394200,0.3640520,0.7644790,0.2131620,0.0159938,
     &   0.2881370,0.7454290,0.3906194,0.3663180,0.0539127,0.3042640,
     &   0.7973894,0.6700200,0.5050870,0.9956230,0.4453065,0.4865920,
     &   0.2968580,0.7726590,0.5000511,0.3929410,0.4655970,0.8357490 /
         TEMP = 0.0 D0
         DO K=1,60
            KDBL = DBLE(K)
            TEMP = TEMP + EXP(-ZETA*(KDBL-1.0D0)**2)*( 0.5 D0 -
     &         R1(K,IPROFILE) )*COS(KDBL*Y)
         END DO
         BASEPROFILE = TEMP
      ENDIF
      RETURN
      END
```

Package diagnostics-level

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE LINEINTEGRAL
C A general purpose routine for computing line integral of some flow
C  variable along a line.  This is to be called from Amrita
#include "../slice.H"
      AMRDBL PATH,S1,S2,S(slicemax),FUNCT(slicemax),INTEGRAL
      AMRSTR*255 LINETYPE,NORMALIZE,FIELD
      AMRINT I,I1,I2,N,TYPE,AMR_LEN
C
      CALL AMR_GET_TOKEN('STR::linetype',LINETYPE)
      CALL AMR_GET_TOKEN('DBL::coordinate',PATH)
      CALL AMR_GET_TOKEN('DBL::Lbound',S1)
      CALL AMR_GET_TOKEN('DBL::Hbound',S2)
      CALL AMR_GET_TOKEN('STR::normalize',NORMALIZE)
      CALL AMR_GET_TOKEN('STR::field',FIELD)
C
      IF (LINETYPE(1:AMR_LEN(LINETYPE)).EQ.'x') THEN
```

```
          TYPE = xslice
        ELSE
          TYPE = yslice
        ENDIF
        CALL GET_SLICE(N,TYPE,PATH,FIELD,S,FUNCT)
C
C Maybe we only compute line integral over a section of the entire line
C
        IF (S(1).LT.S1) THEN
          IF (S(N).LT.S1) THEN
            WRITE(AMR_ERR,*) "S1 too big in lineintegral"
          ELSE
            I1 = 2
            DO WHILE (S(I1).LT.S1)
              I1 = I1 + 1
            END DO
          ENDIF
        ELSE
          I1 = 1
        ENDIF
        IF (S(N).GT.S2) THEN
          IF (S(1).GT.S2) THEN
            WRITE(AMR_ERR,*) "S2 too small in lineintegral"
          ELSE
            I2 = N-1
            DO WHILE (S(I2).GT.S2)
              I2 = I2 - 1
            END DO
          ENDIF
        ELSE
          I2 = N
        ENDIF
C
C Compute the integral
C
        INTEGRAL = 0.0 D0
        DO I=I1,(I2-1)
          INTEGRAL = INTEGRAL + 0.5 D0*(FUNCT(I+1)+FUNCT(I))*(S(I+1)-S(I))
        END DO
        IF (NORMALIZE(1:AMR_LEN(NORMALIZE)).EQ.'y') THEN
          INTEGRAL = INTEGRAL / (S(I2)-S(I1))
        ENDIF
        CALL AMR_SET_TOKEN('DBL::integral',INTEGRAL)
        RETURN
        END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
      SUBROUTINE INTERFACE_BY_LEVELSET
C This routine computes both the amplitude and growth rate of either a
C  single-mode or multi-mode interface.  If it is single-mode, then our
C  work in cut down considerably since we know what y-values the peak and
C  through must be on.  If a multi-mode interface, then we search all possible
C  y values.
      IMPLICIT NONE
      AMRDBL YBOT,YTOP,A,HIGHEST_RES
      AMRSTR*255 TYPE
      AMRDBL Y,X,U,XBOT,UBOT,XTOP,UTOP,XLEFT,XRIGHT,ULEFT,URIGHT
      AMRDBL AMP,DADT
      AMRINT AMR_LEN
C
      CALL AMR_GET_TOKEN('DBL::ybot',YBOT)
      CALL AMR_GET_TOKEN('DBL::ytop',YTOP)
      CALL AMR_GET_TOKEN('DBL::a',A)
      CALL AMR_GET_TOKEN('DBL::highest_res',HIGHEST_RES)
      CALL AMR_GET_TOKEN('STR::interface_type',TYPE)
C
      IF (TYPE(1:AMR_LEN(TYPE)).EQ."single-mode") THEN
         CALL FIND_INTERFACE_BY_LEVELSET (YBOT,XBOT,UBOT)
         CALL FIND_INTERFACE_BY_LEVELSET (YTOP,XTOP,UTOP)
         IF (XTOP .LT. XBOT) THEN
            XLEFT  = XTOP
            XRIGHT = XBOT
            ULEFT  = UTOP
            URIGHT = UBOT
         ELSE
            XLEFT  = XBOT
            XRIGHT = XTOP
            ULEFT  = UBOT
            URIGHT = UTOP
         ENDIF
      ELSE
         Y      = YBOT
         XLEFT  =  1.0 D20
         XRIGHT = -1.0 D20
         DO WHILE (Y .LE. YTOP)
            CALL FIND_INTERFACE_BY_LEVELSET (Y,X,U)
            IF (X .LT. XLEFT) THEN
               XLEFT  = X
               ULEFT  = U
            ENDIF
            IF (X .GT. XRIGHT) THEN
               XRIGHT = X
               URIGHT = U
            ENDIF
```

```
            Y = Y + HIGHEST_RES
         END DO
      ENDIF
C
C If we couldn't find rho_avg, then assume shock hasn't hit interface yet
C
      IF ((XLEFT .EQ. 0.0D0).OR.(XRIGHT .EQ. 0.0D0)) THEN
         AMP  = A
         DADT = 0.0 D0
      ELSE
         AMP  = 0.5 D0*(XRIGHT - XLEFT)
         DADT = 0.5 D0*(URIGHT - ULEFT)
      ENDIF
      CALL AMR_SET_TOKEN('DBL::amp',AMP)
      CALL AMR_SET_TOKEN('DBL::dadt',DADT)
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE FIND_INTERFACE_BY_LEVELSET (Y,XLOC,VEL)
C This routine will slice the flowfield at y=Y and look for the occurance
C   of psi=0.  If so, it will return the x-location and u-velocity there
C   in X and VEL.  If not, it will return X=0.0 d0
#include "../slice.H"
      AMRDBL Y,XLOC,VEL
      AMRDBL X(slicemax),U(slicemax),PSI(slicemax)
      AMRDBL PSI1,PSI2,X1,X2,U1,U2,SLOPE,XOFFSET
      AMRINT I,N
C
      CALL GET_SLICE(N,yslice,Y,"psi",X,PSI)
      CALL GET_SLICE(N,yslice,Y,"u",X,U)
      I = 1
      DO WHILE (PSI(I) .GT. 0.0D0)
        IF (I.EQ.N) GOTO 10
        I = I + 1
      END DO
      PSI1    = PSI(I-1)
      PSI2    = PSI(I)
      X1      = X(I-1)
      X2      = X(I)
      U1      = U(I-1)
      U2      = U(I)
      SLOPE   = (PSI2 - PSI1)/(X2 - X1)
      IF (ABS(SLOPE) .LT. 1.0D-13) THEN
         XOFFSET = 0.0 D0
      ELSE
         XOFFSET = -PSI1/SLOPE
```

```
      ENDIF
      XLOC    = XOFFSET + X1
      VEL     = XOFFSET*(U2-U1)/(X2-X1) + U1
