Front Tracking Methods based on Wave Propagation

by

Keh-Ming Shyue

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

University of Washington

1993

Approved by_____
(Chairperson of Supervisory Committee)

_____

_____

Program Authorized
to Offer Degree_____

Date_____

In presenting this dissertation in partial fulfillment of the requirements for the Doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to University Microfilms, 1490 Eisenhower Place, P. O. Box 975, Ann Arbor, Michigan 48106, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature_____

Date_____

University of Washington

Abstract

Front Tracking Methods based on Wave Propagation

by Keh-Ming Shyue

Chairperson of Supervisory Committee:
Professor Randall J. LeVeque
Departments of Mathematics and Applied Mathematics

In this thesis, we develop and study a simple front tracking approach that models the propagation of discontinuous solutions for nonlinear hyperbolic systems of conservation laws with source terms,

$$\frac{\partial u}{\partial t} + \sum_{j=1}^{N} \frac{\partial}{\partial x_j} f_j(u) = \psi(u),$$

in both one ($N = 1$) and two space dimensions ($N = 2$). In this approach, we use a uniform underlying grid with some grid cells subdivided by tracked interfaces, made up of moving points in one space dimension and curves in two space dimensions, approximately aligned with discontinuities in the flow field. In each time step, we introduce a new set of interfaces that are close to the expected locations of discontinuities in the solution at the end of the time step. A conservative high resolution finite volume method based on the *large time step* wave propagation approach is then applied on the resulting nonuniform grid to update the cell values. A time-splitting method is employed to handle source terms.

Our error estimation results show that this front tracking method is stable even if some of the small cells created by the tracked interfaces are orders of magnitude smaller than the regular cell that is used to determine the time step. In addition, high resolution results can be obtained for cells near the tracked discontinuities without oscillations.

A wide variety of problems have been solved to validate the method for problems involving shock waves and interfaces (contact discontinuities and slip lines) arising in gas dynamics. Typical examples we consider are a one-dimensional unstable detonation wave problem, a two-dimensional shock-ramp interaction, and two-dimensional Kelvin-Helmholtz and Rayleigh-Taylor instabilities. These results show the effectiveness of our front tracking methods in both one and two space dimensions. They also show the importance of using front tracking for these problems.

This thesis also describes an algorithm that can be used to solve a coupled system of elliptic and hyperbolic partial differential equations arising in oil reservoir simulation. Our preliminary results indicate that front tracking is a very useful tool for this problem.

ACKNOWLEDGMENTS

I would like to express sincere appreciation to my thesis advisor Professor Randall J. LeVeque for his invaluable guidance and assistance throughout the work. I appreciate very much the freedom that he has given to me to explore things on my own. Thanks also for giving me the opportunity to visit Europe and work at ETH-Zürich.

I would like to thank the other members of my supervisory committee: Professor Loyce Adams, Professor Scott Eberhardt, and Professor Ka-Kit Tung for their time and constructive suggestions. I also want to thank Dr. Daniel Bentil and Dr. Hoshuenn Huang for their help in studying the unstable interface problems. I want to thank my friends Rolf Walder and Brigitte Müller for their time and hospitality during my stay at Zürich. I am very grateful for that. I also want to thank ETH and particularly the members of the Seminar für Angewandte Mathematik for their help and support. Finally, I would like to thank my parents for their care and support throughout the years.

This research was supported in part by NSF grants DMS-8657319, DMS-9344453, and by ETH-Zürich.

Chapter 1

INTRODUCTION

1.1 Preliminaries

Sharp fronts are commonly seen in the real world. In many applications, they typically move as time evolves and often undergo some complicated physical processes, displaying a rich variety of frontal structure, see [44],[77],[80],[103] for examples. Since in many instances the behavior of this time-dependent front provides some useful information in understanding the underlying physics, front tracking is of great importance in fluid dynamics and in other branchs of research that study the nature of fronts in various physical situations.

In this thesis, we consider a simplified model for such an investigation in which viscosity, heat conduction, and other physical aspects relating to the microscopic structure of fronts are ignored, and the equations that describe front motion are a nonlinear hyperbolic system of conservation laws, perhaps with source terms

$$\frac{\partial u}{\partial t} + \sum_{j=1}^{N} \frac{\partial}{\partial x_j} f_j(u) = \psi(u) \tag{1.1}$$

where N is the number of spatial dimensions. We develop and study a simple front tracking approach that follows the discontinuous solutions explicitly for this model system in both one ($N = 1$) and two space dimensions ($N = 2$). This gives us a good method for modeling the propagation of fronts, and so is an important step toward studying problems involving sharp fronts.

Here the system we consider has m equations, so $u \in \mathbb{R}^m$. The homogeneous system with $\psi(u) = 0$ in (1.1) is assumed to be hyperbolic in the sense that the Jacobian matrix of any linear combination of the flux functions $f_j(u)$, *i.e.*,

$$\frac{\partial}{\partial u}\left(\sum_{j=1}^{N} \alpha_j f_j(u)\right) \qquad \text{for arbitrary} \quad \alpha_j \in \mathbb{R},$$

is assumed to have real eigenvalues and a complete set of eigenvectors for each physically relevant value of the conservative variables u. This is true, for example, for the Euler equations of gas dynamics which we use as our model system (see Sections 3.2 and 7.2). Other examples of practical interest may be found in [32] and [108].

Source terms of the form $\psi(u)$ can arise in various ways. Geometric source terms arise if a system in more than one space dimension is reduced to a one-dimensional problem using symmetry (*e.g.*, radially symmetric flow) or by assuming that the cross-sectional flow is homogeneous, as in the "quasi one-dimensional nozzle" problem discussed in Section 5.2.1. In another instance, gravitational source terms appear if gravity (which acts as a body force in the system) is of importance in the simulation, for example, in the Rayleigh-Taylor problem examined in Section 9.2.2.

Source terms that are more difficult to handle arise in the study of nonequilibrium or chemically reacting flows (*e.g.*, in combustion problems). Here the density is replaced by several different density functions, one for each chemical species, and the fluid equations are coupled to source terms for the production and consumption of individual species. Such problems are often complicated by the fact that the time scale of the chemical reactions may be orders of magnitude faster than the time scale for the fluid dynamics. Shock waves in the flow may be coupled with thin reaction zones in which mesh refinement is required in order to efficiently model the flow. A model system of this form is discussed in Section 5.2.2.

A number of front tracking algorithms have been proposed in the past and used to study front propagation (see Section 1.3). Their results show that the front tracking algorithm is an efficient way to compute flows involving discontinuities, but at a price of complicating the methods. A list of difficulties that need to be overcome in the front tracking algorithm is given in [17].

Here our work is different in that we use a conservative finite volume method based on the *large time step* wave propagation approach[61] to overcome a major difficulty associated with the limit on the time step in the presence of small cells created by the tracked front cutting through the grid, while maintaining conservation of the algorithm. In addition, we have investigated a variety of approaches to obtain high resolution in the grid cells near the tracked interface and have done extensive tests of accuracy and stability. Various approaches to propagating the front have also been studied and compared. This work of analyzing the algorithm is one of the main features of this thesis. In fact, doing so gives us a solid base in understanding solutions obtained from this algorithm, and this helps us understand the physics when the algorithm is employed in various applications.

1.2 Our Approach – thesis work

The basic idea of our front tracking algorithm is quite simple. We choose a uniform underlying grid with some grid cells subdivided by tracked interfaces, made up of moving points in one space dimension and curves in two space dimensions, approximately aligned with discontinuities in the flow field. In each time step, we introduce a new set of interfaces that are approximately aligned with the expected locations of discontinuities in the solution at the end of the time step. A high resolution finite volume method is then applied on the resulting nonuniform grid to update the cell values. If we have chosen the new interface locations well, the resulting solution will remain sharp and be smooth away from these new interfaces. The old interfaces can then be eliminated by recombining the adjacent cells.

Figure 1.1 shows a typical grid system for our front tracking algorithm. In Figure 1.1a, we show a one-dimensional grid where moving points are introduced for the discontinuities, and are inserted into an underlying uniform grid as grid interfaces. In Figure 1.1b, we show a two-dimensional grid where piecewise linear curves are introduced for the discontinuities, and are inserted into the grid.

To advance the tracked interfaces from the current time step to the next, we solve a one-dimensional Riemann problem at each tracked interface using the values from the adjacent cells as data, and follow strong waves to determine the new locations at the end of a time step. In one space dimension, this can be accomplished quite easily as illustrated in Figure 1.1a where $\hat{x}_\xi = x_\xi + \lambda_p k$ is the new location of the old tracked point x_ξ with strong wave speed λ_p (obtained from solving the Riemann problem at x_ξ) over a time step

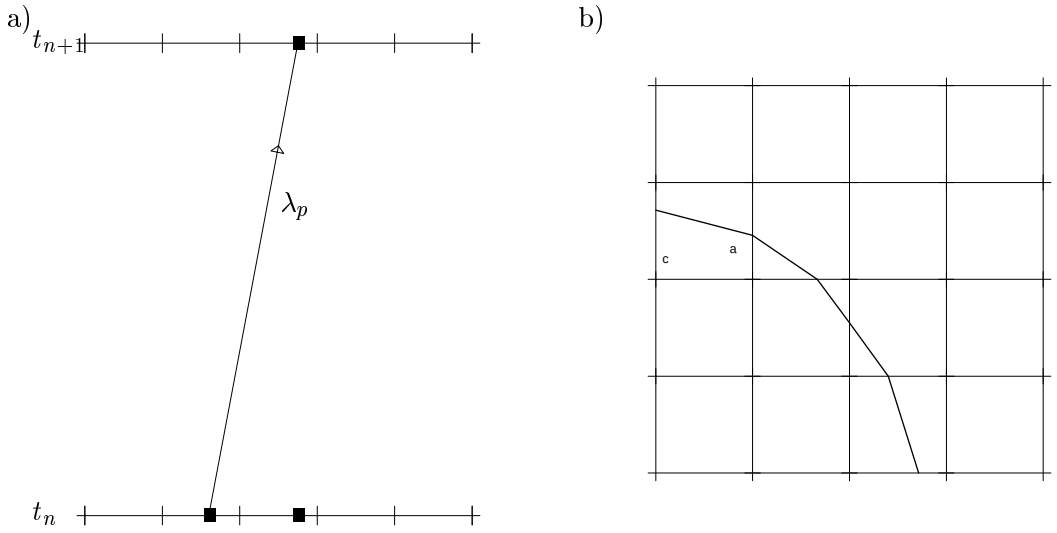


Figure 1.1: A typical grid system for our front tracking algorithm. a) In one space dimension, moving points are introduced for the discontinuities, and are inserted into the underlying uniform grid as grid interfaces. b) In two space dimensions, piecewise linear curves are introduced for the discontinuities, and are inserted into the underlying grid.

k. This new interface location $\hat{x}_\xi$ is then inserted into the grid. This one-dimensional front tracking algorithm is discussed in more detail in Chapter 3.

In two space dimensions, the new interface locations can be chosen by a technique analogous to our one-dimensional approach. Since each tracked interface is the boundary between two cells, we can use the values from the adjacent cells as data to solve one-dimensional Riemann problems in directions normal and tangential to each interface. We expect the solution to this Riemann problem to consist of only one strong wave, corresponding to the shock or interface (contact discontinuity or slip line) being tracked, and other weaker waves. Since we want the new tracked interfaces to form a continuous piecewise linear curve, as shown in Figure 1.1b, we need to use the solutions of neighboring Riemann problems in some coordinated manner to determine the new interfaces. There are various ways that this can be done *via* some sort of curve fitting through points determined by the strong waves from the Riemann solutions. In Chapter 7, we present one simple approach in more detail. Our basic philosophy of tracking, however, gives us some flexibility on this score – we do not view the interface we introduce as being the definitive location of a tracked front, but rather as a grid interface that is sufficiently well located and aligned that the solution can be well captured on the resulting grid.

Once the new grid is constructed, the solution is then advanced using a fully conservative shock capturing method. This method must be able to deal with the irregular cells near the tracked interface. In particular, we must maintain stability even if some of these cells are very small relative to the underlying mesh size used to determine the time step. We also hope to maintain second order accuracy in the smooth flow on either side. This is accomplished using a high resolution method based on the large time step wave propagation approach, developed by LeVeque[60],[61] (see also Chapters 2 and 6). The main idea is that waves arising from the solution of Riemann problems at the cell boundaries are propagated the appropriate distance determined by the wave speed and time step, and used to update

cell averages in any grid cell they encounter. The wave may affect more than one cell if the neighboring cell is very small. In this manner the stencil of the method adjusts automatically so that the CFL (Courant-Friedrich-Lewy) condition is always satisfied regardless of the configuration of the grid.

Source terms in the equations are currently handled using a Strang splitting[97]. The source terms are used to solve the ordinary differential equations $u_t = \psi(u)$ in each grid cell over a half time step. The homogeneous conservation laws are then solved using front tracking with a full time step. Another half time step is then taken with the ODEs (see Section 5.2).

Our goals here are to study the feasibility of this front-tracking procedure and explore various finite volume approaches on the resulting nonuniform grid. We avoid stability problems in the presence of small cells and achieve high resolution even in cells near the tracked interfaces in a conservative manner. In the current formulation, we only consider fronts with simple geometry. The approach we take here could alternatively be incorporated into a more complicated algorithm for complex geometry. However, even this simple form of front tracking can be very useful for certain classes of problems, as some of the examples in this thesis illustrate.

1.3 Other Approaches – overview

A wide variety of approaches have been used over the years to develop shock tracking or interface tracking methods. We will only mention a few of the main ideas and how they relate to our method. A concise survey of several approaches for the two-dimensional problem is given by Hyman[54] (see also Oran and Boris[77]).

With many methods, the conservation laws are solved separately on each side of the discontinuity using a method designed for smooth flow, while the shock or interface is handled in a different manner using the Rankine-Hugoniot jump conditions. This is the case, for example, with the method pioneered by Moretti[73]. Similar methods have been used by others, *e.g.*, Di Giacinto and Valorani[34], and Salas[91]. These methods are typically not conservative, which may be a problem if other shocks are present that are not being tracked. However, very nice results have been obtained for problems with sufficiently simple structure.

Mao[70],[71] has recently introduced a front tracking method in which two sets of data near the interface are constructed by extrapolating the data from each side to the other side. High resolution ENO (essentially non-oscillatory) methods are applied to the extrapolated values which now define smooth functions. The method is not exactly conservative at the interface, although errors in conservation appear to be small. Away from the interface the method is fully conservative.

Glimm and coworkers, *e.g.*, [17],[38],[42], have developed a very extensive set of tools for shock and interface tracking that have been successfully applied to a wide variety of problems. This package includes procedures to deal with complicated interactions of interfaces, Mach triple points, and other such structures in spite of the lack of conservation at the tracked front. The structure of solutions to multi-dimensional Riemann problems is used to determine the behavior of the solution near the intersection points. This complicated algorithm is capable of dealing with some very complex problems.

The approach taken here is fully conservative and based on high resolution shock capturing methods so that features not being tracked can still be accurately computed. The method is quite simple conceptually and algorithmically, although it would be complicated considerably by allowing the interaction of fronts.