      GOTO 100
 10   XLOC    = 0.0 D0                /* Couldn't find psi=0 so set flag */
 100  RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE GET_SLICE(N,TYPE,PATH,FIELD,COORDS,ARRAY)
#include "../slice.H"
      AMRINT N,L,TYPE,GRD,GNO,GIJ,WIJ
      AMRDBL PATH,GET_FLOW,Xcg,Ycg
      AMRDBL COORDS(slicemax),ARRAY(slicemax)
      AMRSTR*255 FIELD
      CALL FLAG_REFINED_CELLS
      N = 0
      DO L=0,LMAX
         DO GNO=1,NGA(L)
            GRD = GP(L)+GNO
            DO I=1,IMX(GRD)
               DO J=1,JMX(GRD)
                  IJ = INDEXgij(GRD,I,J)
                  IF((TYPE.EQ.xslice).AND.(X(GRD,I,J).LE.PATH).AND.
     X              (X(GRD,I+1,J).GT.PATH)) THEN
                     IF(NOT_FLAGGED(IJ)) THEN
                        IF(N.EQ.slicemax) GO TO 200
                        N = N + 1
                        GIJ = INDEXgij(GRD,-1,-1)
                        WIJ = (GIJ-1)*NEQN+1
                        COORDS(N) = Ycg(GRD,I,J)
                        ARRAY(N)  = GET_FLOW(FIELD,GRD,I,J,IMX(GRD),
     X                     JMX(GRD),W(WIJ))
                     ENDIF
                  ELSE IF ((TYPE.EQ.yslice).AND.(Y(GRD,I,J).LE.PATH)
     X              .AND.(Y(GRD,I,J+1).GT.PATH)) THEN
                     IF(NOT_FLAGGED(IJ)) THEN
                        IF(N.EQ.slicemax) GO TO 200
                        N = N + 1
                        GIJ = INDEXgij(GRD,-1,-1)
                        WIJ = (GIJ-1)*NEQN+1
                        COORDS(N) = Xcg(GRD,I,J)
                        ARRAY(N)  = GET_FLOW(FIELD,GRD,I,J,IMX(GRD),
     X                     JMX(GRD),W(WIJ))
                     ENDIF
                  ENDIF
               END DO
```

```
               END DO
            END DO
         END DO
 200     CONTINUE
         CALL SLICE_SORT(COORDS,ARRAY,N)  /* slice_sort already in amr_sol */
         RETURN
         END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         AMRDBL FUNCTION GET_FLOW(FIELD,GRD,I,J,IM,JM,WW)
#include "../slice.H"
         AMRSTR*255 FIELD
         AMRINT GRD,I,J,IM,JM,AMR_LEN
         AMRDBL WW(5,-1:IM+2,-1:JM+2)
         IF (FIELD(1:AMR_LEN(FIELD)).EQ."rho") THEN
           GET_FLOW = WW(1,I,J)
         ELSE IF (FIELD(1:AMR_LEN(FIELD)).EQ."u") THEN
           GET_FLOW = WW(2,I,J)/WW(1,I,J)
         ELSE IF (FIELD(1:AMR_LEN(FIELD)).EQ."v") THEN
           GET_FLOW = WW(3,I,J)/WW(1,I,J)
         ELSE IF (FIELD(1:AMR_LEN(FIELD)).EQ."mx") THEN
           GET_FLOW = WW(2,I,J)
         ELSE IF (FIELD(1:AMR_LEN(FIELD)).EQ."my") THEN
           GET_FLOW = WW(3,I,J)
         ELSE IF (FIELD(1:AMR_LEN(FIELD)).EQ."psi") THEN
           GET_FLOW = WW(5,I,J)/WW(1,I,J)
         ELSE
           WRITE(AMR_ERR,*) '*** GET_FLOW does not understand FIELD of'
           WRITE(AMR_ERR,*) (FIELD(N:N),N=1,AMR_LEN(FIELD))
         ENDIF
         RETURN
         END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
#include "/home/ama/d/mrm/AMRITAv31/src/amr_sol/AMR_PARSE"
#include "/home/ama/d/mrm/AMRITAv31/src/amr_sol/AMR_PRINT"
         SUBROUTINE AREA_CIRCULATION
#include "/home/ama/d/mrm/AMRITAv31/include/f77/AMR_SOL/DSTR.H"
#include "/home/ama/d/mrm/AMRITAv31/include/f77/AMR_SOL/ADAPT.H"
C Compute area integral of voriticity to give circulation.  The area will
C  be determined on a cell-by-cell basis according to the test given in
C  subroutine get_vorticity below.  Note that the test must be compiled
C  in, however, we allow the user one input parameter that can be set from
C  the calling Amrita procedure.
         AMRINT L,GRD,GNO,GIJ,WIJ
         AMRDBL CIRC,GET_VORTICITY,VOL,OMEGA,DA,TESTPARAM
C
```

```
      CALL AMR_GET_TOKEN('DBL::parameter',TESTPARAM)
C
      CIRC = 0.0 D0
      CALL FLAG_REFINED_CELLS
      DO L=0,LMAX
         DO GNO=1,NGA(L)
            GRD = GP(L)+GNO
            DO I=1,IMX(GRD)
               DO J=1,JMX(GRD)
                  IJ = INDEXgij(GRD,I,J)
                  IF(NOT_FLAGGED(IJ)) THEN
                     DA  = VOL(GRD,I,J)
                     GIJ = INDEXgij(GRD,-1,-1)
                     WIJ = (GIJ-1)*NEQN+1
                     OMEGA = GET_VORTICITY(GRD,I,J,IMX(GRD),JMX(GRD),
     X                  W(WIJ),TESTPARAM)
                     CIRC  = CIRC + OMEGA*DA
                  ENDIF
               END DO
            END DO
         END DO
      END DO
C
      CALL AMR_SET_TOKEN('DBL::circulation',CIRC)
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      AMRDBL FUNCTION GET_VORTICITY(GRD,I,J,IM,JM,WW,TESTPARAM)
C Compute voriticty using center differences.  T=top, B=bot, L=left, R=right
      AMRINT GRD,I,J,IM,JM
      AMRDBL WW(5,-1:IM+2,-1:JM+2),TESTPARAM
      AMRDBL Xcg,Ycg
      AMRINT IR,IL,JT,JB
      AMRDBL XR,XL,YT,YB
      AMRDBL VR,VL,UT,UB
      AMRDBL PSI
      AMRLOG FLAG
C
      IR = I + 1
      IL = I - 1
      JT = J + 1
      JB = J - 1
C
C First do the test to see if we need to compute vorticity here
C
      PSI = WW(5,I,J)/WW(1,I,J)
```

```
      IF (ABS(PSI).LE.TESTPARAM) THEN
         FLAG = .TRUE.
      ELSE
         FLAG = .FALSE.
      ENDIF
C
C If needed, compute vorticity in this cell
C
      IF (FLAG) THEN
         XR = Xcg(GRD,IR,J)
         XL = Xcg(GRD,IL,J)
         YT = Ycg(GRD,I,JT)
         YB = Ycg(GRD,I,JB)
         UT = WW(2,I,JT)/WW(1,I,JT)
         UB = WW(2,I,JB)/WW(1,I,JB)
         VR = WW(3,IR,J)/WW(1,IR,J)
         VL = WW(3,IL,J)/WW(1,IL,J)
         GET_VORTICITY = (VR-VL)/(XR-XL) - (UT-UB)/(YT-YB)
      ELSE
         GET_VORTICITY = 0.0 D0
      ENDIF
      RETURN
      END
```

Package biot_layer

```
#define bot 1
#define top 2
#define bignum 1.0D3
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE BIOT_SAVART
C Computes the growth rate using vortiicty distribution computed from
C  Amrita coupled with the Biot-Savart law.
#include "../slice.H"
      AMRDBL XBOT,YBOT,XTOP,YTOP,Xincomp(bot:top),Yincomp(bot:top)
      AMRDBL UBOT,UTOP,Uincomp(bot:top),Vincomp(bot:top)
      AMRDBL OMEGA,Xvort,Yvort,Yimage,PI,PSITEST
      AMRDBL SINHX,COSHX,SINYvort,COSYvort,SINYimage,COSYimage,DXDY
      AMRDBL GET_VORTICITY,GET_FLOW
      AMRINT L,GNO,GRD,GIJ,WIJ,IJ,POINT

      CALL AMR_GET_TOKEN('DBL::ybot',YBOT)
      CALL AMR_GET_TOKEN('DBL::ytop',YTOP)

      PI = 4.0D0*ATAN(1.0D0)
      CALL FIND_INTERFACE_BY_LEVELSET (YBOT,XBOT,UBOT)
```

```
      CALL FIND_INTERFACE_BY_LEVELSET (YTOP,XTOP,UTOP)
      Xincomp(bot) = XBOT
      Yincomp(bot) = YBOT
      Xincomp(top) = XTOP
      Yincomp(top) = YTOP
      DO POINT=bot,top
         Uincomp(POINT) = 0.0D0
         Vincomp(POINT) = 0.0D0
      ENDDO

      CALL FLAG_REFINED_CELLS
      DO L=0,LMAX
         DXDY = Xscale(L)*Yscale(L)
         DO GNO=1,NGA(L)
            GRD = GP(L) + GNO
            DO I=1,IMX(GRD)
               DO J=1,JMX(GRD)
                  Xvort  = Xcg(GRD,I,J)
                  Yvort  = Ycg(GRD,I,J)
                  Yimage = 2.0D0-Yvort
                  IJ = INDEXgij(GRD,I,J)
                  IF (NOT_FLAGGED(IJ)) THEN
                     GIJ = INDEXgij(GRD,-1,-1)
                     WIJ = (GIJ-1)*NEQN+1
                     OMEGA = GET_VORTICITY(GRD,I,J,IMX(GRD),JMX(GRD),
     X                  W(WIJ),bignum)
                     PSITEST = GET_FLOW("psi",GRD,I,J,IMX(GRD),
     X                  JMX(GRD),W(WIJ))
                     IF (ABS(PSITEST).LE.0.99D0) THEN
                        DO POINT=bot,top
                           SINHX     = SINH(PI*(Xincomp(POINT)-Xvort))
                           COSHX     = COSH(PI*(Xincomp(POINT)-Xvort))
                           SINYvort  = SIN(PI*(-Yincomp(POINT)+Yvort))
                           COSYvort  = COS(PI*(-Yincomp(POINT)+Yvort))
                           SINYimage = SIN(PI*(-Yincomp(POINT)+Yimage))
                           COSYimage = COS(PI*(-Yincomp(POINT)+Yimage))
                           Uincomp(POINT) = Uincomp(POINT) + 0.25D0*
     X                       OMEGA*( SINYvort/(COSHX-COSYvort) -
     X                       SINYimage/(COSHX-COSYimage) )*DXDY
                           Vincomp(POINT) = Vincomp(POINT) + 0.25D0*
     X                       OMEGA*SINHX*( 1.0D0/(COSHX-COSYvort)
     X                       - 1.0D0/(COSHX-COSYimage) )*DXDY
                        END DO
                     END IF
                  END IF
               END DO
            END DO
```

```
         END DO
      END DO

      CALL AMR_SET_TOKEN('DBL::xbot',Xincomp(bot))
      CALL AMR_SET_TOKEN('DBL::xtop',Xincomp(top))
      CALL AMR_SET_TOKEN('DBL::ubot',UBOT)
      CALL AMR_SET_TOKEN('DBL::utop',UTOP)
      CALL AMR_SET_TOKEN('DBL::ubot_biot',Uincomp(bot))
      CALL AMR_SET_TOKEN('DBL::vbot_biot',Vincomp(bot))
      CALL AMR_SET_TOKEN('DBL::utop_biot',Uincomp(top))
      CALL AMR_SET_TOKEN('DBL::vtop_biot',Vincomp(top))

      RETURN
      END
```