We use a finite volume method on a grid consisting of a uniform fixed grid in which some cells have been subdivided by tracked interfaces. Potential problems with stability are avoided by the use of a "large time step" method. Another approach would be to eliminate the problematical small grid cells by merging them with adjacent cells, temporarily eliminating a "fixed" cell boundary in the process. This approach is used, for example, in [75] and [98]. However, this may be impossible to do if several tracked fronts fall within one fixed grid cell. Moreover, this seems to be unnecessary with our approach.

Swartz and Wendroff[98] also consider a method in which the flow is entirely represented by a collection of discontinuities, all of which are explicitly tracked. Following Dafermos[26], a piecewise linear equation of state is used to ensure that only discontinuities arise in solutions to Riemann problems. A similar approach was investigated by Hedstrom[51] and has more recently been adopted by Risebro and Tveito[87],[86]. Since every collision must be explicitly handled by solving a Riemann problem, and the collision of two waves typically gives rise to m new waves (for a system of m equations), this can clearly lead to an explosion of information if $m > 2$, as in the Euler equations. (Although Wendroff[105] has studied this method for a problem arising in chromatography and shows that for this special system the number of waves remains bounded.) In general, at some point weak waves must be suppressed in order to limit the amount of information retained, leading to a loss of conservation. Another problem is that smooth flow is not represented with high order accuracy. Finally, there is the obvious difficulty of extending such methods to more than one dimension.

Our method is perhaps closest to that of Chern and Colella[16]. They also use a conservative method on a uniform grid, with some grid cells subdivided by the tracked front. They avoid stability problems in small partial cells by a "flux redistribution" algorithm that modifies fluxes at the boundaries of these and neighboring cells in such a way that stability is restored while conservation is maintained. Our use of the wave propagation algorithm described in Chapters 2 and 6 has the same effect. In addition, we believe it to be more solidly based on the correct physical behavior of the waves, and more amenable to high order extensions and theoretical analysis.

Another way that our algorithm differs from that of Chern and Colella is that we explicitly track the discontinuities, while they use a fractional marker volume (or "volume of fluid") approach in which they keep track of the volume of fluid in each cell that is "behind" the front and then dynamically reconstruct the front in each time step. This approach is used in many interface methods, *e.g.*, [18],[46],[58],[76]. In one space dimension there is little difference in these approaches – determining the front location is trivial from the fractional volume but on the other hand tracking the points explicitly is also quite simple. In two space dimensions one must seriously weigh the tradeoffs. Hyman[54] discusses some of the pros and cons. Our current two-dimensional implementation is based on explicitly tracking the global interface, but the stable high resolution numerical methods we are developing could be equally well used in the cells formed by locally reconstructing the front from fractional volume information.

Another quite different approach to representing interface is proposed by Mulder, Osher,

and Sethian[74],[78]. They represent the front as a level set of an auxiliary function that satisfies an equation of Hamilton-Jacobi type. This seems to be a promising approach since it handles complicated changes of geometry quite easily.

Finally, we want to mention that there are also a number of shock capturing approaches that are capable of improving the resolution of discontinuities. Methods of this type include the self-adjusting grid methods of Harten and Hyman[49], and the ENO method with subcell resolution of Harten[47].

1.4 Outline

This dissertation is divided into three parts. In Parts I and II, we develop, analyze, and apply the front tracking algorithm for nonlinear hyperbolic systems of conservation laws in one and two space dimensions, respectively. In Part III, we consider problems arising from porous media flow where the hyperbolic conservation law is coupled with an elliptic partial differential equation in order to correctly model the problem.

The organization for Parts I and II is quite similar. In Chapters 2 and 6, we begin describing numerical methods that can be used on a nonuniform grid generated by the front tracking algorithm. In Chapters 3 and 7, we describe the front tracking algorithm, introduce our model system (the Euler equations of gas dynamics), discuss the implementation of boundary conditions for this system, and present preliminary results obtained using this algorithm for this model system. In Chapters 4 and 8, we perform error estimation and study accuracy and stability of the algorithm. Several approaches that can be used to improve upon the algorithm have been discussed also. In Chapters 5 and 9, we present more numerical results for various applications arising in gas dynamics.

In Part III, Chapter 10, we describe an algorithm that can be used to solve a coupled system of elliptic and hyperbolic partial differential equations arising in oil reservoir simulation. Some preliminary results for this problem in both one and two space dimensions are also shown. Finally, in Chapter 11, we summarize the thesis work and outline future research.

Part I

One Space Dimension

Chapter 2

FINITE VOLUME WAVE PROPAGATION METHODS

We begin our discussion by describing numerical methods that can be used to compute the smooth solution for the homogeneous conservation laws

$$u_t + f(u)_x = 0. \tag{2.1}$$

Although these methods are related to various upwind or flux-limiter methods[20],[50] that have been widely used for conservation laws, the formulation is somewhat different. We use a wave-propagation viewpoint that allows us to interface the method easily with front tracking and maintain stability even when very small cells are created.

2.1 Preliminaries

We describe the methods on a general nonuniform grid with grid spacing $h_j = x_{j+1} - x_j$. We use a finite-volume formulation in which the value U_j^n approximates the cell average of the solution over the grid cell $[x_j, x_{j+1}]$ at time t_n,

$$U_j^n \approx \frac{1}{h_j} \int_{x_j}^{x_{j+1}} u(x, t_n)\, dx.$$

The time step is denoted by k. Note that the grid may vary from step to step but the method involves only two time levels, so this presents no difficulty.

The methods we use are based on solving Riemann problems at each interface. A *Riemann problem* consists of the original conservation laws (2.1) together with piecewise constant data u_l and u_r to the left and right of a single discontinuity. The Riemann problem for various systems of conservation laws has been extensively studied and the exact solution can often be found [15],[57],[63]. Under certain conditions, satisfied for the Euler equations, for example, the solution is a similarity solution (depending on x/t alone) that consists of m waves for a system of m equations. Each wave is a shock wave, rarefaction wave, or contact discontinuity.

At each interface x_j, we solve the Riemann problem with data U_{j-1}^n and U_j^n. Rather than computing the exact solution to the Riemann problem, which can be done but is rather expensive, we use an approximate solver developed by Roe[88] at most interfaces. This is much more efficient to compute than the exact Riemann solution and in smooth regions of the flow provides a very accurate approximation. Only at front collision points do we use the exact Riemann solver so that the nonlinear interaction is accurately computed (see Section 3.1 for further discussions).

Roe's approximate Riemann solver replaces the nonlinear equations (2.1) with data u_l and u_r by a linear system

$$u_t + \hat{A}(u_l, u_r) u_x = 0. \tag{2.2}$$

The matrix $\hat{A}(u_l, u_r)$ is chosen to have the following properties:

$$\begin{aligned} &i) \quad \hat{A}(u_l, u_r)(u_r - u_l) = f(u_r) - f(u_l) \\ &ii) \quad \hat{A}(u_l, u_r) \text{ is diagonalizable with real eigenvalues} \\ &iii) \quad \hat{A}(u_l, u_r) \to f'(\bar{u}) \text{ smoothly as } u_l, u_r \to \bar{u}. \end{aligned} \tag{2.3}$$

Such matrices have been derived for several systems of practical interest. For the Euler equations with a γ-law gas, the form of the matrix is given by Roe[88].

The solution of the linear system (2.2) is a similarity solution that consists of m discontinuities propagating at constant speeds. The jump across each discontinuity is an eigenvector of the matrix $\hat{A}$, and the propagation speed is the corresponding eigenvalue. We thus have

$$u_r - u_l = \sum_{p=1}^{m} r_p, \tag{2.4}$$

where $r_p \in \mathbb{R}^m$ is an eigenvector of $\hat{A}$,

$$\hat{A} r_p = \lambda_p r_p, \quad p = 1, 2, \ldots, m.$$

Wave propagation methods are based on using these propagating discontinuities to update the cell averages in the cells neighboring each interface.

2.2 Godunov Method

To begin, we assume that waves resulting from the Riemann problems affect only the cells adjacent to the discontinuity during the time step. This requires that the Courant number be less than 1. The Courant number ν is defined by

$$\nu = \frac{k}{h_{\min}} \max_{p,j} |\lambda_{pj}| \tag{2.5}$$

where

$$h_{\min} = \min_j h_j$$

and λ_{pj} represents the pth eigenvalue obtained from the Riemann problem at x_j. Note that $k\lambda_{pj}$ is the distance a wave propagates during the time step. If $\lambda_{pj} < 0$, then we need $k|\lambda_{pj}| < h_{j-1}$, while if $\lambda_{pj} > 0$ we need $k\lambda_{pj} < h_j$ in order that the wave stays within the adjacent cell. Condition (2.5) is sufficient to guarantee this.

A first order accurate version of the wave propagation method is then equivalent to Godunov's method, with the Roe Riemann solver, on a nonuniform grid. That is to say, we solve the Riemann problems at each interface over a time step of length k and then average the resulting solution over the grid cells to obtain U^{n+1}. By computing the effect of each wave on the cell average, we obtain the following wave-propagation form of the algorithm:

Algorithm 2.1

Initialize $U_j^{n+1} := U_j^n$ for all j
For each j do begin
Solve the Riemann problem at x_j based on data $U_{j-1}^n,\ U_j^n$ to obtain
jumps r_{pj} and speeds λ_{pj} $(p = 1, 2, \ldots, m)$
For $p = 1, 2, \ldots, m$ do begin
Update the cell average to the left or right of the interface
depending on the speed:
If $\lambda_{pj} < 0$ then $i := j - 1$ else $i := j$
$U_i^{n+1} := U_i^{n+1} - \lambda_{pj} k r_{pj}/h_i$
Apply second order corrections if desired (See Algorithm 2.2)
end
end

The second order corrections will be discussed below. We can rewrite this method as a standard finite difference method in conservation form if we look at the total contribution to each grid cell. We find that

$$\begin{aligned} U_j^{n+1} &= U_j^n - \sum_{\lambda_{pj}>0} \frac{\lambda_{pj}k}{h_j} r_{pj} - \sum_{\lambda_{p,j+1}<0} \frac{\lambda_{p,j+1}k}{h_j} r_{p,j+1} \\ &= U_j^n - \frac{k}{h_j}[F(U_j^n, U_{j+1}^n) - F(U_{j-1}^n, U_j^n)] \end{aligned}$$

where the numerical flux function F is given by

$$F(U_j^n, U_{j+1}^n) = f(U_j^n) + \sum_{\lambda_{p,j+1}<0} \lambda_{p,j+1} r_{p,j+1}$$

and

$$\begin{aligned} F(U_{j-1}^n, U_j^n) &= f(U_{j-1}^n) + \sum_{\lambda_{pj}<0} \lambda_{pj} r_{pj} \\ &= f(U_j^n) - \sum_{\lambda_{pj}>0} \lambda_{pj} r_{pj}. \end{aligned} \tag{2.6}$$

The last equality follows from Property (2.3i). This property guarantees that the wave-propagation method is conservative.

The advantage of using the wave propagation form rather than the more traditional flux differencing form is that the method can then be easily extended to the case where the Courant number is larger than 1. In this case, waves propagate through more than one grid cell. The flux differencing formula based on the above fluxes would then lead to an unstable method, since the ratio $\lambda_{pj}k/h_j$ would be larger than 1 in magnitude for some waves. On the other hand, using the wave propagation approach allows us to extend the method easily in a stable manner. Note that it is also possible to write a flux differencing method for this larger Courant number case, but the fluxes are more complicated[59].

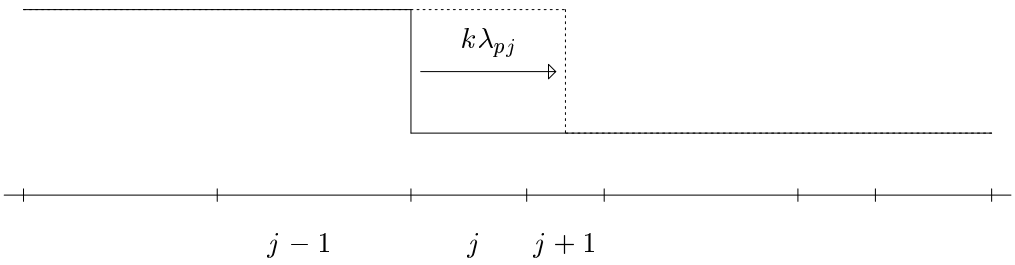


Figure 2.1: Wave propagation in the case $k\lambda_{pj} > h_j$. The wave propagates entirely through one cell and part way through the neighbor.

For example, if $k|\lambda_{pj}| > h_j$ at some point in the algorithm, then the corresponding wave should update more than one cell average, as shown in Figure 2.1. In this figure, U_j^{n+1} is updated by the entire jump r_{pj},

$$U_j^{n+1} := U_j^{n+1} - r_{pj},$$

while U_{j+1}^{n+1} is updated by

$$U_{j+1}^{n+1} := U_{j+1}^{n+1} - \left(\frac{k\lambda_{pj} - h_j}{h_{j+1}}\right) r_{pj}.$$

The method remains conservative with this modification. This "large time step" version of Godunov's method is discussed in some detail in [60],[62]. Regarding stability, we note that for a scalar nonlinear conservation law the method is total variation diminishing (TVD) and hence is stable and convergent[59]. Also, for a linear system of conservation laws the method reduces to a scalar large time step method on each characteristic field and again is stable. For nonlinear systems of equations, some oscillation problems have been observed when large Courant numbers on uniform grids are used in the context of shock capturing[60]. In this case, waves pass through several grid cells and the linearization of the nonlinear interactions apparently results in difficulties.

However, in the context of front tracking, where the Courant number is large only due to occasional small cells and we are capturing smooth flow, we have not observed stability problems for most calculations. In one example presented in Section 5.1.2, the Woodward-Colella blast wave interaction problem, we have experienced difficulties due to negative pressures using the linear wave interactions. It is an extreme case, however, in which a strong rarefaction wave overtakes a shock that has a very low pressure in front of it. The pressure becomes negative when the rarefaction wave passes through the blast wave and enters the low pressure region.

2.3 High Resolution Godunov Method

We now extend the method to a high resolution method, *i.e.*, a method that achieves second order accuracy on smooth flows (except perhaps near extrema) and also avoids oscillations near discontinuities. The approach we use is similar to the MUSCL (standing for "Monotonic Upstream-centered Scheme for Conservation Laws") approach of van Leer[104]

in that we introduce piecewise linear approximations to the solution in place of the piecewise constant functions used in Godunov's method, but the form of the method is quite different and allows easy extension to the case where the Courant number is larger than 1. More details of this approach can be found in [61],[62].

We begin our method by solving the Riemann problems as before, using the piecewise constant data. The resulting jumps r_{pj} are then used to obtain slope information in each characteristic family. Let

$$h_{j-1/2} = \frac{1}{2}(h_{j-1} + h_j)$$

be the distances between cell centers. Note that

$$\begin{aligned} \sum_{p=1}^{m} r_{pj}/h_{j-1/2} &= (U_j^n - U_{j-1}^n)/h_{j-1/2} \\ &= u_x(x_j, t_n) + O(h). \end{aligned}$$

So each component $r_{pj}/h_{j-1/2}$ is the contribution to the slope arising from the pth family. It is important to decompose the slope into components, since the waves in the different families propagate at different speeds and perhaps in different directions. Moreover, when we introduce slope limiting, we will do the limiting separately in each family. We wish to limit slopes near a discontinuity in order to avoid oscillations, but wish to do this in the family with the discontinuity without affecting accuracy in other families where the solution may be smooth.