Package `max_vorticity`

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE MAX_VORTICITY_CURVE
#include "/tmp_mnt2/mrm/AMRITAv31/include/f77/AMR_SOL/AMRITA.H"
      AMRDBL YTOP,YBOT,XNOW,YNOW,XOLD,YOLD,VALMAX
      AMRDBL S
C
      CALL AMR_GET_TOKEN("DBL::ybot",YBOT)
      CALL AMR_GET_TOKEN("DBL::ytop",YTOP)
C
      S = 0.0D0
      YNOW = YBOT
      YOLD = YNOW
      CALL LOCATE_MAX_OMEGA (YNOW,XNOW,VALMAX)
      XOLD = XNOW
C
      DO WHILE (YNOW.LT.YTOP)
         CALL LOCATE_MAX_OMEGA (YNOW,XNOW,VALMAX)
         S = S + SQRT( (XNOW-XOLD)**2 + (YNOW-YOLD)**2 )
         WRITE(23,*) FLOAT(S),FLOAT(VALMAX),FLOAT(XNOW),FLOAT(YNOW)
         XOLD = XNOW
         YOLD = YNOW
         YNOW = YNOW + Yscale(0)
      END DO
      YNOW = YTOP
      CALL LOCATE_MAX_OMEGA (YNOW,XNOW,VALMAX)
      S = S + SQRT( (XNOW-XOLD)**2 + (YNOW-YOLD)**2 )
      WRITE(23,*) FLOAT(S),FLOAT(VALMAX),FLOAT(XNOW),FLOAT(YNOW)

      CALL FLUSH(23)
```

```
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE LOCATE_MAX_OMEGA (Yslice,Xloc,VALMAX)
#include "/tmp_mnt2/mrm/AMRITAv31/include/f77/AMR_SOL/AMRITA.H"
      AMRDBL Yslice,Xloc,VALMAX
      AMRINT N,GRD,GNO,GIJ,WIJ,I,J
      AMRDBL GET_FLOW,Xcg,Ycg
      AMRDBL ABSMAX,COORDS(slicemax),OMEGA(slicemax)
C
      N = 0
      DO GNO=1,NGA(LMAX)                  /* ONLY SEARCH TOP GRID LEVEL */
         GRD = GP(LMAX) + GNO
         DO I=1,IMX(GRD)
            DO J=1,JMX(GRD)
               IJ = INDEXgij(GRD,I,J)
               IF(((Y(GRD,I,J).LE.Yslice).AND.
     X             (Y(GRD,I,J+1).GT.Yslice))) THEN
                  IF(N.EQ.slicemax) GO TO 200
                  GIJ = INDEXgij(GRD,-1,-1)
                  WIJ = (GIJ-1)*NEQN+1
                  PSI = GET_FLOW("PSI",GRD,I,J,IMX(GRD),JMX(GRD),W(WIJ))
                  IF (ABS(PSI).LE.0.99D0) THEN       /* Don't count shocks */
                     N = N + 1
                     COORDS(N) = Xcg(GRD,I,J)
                     OMEGA(N)  = GET_FLOW("omega",GRD,I,J,IMX(GRD),
     X                 JMX(GRD),W(WIJ))
                  ENDIF
               ENDIF
            END DO
         END DO
      END DO
 200  CONTINUE

C Now find the maximum on this slice

      ABSMAX = 1.0 D-10
      DO K=1,N
         IF (ABS(OMEGA(K)).GT.ABSMAX) THEN
            VALMAX    = OMEGA(K)
            ABSMAX = ABS(VALMAX)
            Xloc     = COORDS(K)
         ENDIF
      END DO

      RETURN
```

```
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      AMRDBL FUNCTION GET_FLOW(FIELD,GRD,I,J,IM,JM,WW)
#include "../slice.H"
      AMRSTR*255 FIELD
      AMRINT GRD,I,J,IM,JM,AMR_LEN,L
      AMRDBL WW(5,-1:IM+2,-1:JM+2)
      AMRDBL Up1,Um1,Vp1,Vm1
      AMRDBL Xcg,Ycg
C
      L = AMR_LEN(FIELD)
      IF (FIELD(1:L).EQ."RHO") THEN
        GET_FLOW = WW(1,I,J)
      ELSE IF (FIELD(1:L).EQ."U") THEN
        GET_FLOW = WW(2,I,J)/WW(1,I,J)
      ELSE IF (FIELD(1:L).EQ."V") THEN
        GET_FLOW = WW(3,I,J)/WW(1,I,J)
      ELSE IF (FIELD(1:L).EQ."mx") THEN
        GET_FLOW = WW(2,I,J)
      ELSE IF (FIELD(1:L).EQ."my") THEN
        GET_FLOW = WW(3,I,J)
      ELSE IF (FIELD(1:L).EQ."PSI") THEN
        GET_FLOW = WW(5,I,J)/WW(1,I,J)
      ELSE IF (FIELD(1:L).EQ."X") THEN
        GET_FLOW = Xcg(GRD,I,J)
      ELSE IF (FIELD(1:L).EQ."Y") THEN
        GET_FLOW = Ycg(GRD,I,J)
      ELSE IF (FIELD(1:L).EQ."omega") THEN
        Up1      = WW(2,I,J+1)/WW(1,I,J+1)
        Um1      = WW(2,I,J-1)/WW(1,I,J-1)
        Vp1      = WW(3,I+1,J)/WW(1,I+1,J)
        Vm1      = WW(3,I-1,J)/WW(1,I-1,J)
        GET_FLOW = (Vp1-Vm1)/(Xcg(GRD,I+1,J)-Xcg(GRD,I-1,J)) -
     X             (Up1-Um1)/(Ycg(GRD,I,J+1)-Ycg(GRD,I,J-1))
      ELSE IF (FIELD(1:L).EQ."divergence") THEN
        Up1      = WW(2,I+1,J)/WW(1,I+1,J)
        Um1      = WW(2,I-1,J)/WW(1,I-1,J)
        Vp1      = WW(3,I,J+1)/WW(1,I,J+1)
        Vm1      = WW(3,I,J-1)/WW(1,I,J-1)
        GET_FLOW = (Up1-Um1)/(Xcg(GRD,I+1,J)-Xcg(GRD,I-1,J)) +
     x             (Vp1-Vm1)/(Ycg(GRD,I,J+1)-Ycg(GRD,I,J-1))
      ELSE
        WRITE(AMR_ERR,*) '*** GET_FLOW does not understand FIELD of'
        WRITE(AMR_ERR,*) (FIELD(N:N),N=1,AMR_LEN(FIELD))
      ENDIF
      RETURN
```

```
      END
```

Package `avg_along_interface`

```
      SUBROUTINE AVG_ALONG_INTERFACE
#include "../slice.H"

C Computes the average value of the divergence of (u,v) along the curve
C  psi=0.
C WARNING:  This routine only checks the topmost grid level for the psi=0
C  curve so if the interface isn't completely refined then this routine
C  will fail.

      AMRSTR*255 FIELD
      AMRINT N,L,GRD,GNO,GIJ,WIJ
      AMRDBL XL,XR,PSIL,PSIR,VALL,VALR,SLOPE,XOFFSET
      AMRDBL AVG,VALUE
      AMRDBL Xcg,Ycg,GET_FLOW
      AMRINT AMR_SCREEN

      CALL AMR_GET_TOKEN('STR::field',FIELD)

      N = 0
      AVG = 0.0D0
      L = LMAX
      DO GNO=1,NGA(L)
         GRD = GP(L)+GNO
         DO I=1,IMX(GRD)
            DO J=1,JMX(GRD)
               GIJ  = INDEXgij(GRD,-1,-1)
               WIJ  = (GIJ-1)*NEQN+1
               PSIL = GET_FLOW("PSI",GRD,I,J,IMX(GRD),JMX(GRD),W(WIJ))
               XL   = Xcg(GRD,I,J)
               PSIR = GET_FLOW("PSI",GRD,I+1,J,IMX(GRD),JMX(GRD),W(WIJ))
               XR   = Xcg(GRD,I+1,J)
               IF (PSIL*PSIR.LT.0.0D0) THEN
                  N = N + 1
                  VALL = GET_FLOW(FIELD,GRD,I,J,IMX(GRD),JMX(GRD),
     X               W(WIJ))
                  VALR = GET_FLOW(FIELD,GRD,I+1,J,IMX(GRD),JMX(GRD),
     X               W(WIJ))
                  SLOPE   = (PSIR-PSIL)/(XR-XL)
                  XOFFSET = -PSIL/SLOPE
                  VALUE   = XOFFSET*(VALR-VALL)/(XR-XL) + VALL
                  AVG     = AVG + VALUE
               ENDIF
            END DO
```