We will use σ_{pj} to denote the slope used in the pth family over the cell $[x_j, x_{j+1}]$. The unlimited slope is taken to be

$$\sigma_{pj} = \begin{cases} r_{pj}/h_{j-1/2} & \text{if } \lambda_{pj} < 0 \\ r_{p,j+1}/h_{j+1/2} & \text{if } \lambda_{p,j+1} > 0. \end{cases} \tag{2.7}$$

To avoid oscillations, the slope σ_{pj} should be chosen based on a slope limiter. If we let $\sigma_{pj}^{(i)}$ be the ith component of σ_{pj} $(i = 1, 2, \ldots, m)$ and similarly let $r_{pj}^{(i)}$ be the ith component of r_{pj}, then we apply a slope limiter separately in each component, *i.e.*, we take

$$\sigma_{pj}^{(i)} = \phi(\theta_{pj}^{(i)}) \left(\frac{r_{pj}^{(i)}}{h_{j-1/2}} \right) \tag{2.8}$$

where ϕ is some limiter function applied to the slope ratio

$$\theta_{pj}^{(i)} = \frac{h_{j-1/2}\ r_{p,j+s}^{(i)}}{h_{j+s-1/2}\ r_{pj}^{(i)}}, \tag{2.9}$$

with $s = -sgn(\lambda_{pj})$. One simple choice of limiter is the "minmod" slope limiter ϕ, given by

$$\begin{aligned} \phi(\theta) &= \begin{cases} 0 & \text{if } \theta \le 0 \\ \theta & \text{if } 0 \le \theta \le 1 \\ 1 & \text{if } \theta \ge 1 \end{cases} \\ &= \max(0,\ \min(1, \theta)). \end{aligned} \tag{2.10}$$

Other choices of slopes can also be used, and a variety of limiter functions have been developed that work better than "minmod". Typical examples are the "superbee" limiter of Roe[89]:

$$\phi(\theta) = \max(0,\ \min(1, 2\theta),\ \min(\theta, 2)), \tag{2.11}$$

and the "MUSCL" limiter of van Leer[104]:

$$\phi(\theta) = \max(0,\ \min(2, 2\theta, (1+\theta)/2). \tag{2.12}$$

See Sweby[99] for a more general discussion of limiters.

This slope is used to modify the cell averages computed *via* the first order algorithm. The modification is accomplished by shifting a certain mass between cells in a conservative manner. The idea is best explained by considering the linear advection equation

$$u_t + au_x = 0 \tag{2.13}$$

on a grid with Courant number $\nu = ak/h_{\min} \leq 1$, $a > 0$. Godunov's method is then simply the first order upwind method

$$U_j^{n+1} = U_j^n - \frac{ak}{h_j}(U_j^n - U_{j-1}^n). \tag{2.14}$$

This can be interpreted as follows: view U_j^n as defining a piecewise constant function $\tilde{u}_j(x, t_n)$. Shift this function at the propagation speed a to obtain $\tilde{u}_j(x - ak, t_n)$. Now average this function over the grid cells to obtain

$$U_j^{n+1} = \frac{1}{h_j}\int_{x_j}^{x_{j+1}} \tilde{u}_j(x - ak, t_n)\, dx.$$

It is easy to verify that this gives (2.14). So each cell average is updated by the shaded area in Figure 2.2a divided by the cell length.

A natural way to extend this to second order accuracy is to replace the piecewise constant function by a piecewise linear function of the form

$$\tilde{u}_j(x, t_n) = \begin{cases} U_j^n + (x - x_{j+1/2})\sigma_j & x_j \leq x < x_{j+1} \\ 0 & \text{otherwise,} \end{cases} \tag{2.15}$$

with slopes σ_j on each cell as obtained, for example, from (2.7). For the scalar equation (2.13), this reduces to

$$\sigma_j = (U_{j+1}^n - U_j^n)/h_{j+1/2}. \tag{2.16}$$

We then shift this function at speed a and average onto the grid. We thus obtain U_j^{n+1} by updating U_j^n according to the shaded area of Figure 2.2b.

An easy way to accomplish this is to split the procedure into two pieces. In the first step we update cell averages using the piecewise constant wave as in Figure 2.2a (*i.e.*, we apply (2.14)), and in the second step we propagate the piecewise linear wave form shown

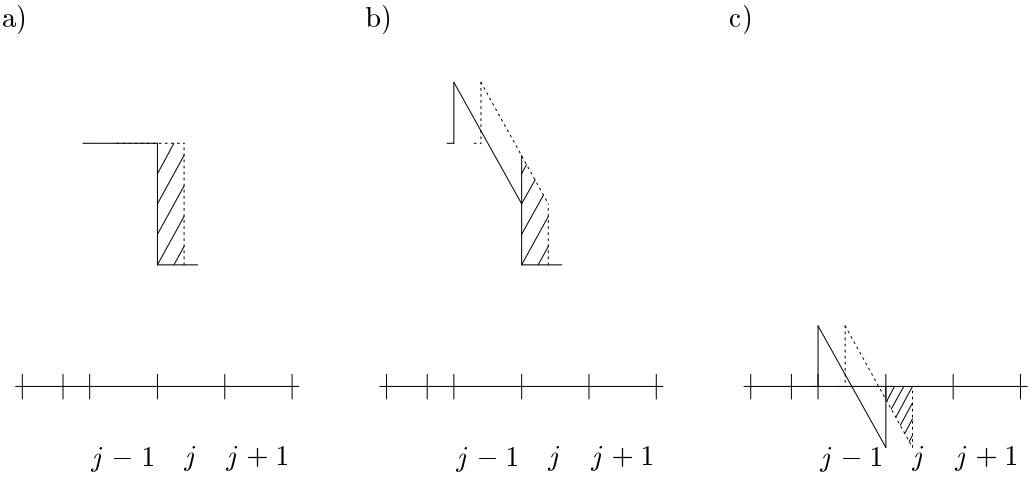


Figure 2.2: a) Propagation of the piecewise constant wave. b) Propagation of a piecewise linear wave form. c) Second order correction wave. The propagation shown in Figure b can be decomposed into propagation of the piecewise constant wave of Figure a together with propagation of this correction wave.

in Figure 2.2c, with zero mean value and slope σ_j over the cell. We then further update U_j^{n+1} by the shaded area in Figure 2.2c, *i.e.*, we set

$$U_j^{n+1} := U_j^{n+1} + \frac{ak}{2h_j}(h_{j-1} - ak)\sigma_{j-1}.$$

We also update U_{j-1}^{n+1} by the area of the portion of the correction wave that overlaps this cell,

$$U_{j-1}^{n+1} := U_{j-1}^{n+1} - \frac{ak}{2h_{j-1}}(h_{j-1} - ak)\sigma_{j-1}.$$

Conservation is maintained in this correction step with any choice of slopes since the above two corrections (weighted by cell size) sum to zero.

Of course, U_j^{n+1} will also be updated by the wave originating from $x_{j+1/2}$. When all of these updates are combined, we find that

$$U_j^{n+1} = U_j^n - \frac{ak}{h_j}(U_j^n - U_{j-1}^n) + \frac{ak}{2h_j}(h_{j-1} - ak)\sigma_{j-1} - \frac{ak}{2h_j}(h_j - ak)\sigma_j.$$

On a uniform grid with slopes (2.16), this reduces to the Lax-Wendroff method for the linear advection equation (2.13) and is second order accurate.

The extension to nonlinear systems is straightforward. We apply Algorithm 2.1 but now for each wave we also apply a correction step. In this algorithm we assume that the Courant number is less than 1, and so the correction step takes the form in Algorithm 2.2.

Algorithm 2.2 (Insert in Algorithm 2.1)

$$
\begin{aligned}
&\# \text{ Second order corrections:} \\
&\text{If } \lambda_{pj} < 0 \text{ then } i := j \text{ else } i := j-1 \\
&U_j^{n+1} := U_j^{n+1} + \frac{|\lambda_{pj}|k}{2h_j}(h_i - |\lambda_{pj}|k)\sigma_{pi} \\
&U_{j-1}^{n+1} := U_{j-1}^{n+1} - \frac{|\lambda_{pj}|k}{2h_{j-1}}(h_i - |\lambda_{pj}|k)\sigma_{pi}
\end{aligned}
$$

Note that this correction can be easily translated into the flux differencing framework. The first order flux $F(U_{j-1}^n, U_j^n)$ from (2.6) is simply replaced by

$$F_{j-1/2}^n = F(U_{j-1}^n, U_j^n) - \sum_{p=1}^{m} \frac{1}{2}|\lambda_{pj}|(h_{i(p)} - |\lambda_{pj}|k)\sigma_{p,i(p)} \tag{2.17}$$

where

$$i(p) = \begin{cases} j-1 & \text{if } \lambda_{pj} > 0 \\ j & \text{if } \lambda_{pj} \le 0. \end{cases}$$

Notice that the flux now depends on four neighboring points rather than two, so we use the abbreviated notation $F_{j-1/2}^n$. This flux formulation will be useful in conjunction with grid refinement in Section 5.1.

For a scalar nonlinear problem on a uniform grid with slopes (2.7), this again reduces to a form of the Lax-Wendroff method and can easily be verified to be second order accurate. For a nonlinear system of equations on a uniform grid, this method is quite comparable to other slope limiter of flux limiter methods and yields similar high quality results.

The method can also be generalized quite easily to nonuniform grids with Courant number greater than one by appropriately averaging the correction wave onto whatever grid cells it overlaps. Algorithmic details may be found in [62].

Chapter 3

FRONT TRACKING ALGORITHM

Having described the numerical methods that can be used on the nonuniform grid for the homogeneous conservation laws (2.1), we now discuss the front tracking algorithm for this system. As we will see from the discussion, this algorithm is very simple and robust. Moreover, it is a conservative algorithm with no stringent time step limitations in the presence of small cells created by the tracked interfaces cutting through the grid. One example will be given here for our model system, the Euler equations of gas dynamics, to demonstrate the effectiveness of the algorithm. The implementation of boundary conditions for this model system will also be discussed.

3.1 Algorithm

Our grid consists of two parts. We choose a uniform underlying grid with mesh size h that remains fixed for all time, and we also introduce tracked points which vary from step to step for discontinuities in the flow field. These tracked points subdivide some regular cells into two or more subcells, creating some irregular cells. We then view the union of the regular cells and irregular cells as our global grid (see Figure 3.1). In each grid cell the cell average is denoted by U_j^n.

In each time step our front tracking algorithm consists of the following steps:

Algorithm 3.1

1) Determine the size of the next time step and the location of the tracked points at the next time step.

2) Modify the current grid by inserting these new tracked points. Some cells will be subdivided and the values in each subcell must be initialized.

3) Take a time step on this nonuniform grid using a finite volume method described in Chapter 2 to update the cell averages.

4) Delete the old tracked points from the previous time step. Some subcells will be combined and a value in the combined cell must be determined from the subcell values.

Before describing each of these steps in more detail, we first discuss some possible approaches to setting up the data structure. One possibility is to use a doubly linked list for the entire grid (see [1] or [55] for more information on the use of linked lists). Each grid cell is an element of this list, with pointers to the previous and next grid cells. With this data structure, it is easy to insert and delete grid points and the distinction between regular and irregular cells disappears. This is reasonable in Step 3 of our algorithm, where

little distinction is made between regular and irregular cells, although we will see that we must be careful in our choice of slopes near tracked points. We also need to keep track of which points must be deleted in Step 4. For these reasons we would also maintain a flag for each point that tells whether it is a regular point, an old tracked point, or a new tracked point.

The use of a doubly linked list does not extend very well to two space dimensions. Another more general approach is to use a standard representation for the fixed grid together with a flag for each grid cell that indicates whether the grid cell is subdivided by one or more tracked points. For subdivided cells, this flag can be a pointer to another data structure containing information on each subcell.

This latter data structure also interfaces more easily with the adaptive mesh refinement algorithm we use, and so we have taken this approach in our code.

We now discuss each step of Algorithm 3.1 in more detail.

Step 1: We begin our algorithm by solving the Riemann problem at each interface and obtain the resulting jumps r_{pj} and speeds λ_{pj}. Then at each interface we check each jump r_{pj} to see if it should be tracked. This can be done by checking, for example, if the max-norm of r_{pj} is greater than some prescribed tolerance ε, or if the jump in some physically meaningful variable (*e.g.*, density or entropy) is greater than the tolerance ε. The choice of the checking criterion and tolerance ε for determining the tracked waves may depend on the specific problem and should be adjusted accordingly. In order to capture the shock formation, the jumps have to be checked at regular interfaces as well as at the tracked interfaces so that new tracked points can be introduced. By examining the jumps at tracked interfaces, decaying shocks can also be detected and hence ignored.

It should be noted that only waves corresponding to the physically relevant discontinuities should be tracked, *i.e.*, shocks or contact discontinuities. Although rarefaction waves are also approximated by discontinuities in the Roe Riemann solver, we want them to be smeared rather than remaining sharp and so they should not be tracked even if their strength is greater than ε. Moreover, due to this rarefaction wave approximation, we may obtain an entropy-violating solution if the rarefaction wave is a transonic one. This entropy violation can be fixed in various ways, for example, by replacing the single entropy-violating discontinuity by two discontinuities traveling in opposite directions[49].

Before entering the front tracking algorithm, we have some basic time step k. In order to avoid the interaction of tracked waves during this time step, we adjust our time step if needed. It will be adjusted in such a way that the collision of two tracked waves occurs exactly at the end of a time step (see Figure 3.1). This can be accomplished quite easily. Assume that there are two tracked waves, one originating from x_η with speed $\lambda_{p\eta}$ and the other from x_ξ with speed $\lambda_{q\xi}$. Here p, q are the wave families, and η, ξ are the indices of the cell interfaces. Let $\hat{x}_\eta$, $\hat{x}_\xi$ be the new locations of x_η, x_ξ under the current time step, *i.e.*,

$$\begin{aligned} \hat{x}_\eta &= x_\eta + \lambda_{p\eta} k, \\ \hat{x}_\xi &= x_\xi + \lambda_{q\xi} k. \end{aligned}$$

If no interaction occurs, the product of the two relative distances $(\hat{x}_\eta - \hat{x}_\xi)(x_\eta - x_\xi)$ will be greater than zero for each pair of tracked waves. If interaction happens, the time step for

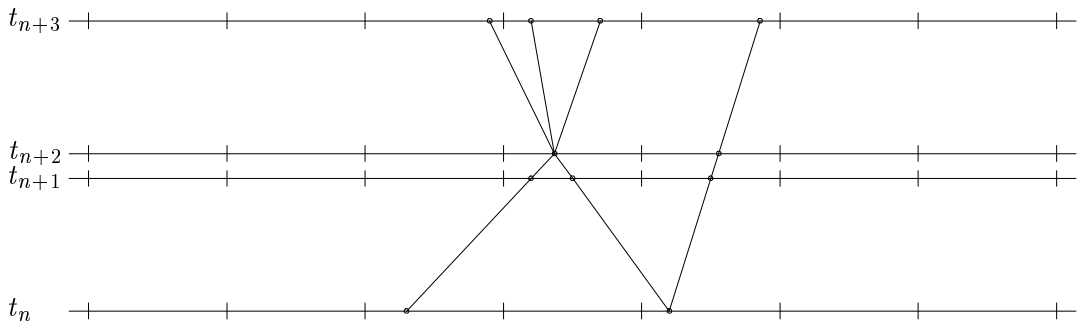


Figure 3.1: A typical grid in the x-t plane when front-tracking is used to model the collision of two shocks with the Euler equations. The uniform grid is augmented by cell interfaces at the discontinuities. The time step is adjusted so that the shock collision is correctly resolved.

this pair of interaction can be computed by

$$k = (x_\eta - x_\xi)/(\lambda_{q\xi} - \lambda_{p\eta}).$$

We choose the time step to be the minimum of all collision times found by checking adjacent tracked waves.