```
          END DO
        END DO

        AVG = AVG / DBLE(N)
        CALL AMR_SET_TOKEN('DBL::average',AVG)

        RETURN
        END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

        AMRDBL FUNCTION GET_FLOW(FIELD,GRD,I,J,IM,JM,WW)
#include "../slice.H"
        AMRSTR*255 FIELD
        AMRINT GRD,I,J,IM,JM,AMR_LEN,L
        AMRDBL WW(5,-1:IM+2,-1:JM+2)
        AMRDBL Up1,Um1,Vp1,Vm1
        AMRDBL Xcg,Ycg
        AMRINT AMR_SCREEN
C
        L = AMR_LEN(FIELD)
        IF (FIELD(1:L).EQ."RHO") THEN
          GET_FLOW = WW(1,I,J)
        ELSE IF (FIELD(1:L).EQ."U") THEN
          GET_FLOW = WW(2,I,J)/WW(1,I,J)
        ELSE IF (FIELD(1:L).EQ."V") THEN
          GET_FLOW = WW(3,I,J)/WW(1,I,J)
        ELSE IF (FIELD(1:L).EQ."mx") THEN
          GET_FLOW = WW(2,I,J)
        ELSE IF (FIELD(1:L).EQ."my") THEN
          GET_FLOW = WW(3,I,J)
        ELSE IF (FIELD(1:L).EQ."PSI") THEN
          GET_FLOW = WW(5,I,J)/WW(1,I,J)
        ELSE IF (FIELD(1:L).EQ."X") THEN
          GET_FLOW = Xcg(GRD,I,J)
        ELSE IF (FIELD(1:L).EQ."Y") THEN
          GET_FLOW = Ycg(GRD,I,J)
        ELSE IF (FIELD(1:L).EQ."omega") THEN
          Up1      = WW(2,I,J+1)/WW(1,I,J+1)
          Um1      = WW(2,I,J-1)/WW(1,I,J-1)
          Vp1      = WW(3,I+1,J)/WW(1,I+1,J)
          Vm1      = WW(3,I-1,J)/WW(1,I-1,J)
          GET_FLOW = (Vp1-Vm1)/(Xcg(GRD,I+1,J)-Xcg(GRD,I-1,J)) -
     X               (Up1-Um1)/(Ycg(GRD,I,J+1)-Ycg(GRD,I,J-1))
        ELSE IF (FIELD(1:L).EQ."divu") THEN
          Up1      = WW(2,I+1,J)/WW(1,I+1,J)
```

```
    Um1      = WW(2,I-1,J)/WW(1,I-1,J)
    Vp1      = WW(3,I,J+1)/WW(1,I,J+1)
    Vm1      = WW(3,I,J-1)/WW(1,I,J-1)
    GET_FLOW = ABS( (Up1-Um1)/(Xcg(GRD,I+1,J)-Xcg(GRD,I-1,J)) +
x                   (Vp1-Vm1)/(Ycg(GRD,I,J+1)-Ycg(GRD,I,J-1)) )
  ELSE
    WRITE(AMR_ERR,*) '*** GET_FLOW does not understand FIELD of'
    WRITE(AMR_ERR,*) (FIELD(N:N),N=1,AMR_LEN(FIELD))
  ENDIF
  RETURN
  END
```

Bibliography

[1] ABRAMOWITZ, M., STEGUN, I. A. 1972 Handbook of mathematical functions, with formulas, graphs, and mathematical tables, 9th edition. Dover Publications, New York.

[2] BALTRUSAITIS, R.M., GITTINGS, M. L., WEAVER, R. P., BENJAMIN, R. F., BUDZINSKI, J. M. 1996 Simulation of shock-generated instabilities *Phys. Fluids* **8**, pp. 2471–2483.

[3] BEST, J. P. 1991 A generalization of the theory of geometrical shock dynamics. *Shock Waves* **1**, pp. 251–273.

[4] BEST, J. P. 1993 Accounting for transverse flow in the theory of geometrical shock dynamics. *Proc. R. Soc. Lond. A* **442**, pp. 585–598.

[5] BROUILLETTE, M., STURTEVANT, B. 1993 Experiments on the Richtmyer-Meshkov instability: Small-scale perturbations on a plane interface. *Phys. Fluids A* **5**, pp. 916–930.

[6] BROUILLETTE, M., STURTEVANT, B. 1994 Experiments on the Richtmyer-Meshkov instability: single-scale perturbations on a continuous interface. *J. Fluid Mech.* **263**, pp. 271–292.

[7] CANUTO, C., HUSSAINI, M. Y., QUARTERONI, A., ZANG, T. A. 1988 Spectral Methods in Fluid Dynamics. Springer-Verlag, New York.

[8] CATES, J. E. 1996 Studies of Shock Wave Focusing Using Geometrical Shock Dynamics. PhD Thesis, California Institute of Technology.

[9] CATHERASOO, C. J., STURTEVANT, B. 1983 Shock dynamics in non-uniform media. *J. Fluid Mech.* **127**, pp. 539–561.

[10] CHISNELL, R. F. 1955 The normal motion of a shock wave through a non-uniform one-dimensional medium. *Proc. Roy. Soc.* **A232**, pp. 350–370.

[11] Colella, P. 1985 A direct Eulerian MUSCL scheme for gas dynamics. *SIAM J. Sci. Stat. Comput.* **6**, pp. 104–117.

[12] Colella, P. 1990 Multidimensional upwind methods for hyperbolic conservation laws. *J. Comput. Phys.* **87**, pp. 171–200.

[13] Collins, R., Chen, H. T. 1971 Motion of a shock wave through a non-uniform fluid. *Proc. 2nd Int. Conf. on Numerical Methods in Fluid Dynamics* Lecture Notes in Physics, vol. 8, pp. 264–269. Springer.

[14] Einfeldt, B. 1988 On Godunov-Type Methods for Gas Dynamics. *SIAM J. Numer. Anal.* **25**, pp. 294–318.

[15] Fraley, G. 1986 Rayleigh-Taylor stability for a normal shock-wave–density discontinuity interaction. *Phys. Fluids* **6**, pp. 376–386.

[16] Greenough, J. A. 1996 Private communication.

[17] Haas, J. F., Sturtevant, B. 1987 Interaction of weak shock waves with cylindrical and spherical gas inhomogeneities. *J. Fluid Mech.* **181**, pp. 41–76.

[18] Han, Z. Y., Yin, X. Z. 1989 2-dimensional equations of shock dynamics for moving gas ahead of shock wave. *Sci. in China A* **32**, pp.1333-1344.

[19] Henshaw, W. D., Smyth, N. F., Schwendeman, D. W. 1986 Numerical shock propagation using geometrical shock dynamics. *J. Fluid Mech.* **171**, pp. 519–545.

[20] Holmes, R. L, Grove, J. W., Sharp, D. H. 1995 Numerical investigation of Richtmyer-Meshkov instability using front tracking. *J. Fluid Mech.* **301**, pp. 51–64.

[21] Jacobs, P. A. 1991 Single-block Navier-Stokes integrator. *NASA CR-187613*, ICASE Interim Report 18.

[22] LeVeque, R. J. 1992 Numerical methods for conservation laws. Birkhauser Verlag, Boston.

[23] Li, C., Kailasanath, K., Book, D. L. 1991 Mixing enhancement by expansion waves in supersonic flows of different densities. *Phys. Fluids* **A**, pp. 1369–1373.

[24] LINDL, D. L., MCCRORY, R., CAMBELL, E. M. 1992 Progress toward ignition and burn propagation in inertial confinement fusion. *Physics Today* pp. 32–40, September.

[25] LIOU, M. S., STEFFEN, C. J. 1993 A New Flux Splitting Scheme. *J. Comput. Phys.* **107**, pp. 23–39.

[26] MESHKOV, E. E. 1970 Instability of a Shock Wave Accelerated Interface Between Two Gases. *NASA Tech. Trans.* **F-13**, 074.

[27] MEYER, K. A., BLEWETT, P. J. 1972 Numerical Investigation of the Stability of a Shock-Accelerated Interface between Two Fluids. *Phys. Fluids* **15**, pp. 753–759.

[28] MIKAELIAN, K. O. 1985 Richtmyer-Meshkov instabilities in stratified fluids. *Phys. Rev. A* **31**, pp. 410–419.

[29] MIKAELIAN, K. O. 1989 Turbulent mixing generated by Rayleigh-Taylor and Richtmyer-Meshkov instabilities. PHYSICA D **36**, pp. 343–357.

[30] MIKAELIAN, K. O. 1991 Density gradient stabilization of the Richtmyer-Meshkov instability. *Phys. Fluids A* **3**, pp. 2638–2643.

[31] MIKAELIAN, K. O. 1993 Growth rate of the Richtmyer-Meshkov instability at shocked interfaces. *Phys. Rev. Lett.* **71**, pp. 2903–2906.

[32] MIKAELIAN, K. O. 1994 Freeze-out and the effect of compressibility in the Richtmyer-Meshkov instability. *Phys. Fluids* **6**, pp. 356–368.

[33] MULDER W., OSHER, S., SETHIAN, J. A. 1992 Computing Interface Motion in Compressible Gas Dynamics. *J. Comp. Phys.* **100**, pp. 209–228.

[34] PEYSER, T. A., MILLER, P. L., STRY, P. E., BUDIL, K. S., BURKE, E. W., WOJTOWICZ, D. A., GRISWOLD, D. L., HAMMEL, B. A., PHILLION, D. W. 1995 Measurement of radiation-driven shock-induced mixing from nonlinear initial perturbations. *Phys. Rev. Lett.* **75**, pp. 2332–2335.