If collision occurs, in the next time step the approximate Riemann solver is replaced by the exact Riemann solver at the collision point to insure that the resulting waves in the next time step are well resolved. By choosing the time step in this way and using the exact Riemann solver, we guarantee that the collision of two tracked waves is always handled correctly.

Step 2: After choosing the time step k, we can compute the locations of each tracked wave at the end of the time step. Some of these locations may coincide if two waves collide, or if the new locations are exactly at the old grid interfaces. Also, some waves may pass outside of our computational domain at an outflow boundary. For each distinct wave location in the domain, we insert a new cell interface into our old grid. Each new point subdivides some cell into two subcells. We must assign a cell value on each of these subcells (see Figure 3.2 for an example). The simplest approach is to assign the previous cell value to each subcell. It would be preferable to use some form of interpolation to determine more accurate values on these cells. However, doing so would change the solutions to neighboring Riemann problems and perhaps the speed of the tracked waves. The location of the point we are inserting might therefore be incorrect. For this reason we use the simpler approach.

Step 3: Once the new grid is constructed, the cell average values U_j^n are then updated by applying the finite volume wave propagation method described in the previous chapter on the resulting nonuniform grid, see Figure 3.3 for illustration. Since the new grid has been chosen carefully so that all the strong waves are propagated exactly to cell boundaries, there is no smearing of the tracked waves during the averaging process. Smooth flow is captured as usual. Note that during this propagation process, all waves are propagated independently, and in principle no distinction need be made between tracked points and ordinary grid boundaries. Near tracked points, waves may propagate through several cells due to the fact that we have created small subcells. A consequence of this is that waves pass

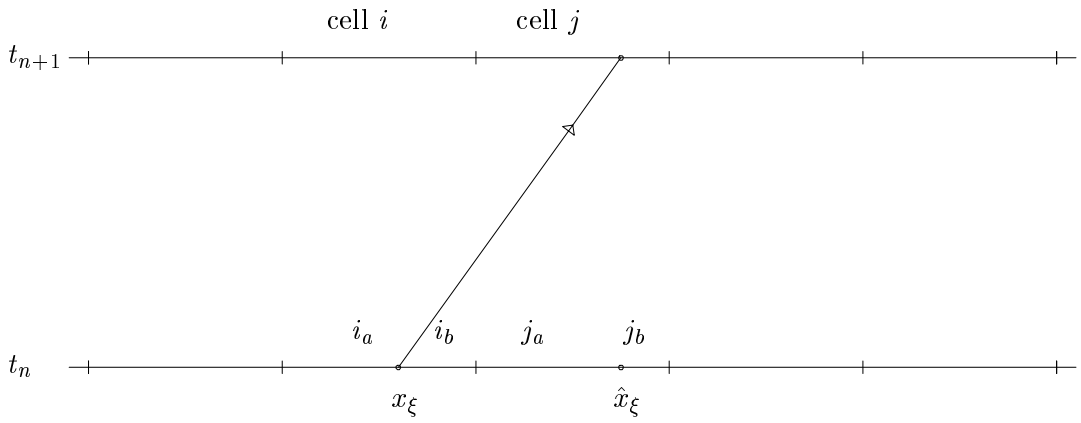


Figure 3.2: A shock propagating from cell i to cell $j = i + 1$ leads to a subdivision of cells i and j. In time step n we split cell j in two, setting $U^n_{j_a} = U^n_{j_b} = U^n_j$. In time step $n + 1$ we eliminate the old tracked point in cell i using (3.1).

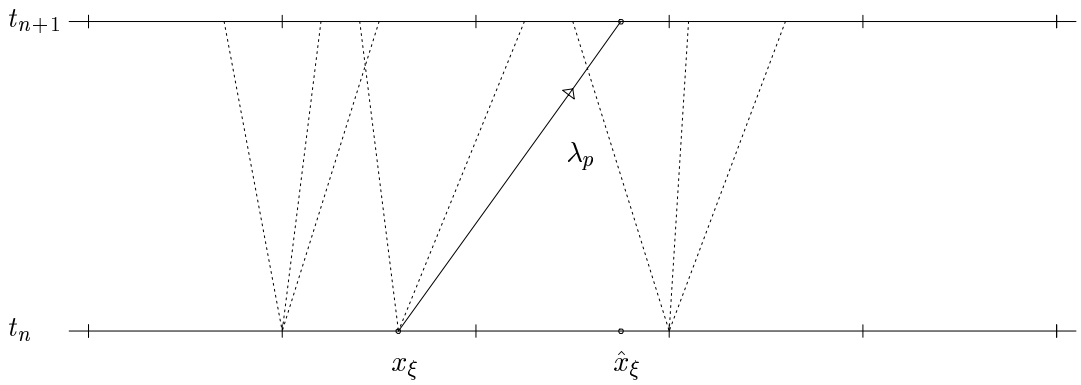


Figure 3.3: Wave propagation in step 3. Each wave is propagated independently, and for waves passing through each other the interaction is linearized. Note that the tracked wave is propagated exactly to the new cell boundary $\hat{x}_\xi$ introduced in Step 2. (The solid lines shown in the figure represent the tracked waves, while the dashed lines represent the weak waves.)

through one another as they would in a linear equation, without undergoing the nonlinear interaction that should occur. For weak waves, this is a good approximation, as described in [60]. For the interaction between a strong tracked wave and the weak waves arising from the nearby Riemann problems, this linearization is less valid. In the next chapter, we investigate the error introduced by this procedure, and discuss one possible approach that may improve the accuracy.

Step 4: We now delete the old tracked points from the current grid. This corresponds to merging two subcells into one, and the cell value in the combined cell is calculated by the appropriate weighted combination of these two deleted subcells to maintain the correct cell average. For example, in Figure 3.2, the old tracked point x_ξ is deleted from the ith regular cell, and so the ith cell average after deletion becomes

$$U_i^{n+1} = \frac{x_\xi - x_i}{h} U_{i_a}^{n+1} + \frac{x_{i+1} - x_\xi}{h} U_{i_b}^{n+1} \tag{3.1}$$

where $U_{i_a}^{n+1}$, $U_{i_b}^{n+1}$ are the cell averages in the first and second subcell of the ith cell respectively, and h is the underlying fixed mesh size.

3.2 The Euler Equations and Boundary Conditions

Before presenting numerical results with this front tracking algorithm, we pause to introduce the Euler equations of gas dynamics and discuss the implementation of boundary conditions for this system.

The inviscid Euler equations of gas dynamics in one space dimension take the form

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho v \\ \rho E \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho v \\ \rho v^2 + p \\ (\rho E + p) v \end{pmatrix} = 0 \tag{3.2}$$

where ρ, v, p, E are the density, velocity, pressure, and total energy of the gas per unit mass, respectively. We assume a γ-law gas, in which the internal energy satisfies $e = \frac{1}{\gamma - 1} p/\rho$, where γ is the ratio of specific heats ($\gamma \cong 1.4$ for air). Then the total energy of the gas per unit mass is $E = e + \frac{1}{2} v^2$. The three components of equations (3.2) express the conservation of mass, momentum, and energy, respectively[25].

Outflow boundary conditions are easily achieved with the wave propagation approach by simply ignoring waves once they leave the computational domain, and by not introducing any new waves at the boundary.

For periodic boundaries, we now allow all the outgoing waves which leave at one boundary to return to the domain at the other boundary with the same speed λ_{pj}, jump r_{pj} and slope σ_{pj}, say for the pth wave from the x_j interface. We can think of these "incoming" waves as coming from the solution of the Riemann problems on an extended grid with periodically extended data.

At a solid wall boundary, say at $x = 0$, we have the no-flow boundary condition

$$v(0, t) = 0.$$

This boundary can be treated as a line of symmetry. If we reflect our grid near the boundary to the region $x < 0$, we can assign grid values in the reflected cells using

$$U_{-j}^n = \mathcal{R}(U_j^n), \qquad j = 1, 2, \cdots$$

where $\mathcal{R}$ represents the operator that negates the second component of U (the velocity) while leaving the first and third components (density and energy) unchanged. Applying the algorithm over a slightly extended domain simulates the solid wall boundary condition.

Alternatively, we can avoid extending the grid if we note that each incoming wave (a wave entering our true computational domain from $x < 0$) can be viewed as the reflection of an outgoing wave (a wave crossing $x = 0$ with negative speed). This is illustrated in [61],[62]. It is easy to verify that the relation between the reflected jump $\bar{r}_{pj}$ and the outgoing jump r_{pj} is simply

$$\bar{r}_{pj} = -\mathcal{R}(r_{pj})$$

and the speed of the reflected wave is $\bar{\lambda}_{pj} = -\lambda_{pj}$. So with this approach, we need only solve Riemann problems on our original grid and then reflect any waves that hit the boundary. In the high resolution version, we must also reflect the outgoing slope in the same way,

$$\bar{\sigma}_{pj} = -\mathcal{R}(\sigma_{pj}).$$

In addition, we must solve a boundary Riemann problem with data $u_r = U_1^n$ given by the cell adjacent to the boundary and $u_l = \mathcal{R}(u_r)$. With this data, there is one incoming wave that affects the grid values (the contact discontinuity will have speed zero by symmetry and the outgoing wave is ignored).

This wave reflection procedure is quite easy to implement, and is applicable for any mesh size and any time step.

Finally, we will discuss how this reflection procedure can be applied to a moving boundary, *e.g.*, a moving piston. We approximate the piston motion by assuming that the velocity is constant within each time step. Assume that the piston is located at $x = z_n$ at time t_n and is moving with speed s_n for $t_n \le t \le t_{n+1}$. Then the physically correct boundary condition is

$$v(z_n + s_n(t - t_n), t) = s_n$$

for $t_n \le t \le t_{n+1}$. Using the Galilean transformation, we can derive that

$$\begin{aligned} \rho(z_n-) &= \rho(z_n+) \\ v(z_n-) &= 2s_n - v(z_n+) \\ p(z_n-) &= p(z_n+) \end{aligned} \tag{3.3}$$

is the correct data for the boundary Riemann problem. This defines a generalization of the reflection operator $\mathcal{R}$. Determining the corresponding reflection of the energy, we find that a jump which hits the boundary should now be reflected using the following relations:

$$\begin{aligned} \bar{r}_{pj}^{(1)} &= -r_{pj}^{(1)} \\ \bar{r}_{pj}^{(2)} &= r_{pj}^{(2)} - 2s_n r_{pj}^{(1)} \\ \bar{r}_{pj}^{(3)} &= -r_{pj}^{(3)} + 2s_n r_{pj}^{(2)} - 2(s_n)^2 r_{pj}^{(1)}. \end{aligned} \tag{3.4}$$

For shorthand, we write $\bar{r}_{pj} = -\mathcal{R}_n(r_{pj})$. The reflected slopes can be determined by the same reflection, $\bar{\sigma}_{pj} = -\mathcal{R}_n(\sigma_{pj})$. The reflected speed $\bar{\lambda}_{pj}$ is simply equal to $2s_n - \lambda_{pj}$.

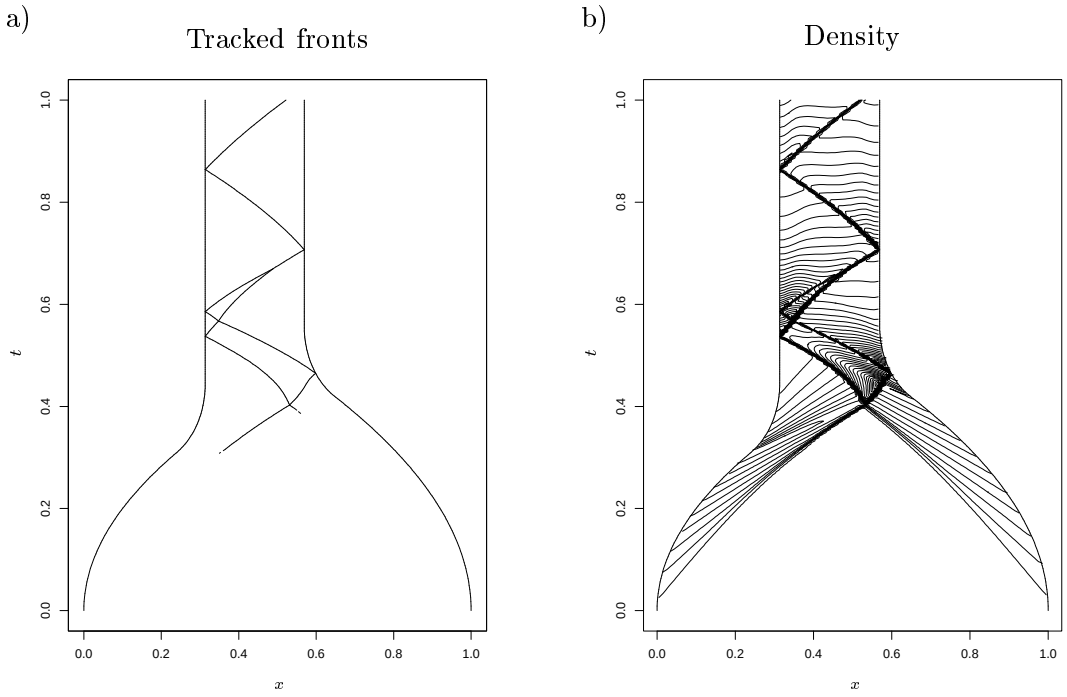


Figure 3.4: Results for the double piston problem. a) Tracked fronts. b) Density contour plot in the x-t plane up to time $t = 1$.

3.3 A Double Piston Problem

Here we present our first test problem consisting of a double piston. The problem is formulated as follows: take a shock tube with unit length and consider two pistons moving from the left and right boundary individually into the stationary gas ($\gamma = 1.4$) with $\rho = 1.4$ and $p = 1$. We choose piston velocities $s_p(t)$ of the form

$$s_p(t) = \begin{cases} \kappa_1 t & t \le t_1 \\ \frac{\kappa_2 (t_2 - t)}{\sqrt{r^2 - (t_2 - t)^2}} & t_1 < t \le t_2 \\ 0 & t > t_2. \end{cases}$$

The parameters for each piston are given by:

left piston: $\kappa_1 = 5$, $\kappa_2 = 1$, $t_1 = 0.31$, $t_2 = 0.444$

right piston: $\kappa_1 = -4$, $\kappa_2 = -1$, $t_1 = 0.42$, $t_2 = 0.557$

and $r = 0.16$ in each case.