[35] PHAM, T. 1991 Numerical studies of incompressible Richtmyer-Meshkov instability in a stratified fluid. PhD Thesis, California Institute of Technology.

[36] Pham, T., Meiron, D. I. 1993 A numerical study of Richtmyer-Meshkov instability in continuously stratified fluids. *Phys. Fluids A* **5**, pp. 344–368.

[37] Pullin, D. I. 1980 Direct Simulation Methods for Compressible Inviscid Ideal-Gas Flow. *J. Comp. Phys.* **34**, pp. 231–244.

[38] Quirk, J. J. 1994 A contribution to the great Riemann solver debate. *Internat. J. Numer. Methods Fluids* **18**, pp. 555–574.

[39] Quirk, J. J. 1996 A parallel adaptive grid algorithm for computational shock hydrodynamics. *Appl. Numer. Math* **20**, pp. 427–453.

[40] Rayleigh, Lord 1900 Investigation of the Character of the Equilibrium of an Incompressible Heavy Fluid of Variable Density. *Scientific Papers* Vol. 2, Dover, New York.

[41] Richtmyer, R. D. 1960 Taylor Instability in Shock Acceleration of Compressible Fluids. *Comm. Pure Appl. Math.* **13**, pp. 297–319.

[42] Roberts, A. E. 1945 Stability of a steady plane shock. *Los Alamos National Laboratory Report No. LA299.*

[43] Roe, P. L. 1981 Numerical algorithms for the linear wave equation. *Royal Aircraft Establishment Technical Report 81047.*

[44] Roe, P. L. 1981 Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. *J. Comput. Phys.* **43**, pp. 357–372.

[45] Roe, P. L., Pike, J. 1984 Efficient Construction and Utilization of Approximate Riemann Solutions. *Comput. Math. Appl. Sci. & Eng.* **VI**, eds. Glowinski, R. and Lions, J-L., pp. 499–518.

[46] Roe, P. L. 1985 Some contributions to the modeling of discontinuous flows. *Lect. Notes Appl. Math.* **22**, pp. 163–193.

[47] Rupert, V. 1992 Shock-interface interaction: Current research on the Richtmyer-Meshkov problem. *Shock Waves*, Proceedings of the 18th Intl. Symp. on Shock Waves, pp. 83–94.

[48] Saffman, P. G. 1992 Vortex Dynamics. Cambridge University Press, Cambridge.

[49] Saffman, P. G., Meiron, D. I. 1989 Kinetic energy generated by the incompressible Richtmyer-Meshkov instability in a continuously stratified fluid. *Phys. Fluids A* **1**, pp. 1767–1771.

[50] Samtaney, R. 1994 Private communication.

[51] Samtaney, R., Zabusky, N. J. 1994 Circulation deposition on shock-accelerated planar and curved density-stratified interfaces: Models and scaling laws. *J. Fluid Mech.* **269**, pp. 45–78.

[52] Samtaney, R., Zabusky, N. J. 1994 Vorticity Generation and Growth Rate in Richtmyer-Meshkov Instability. *Caip Tech. Report No. 187,* Rutgers University.

[53] Samtaney, R., Pullin, D. I. 1996 Initial-value and self-similar solutions of the compressible Euler equations. *Phys. Fluids* **8**, pp. 2650–2655.

[54] Samtaney, R., Meiron, D. I. 1997 Hypervelocity Richtmyer-Meshkov Instability. *Phys. Fluids,* **9**, pp. 1783–1803.

[55] Schwendeman, D. W. 1988 Numerical shock propagation in non-uniform media. *J. Fluid Mech.* **188**, pp. 383–410.

[56] Schwendeman, D. W. 1993 A new numerical method for shock wave propagation based on geometrical shock dynamics. *Proc. R. Soc. Lond. A.* **441**, pp. 331–341.

[57] Strang, G. 1968 On the construction and comparison of difference schemes. *SIAM J. Num. Anal.* **5**, pp. 506–517.

[58] Sweby, P. K. 1984 High resolution schemes using flux limiters for hyperbolic conservation laws. *J. Numer. Anal.* **21**, pp. 995–1011.

[59] Taylor, G. I. 1950 The instability of liquid surfaces when accelerated in a direction perpendicular to their planes, I. *Proc. R. Soc. London Ser. A* **201**, pp. 192–196.

[60] Toro, E.F. 1989 A Fast Riemann Solver with Constant Covolume Applied to the Random Choice Method. *Int. J. Numer. Method Fluids* **9**, pp. 1145–1164.

[61] van Leer, B. 1974 Towards the ultimate conservative difference scheme, II. Monotonicity and conservation combined in a second-order scheme. *J. Comput. Phys.* **14**, pp. 361–370.

[62] Velikovich, A. L. 1996 Analytic theory of Richtmyer-Meshkov instability for the case of reflected rarefaction wave. *Phys. Fluids* **8**, pp. 1666–1678.

[63] Vetter, M., Sturtevant, B. 1995 Experiments on the Richtmyer-Meshkov instability of an Air/SF_6 interface. *Shock Waves* **4**, pp. 247–252.

[64] Whitham, G. B. 1957 A new approach to problems of shock dynamics. Part I, Two-dimensional problems. *J. Fluid Mech.* **2**, pp. 146–171.

[65] Whitham, G. B. 1959 A new approach to problems of shock dynamics. Part II, Three-dimensional problems. *J. Fluid Mech.* **5**, pp. 369–386.

[66] Whitham, G. B. 1968 A note on shock dynamics relative to a moving frame. *J. Fluid Mech.* **31**, pp. 449–453.

[67] Whitham, G. B. 1974 Linear and nonlinear waves. Wiley, New York.

[68] Wouchuk, J. G., Nishihara, K. 1997 Asymptotic growth in the linear Richtmyer-Meshkov instability. *Phys. Plasmas*, **4**, pp. 1028–1038.

[69] Yang, Y., Zhang, Q., Sharp, D. H. 1994 Small Amplitude Theory of Richtmyer-Meshkov Instability. *Phys. Fluids* **6**, pp. 1856–1873.

[70] Yee, H. C., Warming, R. F., Harten, A. 1985 Implicit total variation diminishing (TVD) schemes for steady-state calculations. *J. Comput. Phys.* **57**, pp. 327–360.

[71] Yin, X. 1995 Characteristic Method of Shock Dynamics Used in Shock-Vortex Interaction. *Trans. Japan Soc. Aero. Space Sci.* **38**, pp. 151–160.

[72] Zhang, Q., Sohn, S. 1996 An analytical nonlinear theory of Richtmyer-Meshkov instability. *Phys. Let. A* **212**, pp. 149–155.