Due to the piston motions, two compression waves arise from the left and right pistons, and eventually form shock waves. These new shock waves travel toward one another and subsequently interact. Two outgoing shocks result from the interaction, and begin to interact with the rarefaction waves and the pistons. The rarefaction waves are the consequence of stopping the pistons (see Figure 3.4b).

In the numerical method, we replace the piston path by a piecewise constant path using the constant velocity $s_p(t) = s_p(t_n + \frac{1}{2}k)$ over the time interval $t_n \le t \le t_{n+1}$, where k is the time step. Then the piston boundary conditions described in Section 3.2 are applied to

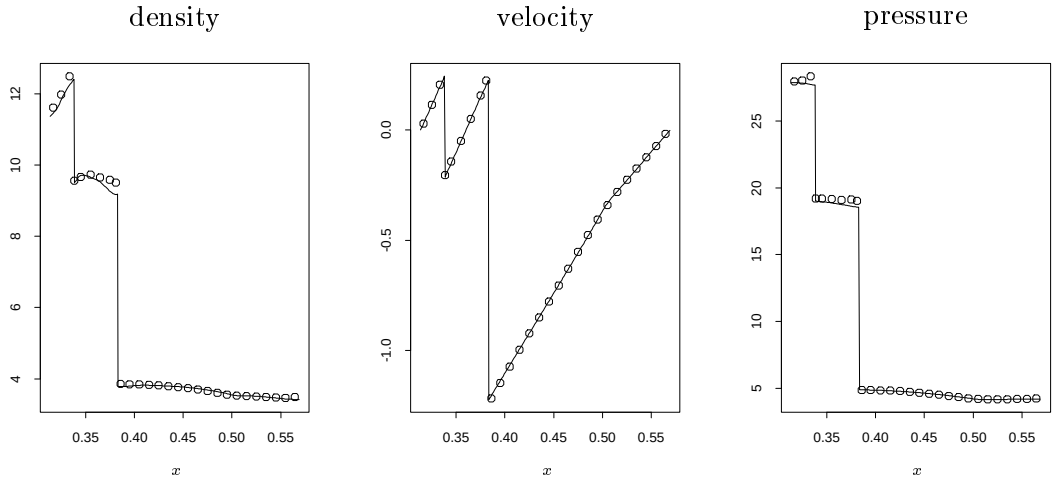


Figure 3.5: Comparison plots for the double piston problem at time $t = 0.6$. In each figure the solid line is the fine grid solution with $h = 1/800$ and the points show the solution with $h = 1/100$.

each piston. In Figure 3.4a, we show the tracked points which include the location of the tracked shocks and the pistons' path, by using the high resolution front tracking algorithm with mesh size $h = 1/100$ and Courant number $\nu = 0.9$. It is clearly seen that the shock formation and the tracked wave interactions have been handled quite well by using our front tracking algorithm.

For this problem, we track waves for which the density jump is greater than the tolerance $\varepsilon = 1$. From now on, unless otherwise stated, we use this tracking criterion to determine the tracked waves in the numerical examples given below.

The density contour plot in the x-t plane is shown in Figure 3.4b for the same run. From it, we can see that numerous wave interactions occur. The linear wave interaction is used for the interaction of tracked shocks with the background smooth flow and gives satisfactory results. In Figure 3.5, we compare our numerical result ($h = 1/100$) with the fine grid solution ($h = 1/800$) at $t = 0.6$, observing good agreement.

From this test problem, we see that our front tracking algorithm is capable of handling shock formation, moving boundaries, and wave interactions.

Chapter 4

ERROR ANALYSIS

In our front tracking algorithm, we use a high resolution method that is essentially second order accurate away from the tracked points. We use front tracking in order to resolve discontinuities properly, and so our method does not suffer the standard loss of accuracy due to smearing that a shock capturing method would suffer. Nevertheless, there can be some loss of accuracy near the discontinuity due to the nonuniformity of the grid. An isolated discontinuity separating two constant states is tracked perfectly, but in a more realistic situation there is some smooth background flow with which the discontinuity interacts. There are several factors that can then lead to loss of accuracy near the tracked front, such as the choice of slopes in neighboring cells, loss of accuracy due to the use of nonuniform and time-dependent grid, and the linearization of the interaction between the tracked discontinuity and weak waves from the neighboring cell interfaces. Here we will examine each of these problems in more detail. To begin, we report results on the order of accuracy for some sample problems where exact solutions are available.

4.1 Preliminaries

We first describe some notation and terms for later use. Let $u_j^n = u(x_j, t_n)$ be the pointwise value of the true solution at the discrete mesh point (x_j, t_n), and let $\bar{u}_j^n$ be the true cell average solution over the grid cell $[x_j, x_{j+1}]$ at time t_n,

$$\bar{u}_j^n \equiv \frac{1}{h_j} \int_{x_j}^{x_{j+1}} u(x, t_n)\, dx.$$

The *global error* of a numerical method is defined to be the difference between the true and computed solutions. Here we consider using either the pointwise error

$$E_j^n = U_j^n - u_j^n, \tag{4.1}$$

or the cell average error

$$\bar{E}_j^n = U_j^n - \bar{u}_j^n, \tag{4.2}$$

to define the global error. Although for conservation laws it is preferable to consider the latter error, pointwise error is more convenient to compute in practice. We will make it clear what approach we use in due course, and for now we simply write E_j^n to denote the error in both cases.

With these definitions, we define the *order of accuracy* of a method as the largest real number p in some particular norm $\| \cdot \|$ for which

$$\| E^n \| = O(h^p) \qquad \text{for all } t_n \geq 0 \qquad \text{as } h \to 0. \tag{4.3}$$

Clearly, if p is a positive number in (4.3), a method is *convergent* in the sense that

$$\| E^n \| \to 0 \qquad \text{as } h \to 0,$$

for any fixed $t_n \geq 0$ and Courant number. Note that in the present context of front tracking h is the underlying uniform mesh size.

Here the norm we consider is a discrete norm that can be applied to the discrete grid function E^n having errors in both the regular and irregular cells as elements. For example, in the 1-norm, we use

$$\| E^n \|_1 = \sum_j h_j |U_j^n - u_j^n|,$$

where h_j is the mesh size of the jth grid cell, and in the max-norm, we use

$$\| E^n \|_{\max} = \max_j |U_j^n - u_j^n|.$$

To compute the order of accuracy of a method, we employ a linear least-squares fit to a sequence of mesh refinement data $\{(\log h_l, \log \| E^n \|), l = 1, \cdots, m\}$, and take the slope as the order of accuracy of the method.

Now let us consider some sample problems and investigate the order of accuracy that is achieved by using our front tracking algorithm.

Example 4.1. We first consider a scalar linear problem consisting of the linear advection equation

$$u_t + u_x = 0 \qquad \text{for} \quad 0 \leq x \leq 1 \tag{4.4}$$

with initial data

$$u(x,0) = \begin{cases} 2 + 1.5e^{20(x-0.32)} & x < 0.32 \\ 1 + 0.5\tanh(6\pi(0.36 - x)) & \text{otherwise,} \end{cases} \tag{4.5}$$

and outflow boundary conditions on the left and right boundaries. The exact solution for this problem can be obtained by simply shifting this initial profile with speed 1 as illustrated in Figure 4.1a. Note that this initial data gives a single discontinuity with an extreme point just behind the discontinuity.

To examine the error behavior of the method as time evolves and as the mesh is refined, we perform error estimation at 10 different times (at every integer multiple of the time interval $k = 0.04$) with a mesh refinement sequence $\{h_l = 2^{1-l}/25, k_l = h_l/2, l = 1, 2, \cdots, 5\}$. The result is shown in Figure 4.2 where the errors and order of accuracy in the 1-norm and max-norm are presented for the Godunov method and the high resolution Godunov method with various slope limiters, namely, with the "minmod" (2.10), the "superbee" (2.11), and the "MUSCL" (2.12) limiters. From the figure, we observe that the accuracy of the methods we employed here is far less than satisfactory, particularly, for the high resolution methods in the max-norm; they are only slightly better than the Godunov method. This is expected, however, because the use of a slope limiter tends to clip the extreme point behind the discontinuity, and so the method is in fact first order accurate near the tracked point. Notice that no matter what method we use the 1-norm error grows as time evolves. There

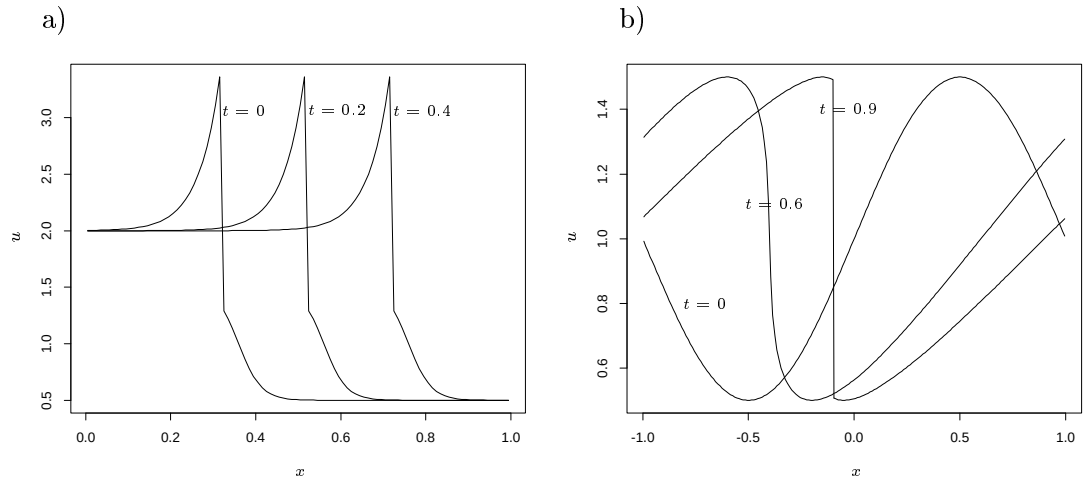


Figure 4.1: Snap shots of the exact solution at three different times. a) For the scalar linear problem, Example 4.1. b) For the scalar nonlinear problem, Example 4.2.

is little distinction between the results for different choices of the limiter. For convenience in reading, the errors shown in the figure were plotted in the logarithmic scale with base 10. (This is also the case for other figures shown below relating to errors of a method.)

For comparison, it is interesting to see how the standard shock capturing methods work on this problem. As seen from the result shown in Figure 4.3, our front tracking result is clearly superior to that obtained from shock capturing.

Note that in the above calculations the Courant number $\nu = 0.5$ (*i.e.*, $k = h/2$) is used, and the error is calculated based on the cell average error (4.2). From now on, we use this Courant number in all the test problems considered here. Although the computed order of accuracy is slightly different from other choices of Courant number, the convergence behavior of the method is quite similar.

Example 4.2. Next, we consider a scalar nonlinear problem consisting of the inviscid Burgers' equation

$$u_t + (u^2/2)_x = 0 \qquad \text{for} \quad -1 \le x \le 1 \tag{4.6}$$

with initial data

$$u(x, 0) = 1 + 0.5 \sin(\pi x), \tag{4.7}$$

and periodic boundary conditions. With these initial and boundary conditions, it is easy to show that the exact solution is smooth up to the shock formation time $t = 2/\pi \approx 0.64$, and is discontinuous afterward, see Whitham[108] for the detail on the construction of the exact solution. Figure 4.1b shows several snap shots of the exact solution.

In Figure 4.4, we show results for a similar accuracy study as the one performed in the previous example up to time $t = 1.2$. Now we observe that in the 1-norm the method is convergent with first order and second order accuracy, respectively, for the Godunov method and the high resolution methods we employed here. The result in the max-norm, however, falls short of the desirable value to some extent, particularly for the high resolution methods during the time when the solution is smooth. This loss of accuracy for the smooth

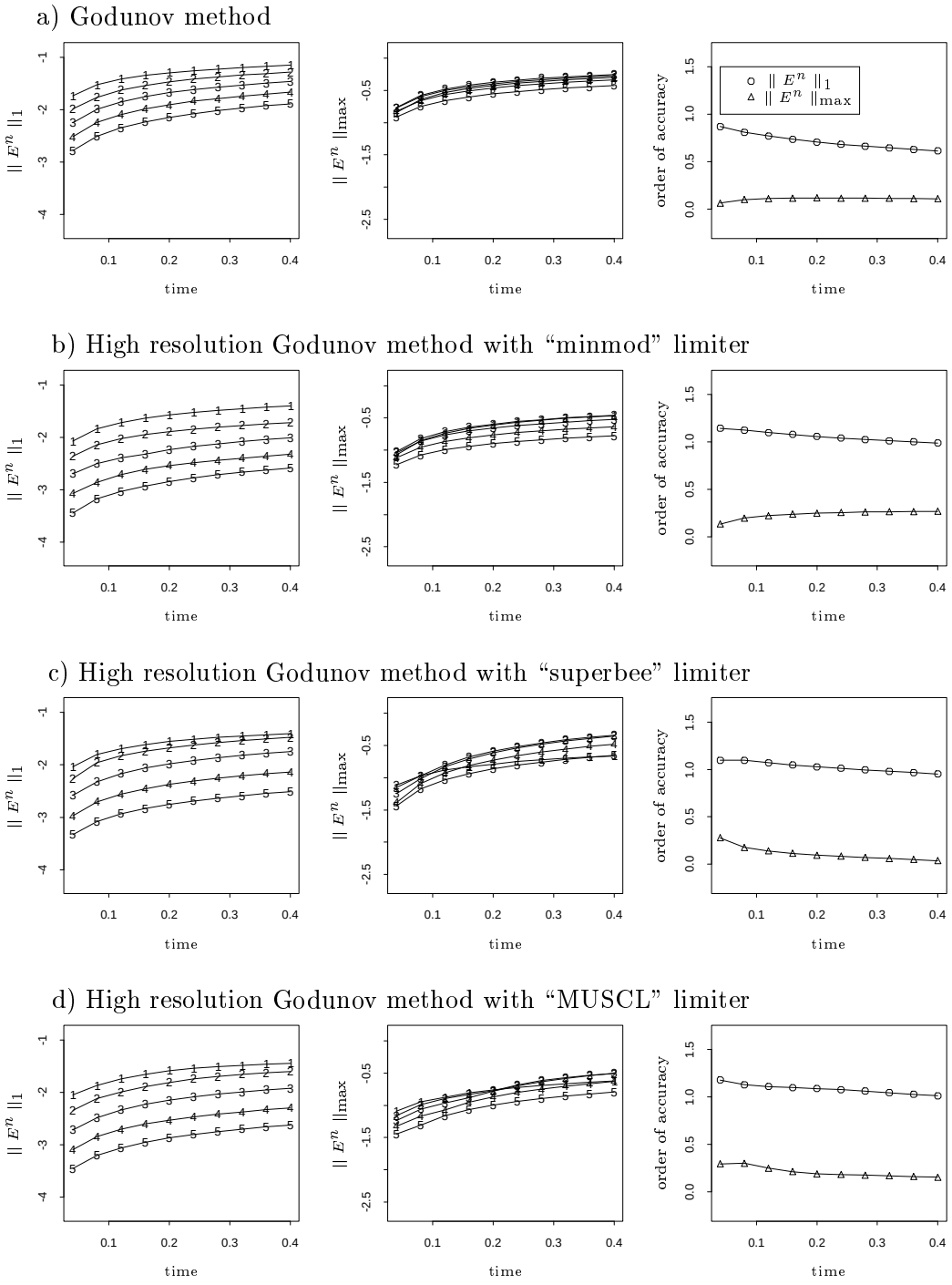


Figure 4.2: An accuracy study of the front tracking algorithm for the linear advection equation (4.4) with initial data (4.5) up to time $t = 0.4$. Note that all the errors shown in the figure are plotted in the logarithmic scale with base 10. Error estimation is performed at 10 different times with a mesh refinement sequence $\{h_l = 2^{1-l}/25, l = 1, 2, \cdots, 5\}$.

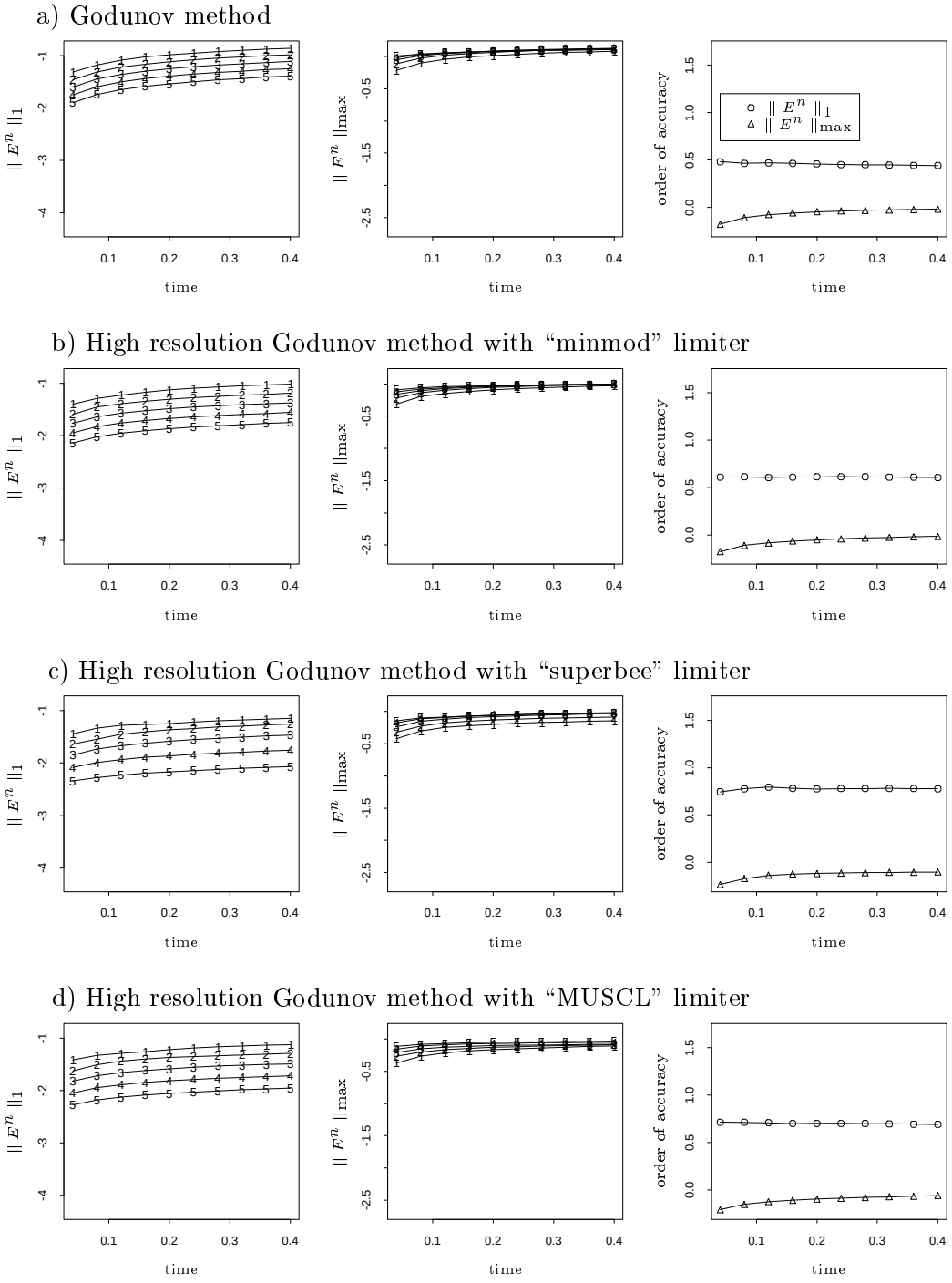


Figure 4.3: An accuracy study of the shock capturing method for Example 4.1. (See Figure 4.2 for comparison.)

solution results from using a slope limiter that gives a TVD method that clips solutions at the extreme points[114].

Notice that the method is somewhat less accurate near the transition period from the continuous solution to the discontinuous solution. This is reasonable, however, since correct shock formation time is not built into the algorithm. The algorithm determines that a shock has formed when the solution to some Riemann problem has a wave of sufficient strength. This will not be inserted at precisely the correct time or location. The third column of Figure 4.4 shows that the shock location tends to improve as time goes on. (Here the tolerance we use is $\varepsilon = 0.35$ for the wave strength at which we start out tracking.) Once the shock has been detected, the error decreases later on. Note that the error in the tracked point location, defined by

$$E^n_{\text{front}} = x^n_{\text{true}} - x^n_{\text{computed}},$$

converges with high precision, $O(10^{-6}) - O(10^{-7})$, for the high resolution methods.

To rule out the error due to dealing with the shock formation, we have also done experiments using the exact (discontinuous) solution at time $t_0 = 2/\pi + 0.2$ as initial data. The results are shown in Figure 4.5. From it, we again observe the nice error behavior in the 1-norm. The error in the max-norm tends to converge at a first order rate for the high resolution methods, and the error in the tracked point location is convergent at a rate of more than first order for the high resolution methods. Comparing our front tracking result with the shock capturing result shown in Figure 4.6, our tracking result is again better than the capturing result. Here the error is computed based on the pointwise error (4.1).

Example 4.3. We now consider a wave interaction problem arising from the nonlinear isothermal equations

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho v \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho v \\ \rho v^2 + c^2\rho \end{pmatrix} = 0 \tag{4.8}$$

where c is the speed of sound, a constant here for which we take $c = 1$. The initial condition we use consists of a leftward going simple wave with velocity profile

$$v(x,0) = \tanh(6\pi(x - 0.64)) \qquad \text{for} \quad 0.4 \le x \le 1, \tag{4.9}$$

traveling from the left to right, and a rightward going Mach 2.89 shock wave at $x = 0.4$ traveling from the right to left. The density of the simple wave is computed from the Riemann invariant $R_+ = v + c\log(\rho)$, which is constant on the entire $\lambda_1 = v - c$ wave family, with $\rho_0 = 0.5$ and $v_0 = 0$ as the reference state (this determines the Riemann invariant constant). Note another Riemann invariant for this system is $R_- = v - c\log(\rho)$, which is constant on the $\lambda_2 = v + c$ family. Since these waves are approaching each other, wave interactions occur subsequently, see Figure 4.7a.

For this nonlinear wave interaction problem, due to the fact that there is no new wave family appearing after the head-on collision, we can compute the "exact" solution by employing the Rankine-Hugoniot jump conditions at the shock together with the simple wave solutions on each side of the shock. Using this information would lead to a nonlinear ordinary differential equation for the shock location with respect to time. This is solved using a numerical ODE solver in the LSODE (Livermore Solver for Ordinary Differential Equations) Package. Once the shock location is known, the solution on both sides of the shock

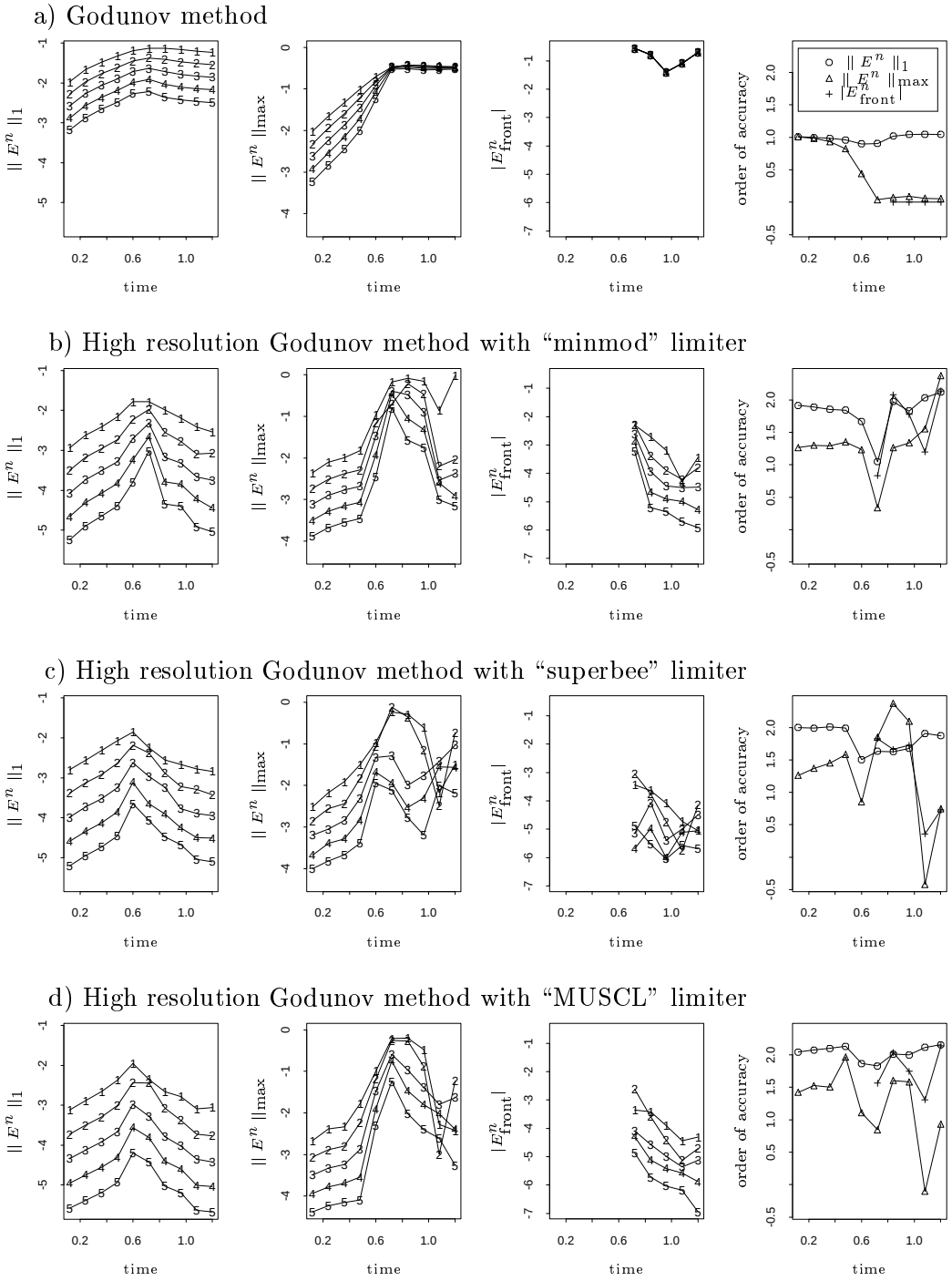


Figure 4.4: An accuracy study of the front tracking algorithm for the Burger's equation (4.6) with initial data (4.7) up to time $t = 1.2$.

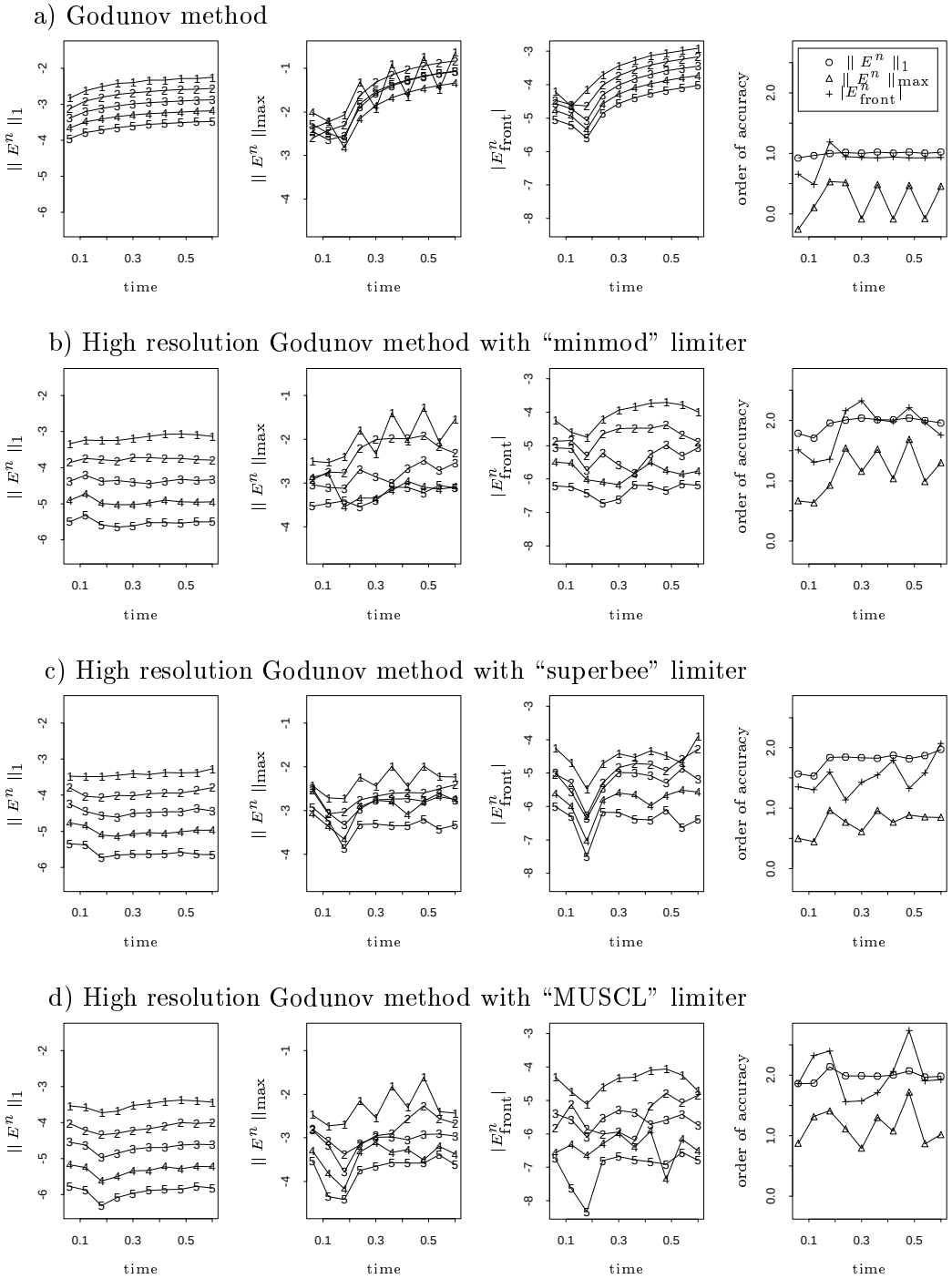


Figure 4.5: An accuracy study of the front tracking algorithm for Example 4.2 using the exact (discontinuous) solution at time $t_0 = 2/\pi + 0.2$ as initial data.

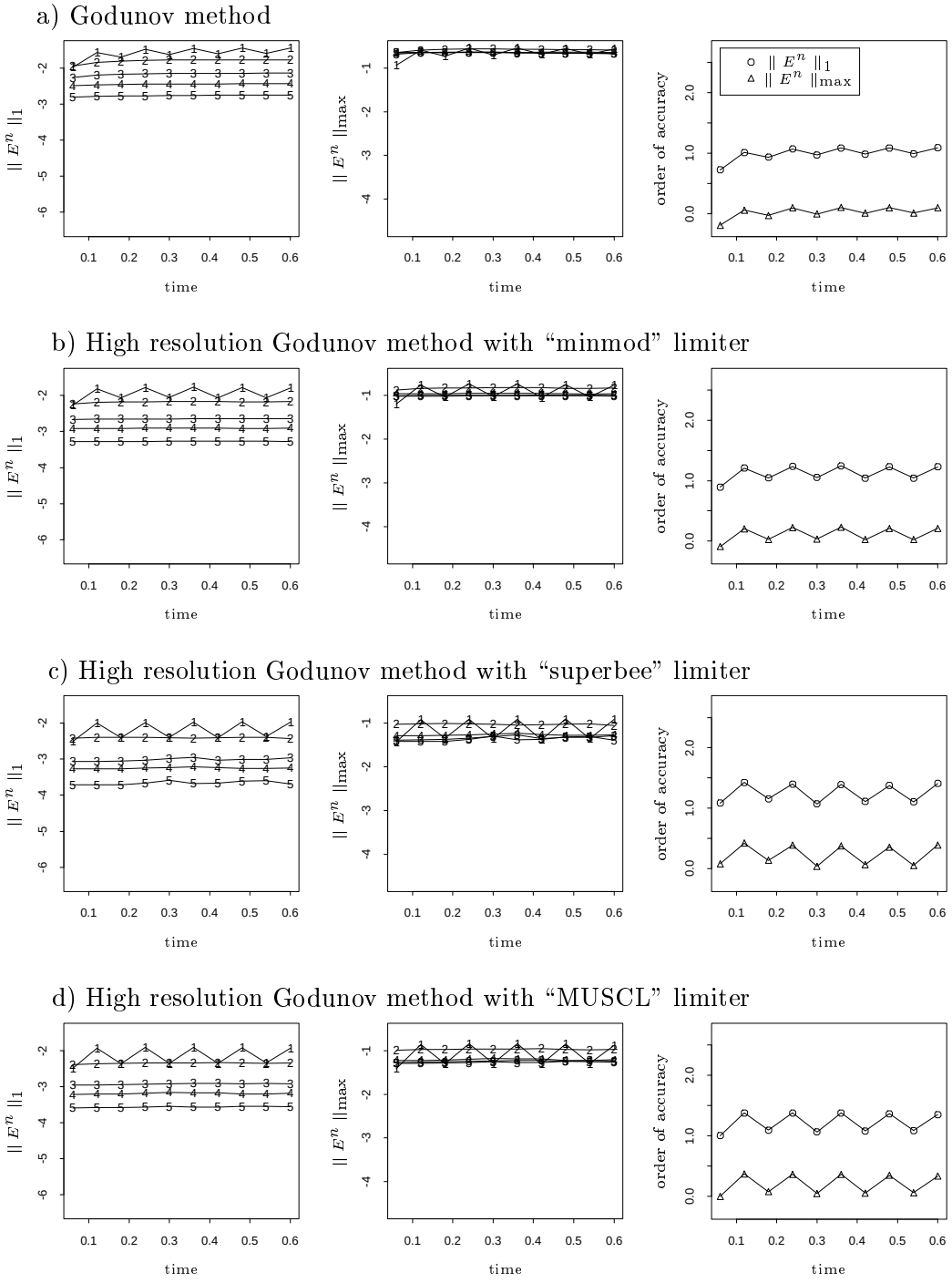


Figure 4.6: An accuracy study of the shock capturing method for Example 4.2 using the exact (discontinuous) solution at time $t_0 = 2/\pi + 0.2$ as initial data. (See Figure 4.5 for comparison.)

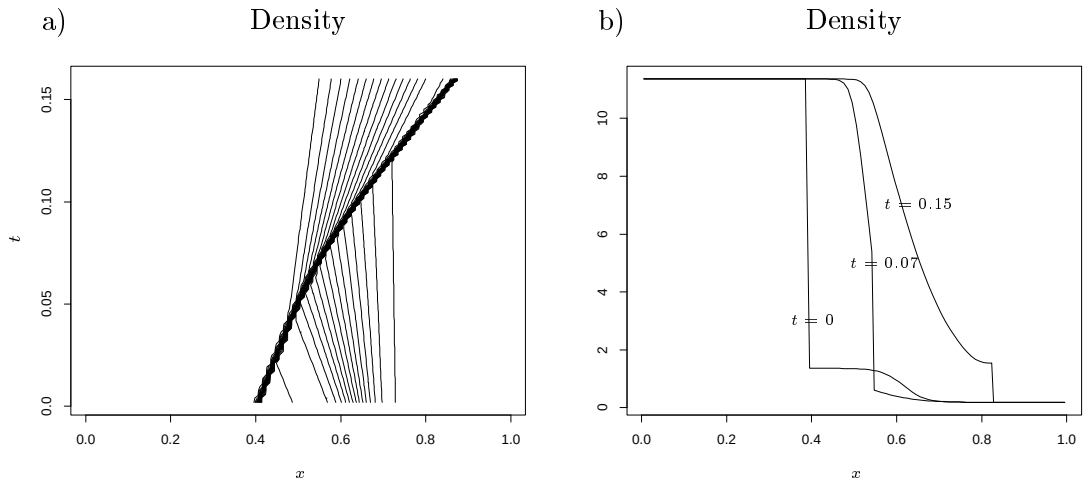


Figure 4.7: The exact solution for a wave interaction problem arising from the isothermal equations, Example 4.3. a) Density contour plot in the x-t plane, plotted in the logarithmic scale. b) Snap shots of density at three different times.

can be found using the method of characteristics. Figure 4.7b shows several snap shots of the exact solution.

Results for this problem using the front tracking algorithm are shown in Figure 4.8 where the front error, the 1-norm error, and the max-norm error of the Riemann invariant R_- are presented. It is very encouraging that our method produces results that converge at a fairly good rate in the 1-norm, despite the fact that the wave interaction of the strong and weak waves is handled linearly by allowing them to pass through each other without changing speed or magnitude. Nevertheless, it can be seen quite easily, particularly in the max-norm, that there is some loss of accuracy due to the use of the linear wave interaction. Looking at the max-norm error more closely, this loss of accuracy apparently depends on the wave structure in the smooth flow that the shock interacts. That is, when the shock interacts with the smooth flow of a steep gradient it results in a bigger error than the one appearing in the interaction between the shock and a flatter profile. Note that the method gives a very accurate result in the front location. Results for this problem obtained using the shock capturing methods are shown in Figure 4.9. Here we again use the pointwise error (4.1) to compute the global error.

4.2 Improved Slopes

In the above accuracy study, the high resolution method described in Section 2.3 was used directly on the nonuniform grid generated by the uniform grid together with the appropriate tracked points. It turns out that we can improve upon the method by taking advantage of the fact that we know that large jumps in the solution should appear at the tracked points whereas the nearby flow should be smooth. High resolution methods using limiters were originally developed for shock capturing methods where a shock will typically be smeared over several grid cells. Since reasonable slope information may be unavailable in this region, limiting the slope to a value near zero may be appropriate. In the present context, however,

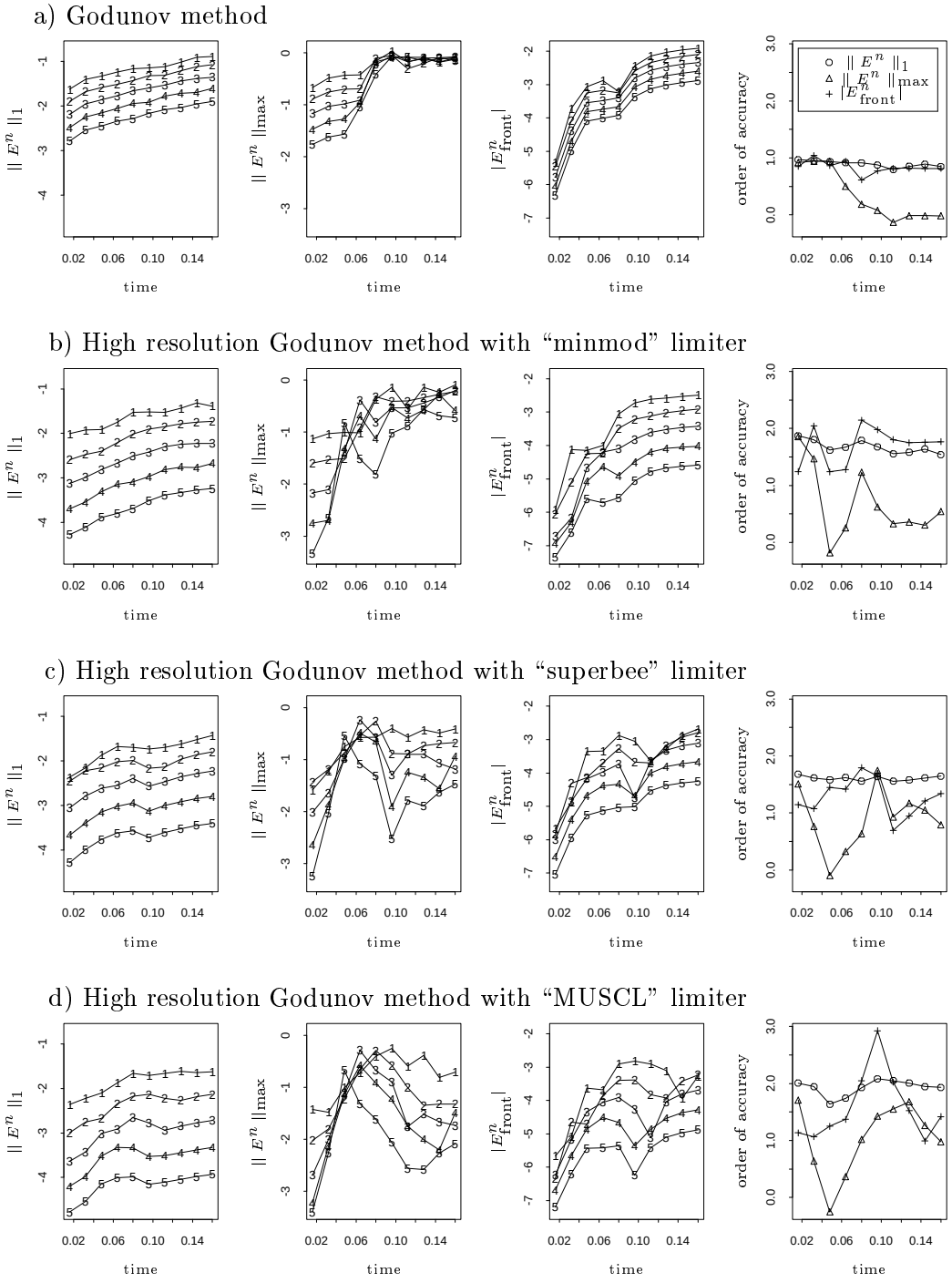


Figure 4.8: An accuracy study of the front tracking algorithm for the isothermal equation (4.8) with initial data (4.9) up to time $t = 0.16$. (Riemann invariant R_- is shown.)

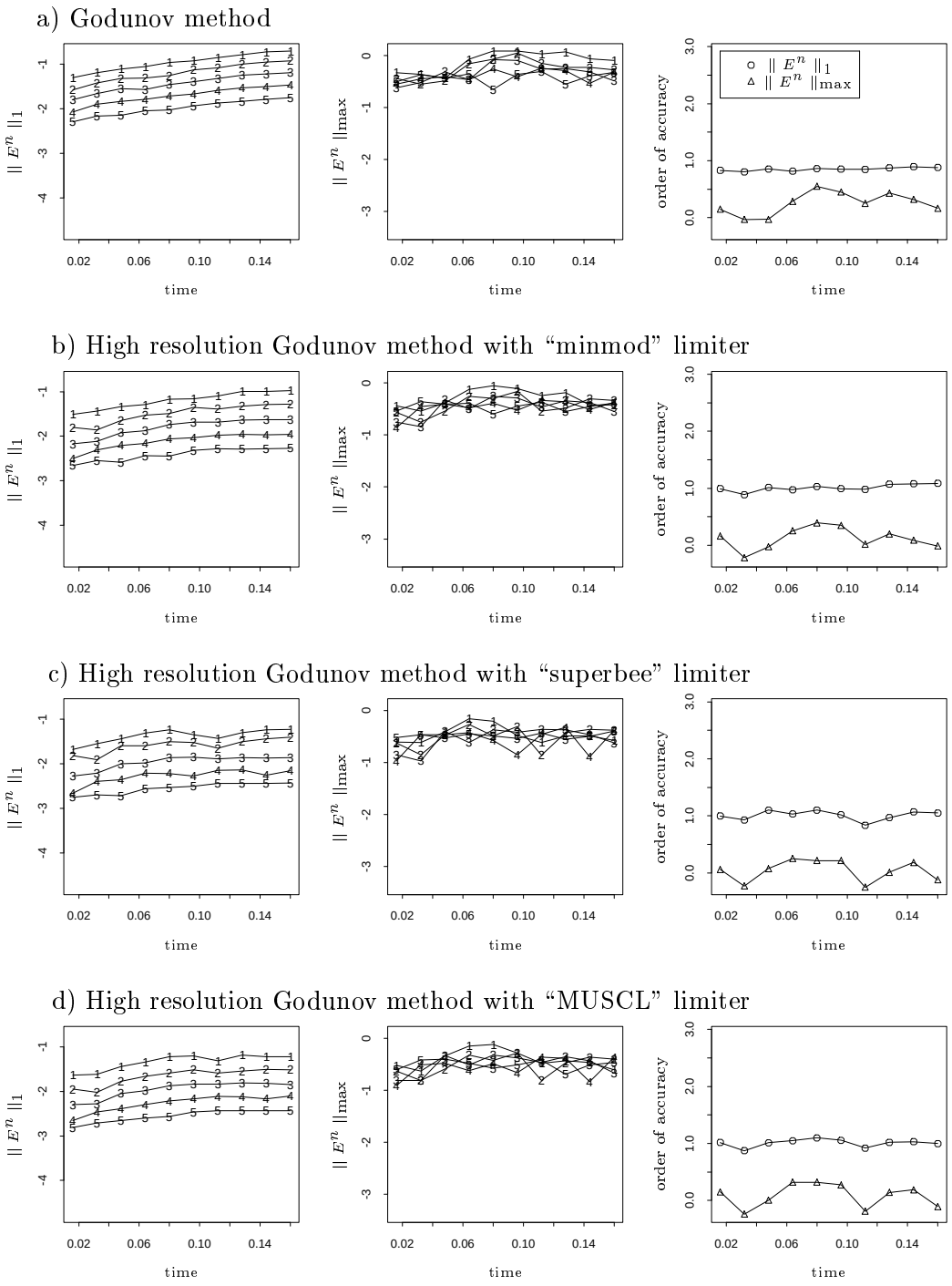


Figure 4.9: An accuracy study in Riemann invariant R_- of the shock capturing method for Example 4.3. (See Figure 4.8 for comparison.)

we might expect to have meaningful slope information in the cells near the discontinuity.

Consider a cell j, for example, where the interface to the right is a tracked point and interfaces to the left are regular grid interfaces. The solution to the Riemann problem on the right, at the tracked point, should clearly not be used to estimate a slope over this grid cell for the family of the tracked wave. But the wave arising from the Riemann problem to the left may give a very useful slope estimate. Since it is still valuable to compare adjacent slopes *via* a limiter in case other discontinuities are present that are not being tracked, we choose the slope σ_j based on a one-sided formula similar to (2.8) but using the wave r_{pj} at the boundary to the left and the wave $r_{p,j-1}$ at the left boundary of the adjacent cell.

This choice of slopes is particularly important if we wish to solve problems where the solution has an extremum at the discontinuity. This occurs in many applications, *e.g.*, in Figures 5.6 and 5.7. If we are not careful about the choice of slopes near the discontinuity, these extreme points will be severely clipped.

As an example, we consider the linear problem in Example 4.1, and we perform the same accuracy study as before, but using the one-sided slopes for the high resolution method. The result is shown in Figure 4.10. Notice that now errors in the 1-norm and the max-norm have been reduced. More importantly the order of accuracy have been improved also, particularly in the max-norm, see Figure 4.2 for comparison. It is interesting to note that no matter what slopes we used, see Figures 4.2 and 4.10, results obtained using the high resolution methods we employed here tend to converge only at the first order rate in the 1-norm. We have not yet achieved the desired second order accuracy for this problem with this modification of the method.

For a nonlinear problem, such as Example 4.2 with discontinuous initial data, we also examine the effect of accuracy by using this one-sided limited slopes in the high resolution methods. The result is shown in Figure 4.11, see Figure 4.5 for comparison. From it, we observe some improvement of the results, but the improvement is not significant. Note that unlike the previous linear problem where the grid is exact, here the error on the grid due to the discrepancy between the tracked point and the exact shock location contributes a source of error for the accuracy. Also, for the nonlinear problem errors are swept into the shock; so the slope improvement is not as important.

4.3 Nonuniform Grids and Accuracy

Although the underlying grid is uniform in our front tracking algorithm, the tracked points subdivide some cells into irregular cells. The analysis of the accuracy of finite difference method on nonuniform grids is more subtle than for uniform grids. A straightforward truncation error analysis may be misleading. For example, a natural generalization of Godunov's method for the linear advection equation (4.4) to a fixed nonuniform grid takes the form

$$U_j^{n+1} = U_j^n - \frac{k}{h_j}(U_j^n - U_{j-1}^n)$$

where h_j is the mesh size of the jth cell. A standard truncation analysis would show this method to be inconsistent unless the grid is very smoothly varying ($h_j/h_{j-1} = 1 + O(h)$), and yet it can be shown that the global error remains first order accurate on arbitrary grids. Similarly, generalizations of the Lax-Wendroff method to nonuniform grids has been

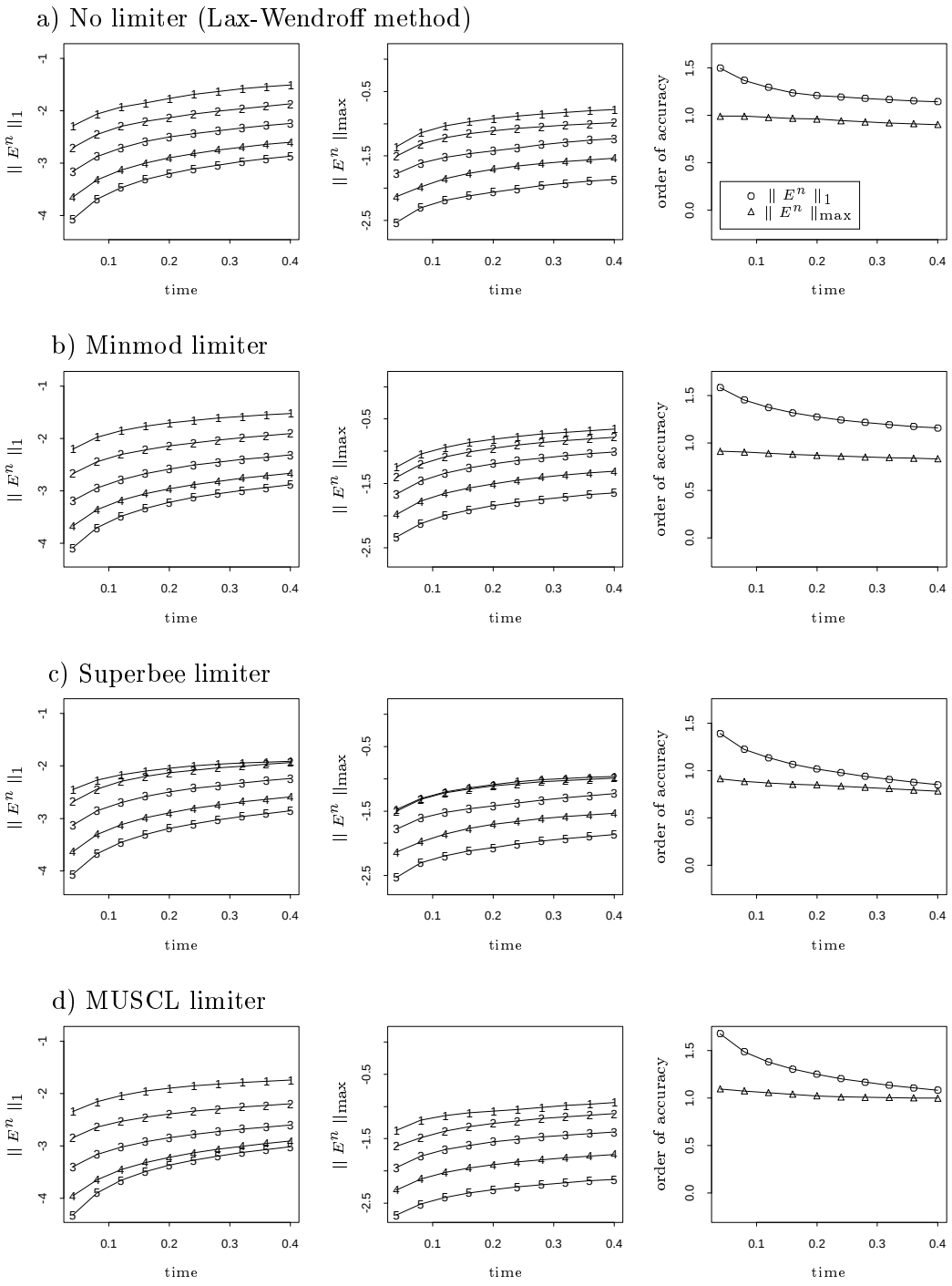


Figure 4.10: Revisit Example 4.1 using the high resolution methods with one-sided slopes. (See Figure 4.2 for comparison.)

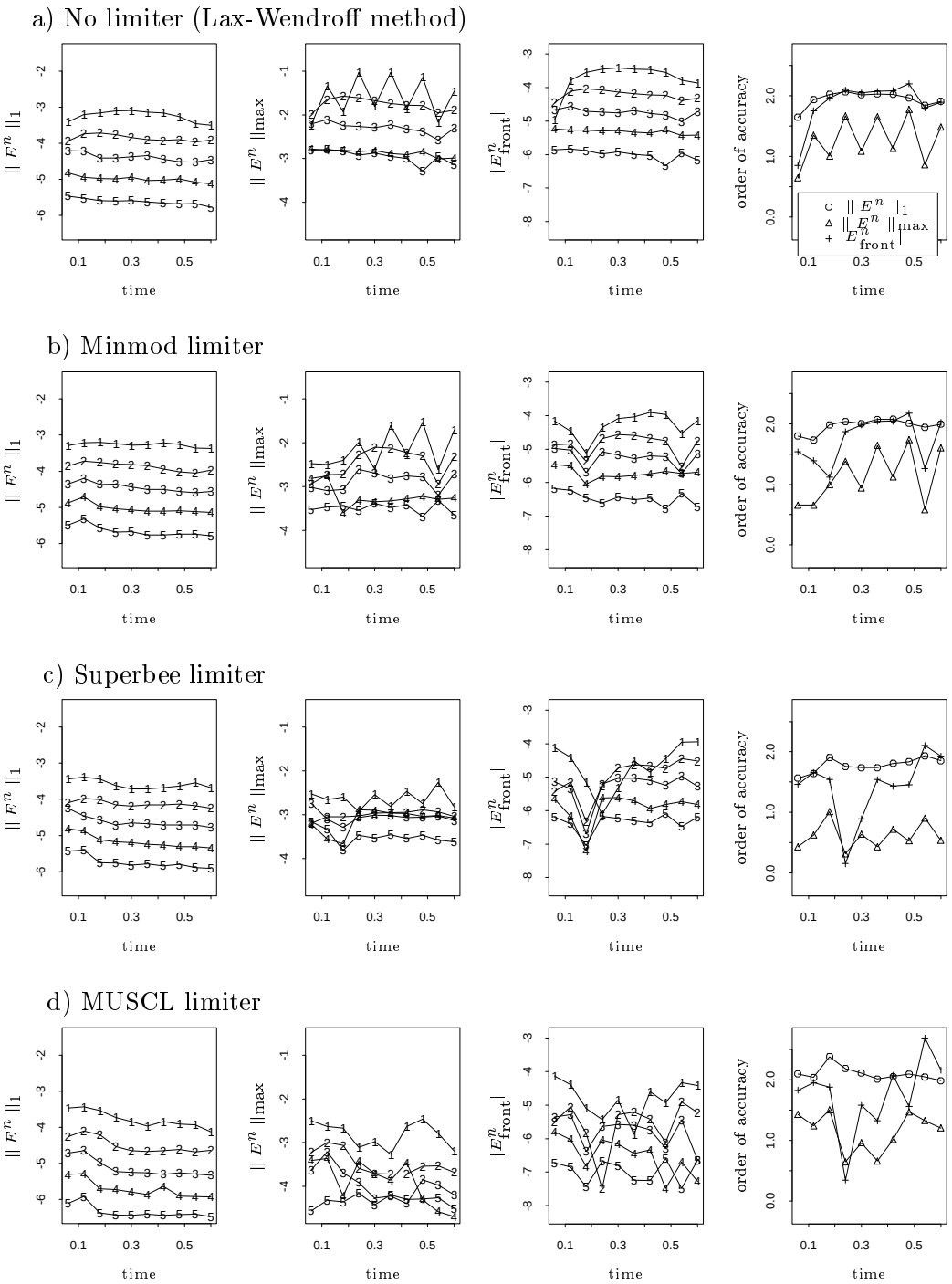


Figure 4.11: Revisit Example 4.2 using the high resolution methods with one-sided slopes. (See Figure 4.5 for comparison.)

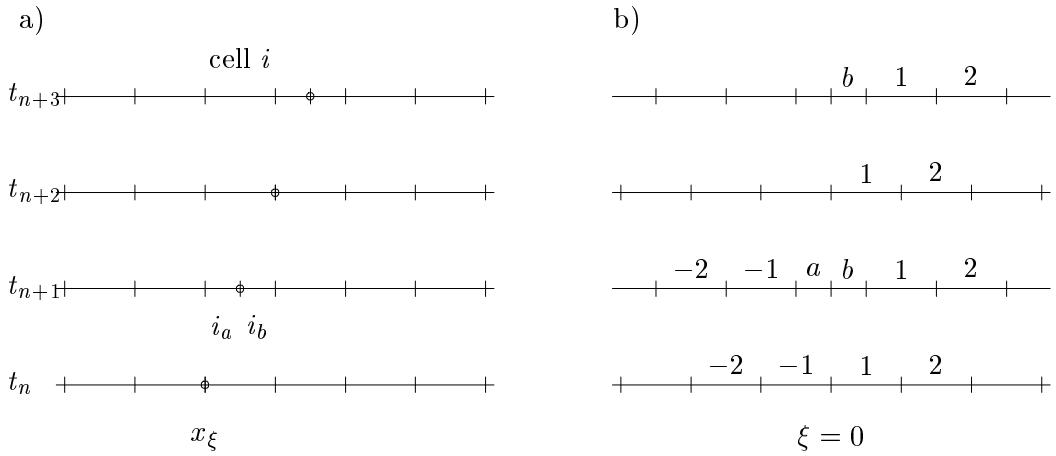


Figure 4.12: a) Grid system in the x-t plane for the linear advection equation (4.4) using the front tracking algorithm with $k = h/2$. b) Grid system for the transformed equation $u_t = 0$ in the ξ-t plane.

shown to maintain global second order accuracy[64],[106]. We might therefore hope that our method maintains second order accuracy on the smooth flow even in the irregular cells containing tracked points. Here, however, we have an additional complication in that our nonuniform grid varies with time. We will demonstrate with a simple example that loss of accuracy can occur under certain conditions.

Consider the linear advection equation (4.4) and suppose we solve this with Godunov's method using $k = h/2$ on the grid shown in Figure 4.12a. We introduce a single tracked point moving with speed 1 between cell interfaces and cell centers in alternating time steps. Away from the tracked point, the method reduces to

$$U_j^{n+1} = \frac{1}{2}(U_{j-1}^n + U_j^n).$$

The method differs from this only in cells that are split in two. Suppose cell i is subdivided at time t_{n+1} into two subcells i_a and i_b. According to the front tracking algorithm introduced in Section 3.1, we then introduce cells i_a and i_b at time t_n, initialized by

$$U_{i_a}^n = U_{i_b}^n = U_i^n. \tag{4.10}$$

Therefore we set

$$\begin{aligned} U_{i_a}^{n+1} &= U_{i_a}^n - \frac{k}{h/2}(U_{i_a}^n - U_{i-1}^n) \\ &= U_i^n - (U_i^n - U_{i-1}^n) \\ &= U_{i-1}^n \end{aligned}$$

and

$$\begin{aligned} U_{i_b}^{n+1} &= U_{i_b}^n - \frac{k}{h/2}(U_{i_b}^n - U_{i_a}^n) \\ &= U_i^n. \end{aligned}$$

In the next time step, we have

$$U_i^{n+1} = \frac{1}{2}(U_{i-1}^n + U_{i_a}^n)$$

and

$$U_{i+1}^{n+1} = \frac{1}{2}(U_{i_b}^n + U_{i+1}^n).$$

Notice that because the tracked point moves with the characteristic speed there is no transfer of information across this curve. The solution on each side is independent of the data on the other side.

Now consider a change of variable to $\xi = x - t$. In the ξ-t plane, the above method can be viewed as a staggered grid method to solve the equation

$$u_t = 0$$

on the grid shown in Figure 4.12b. Restricting our attention to the right of the discontinuity, the method is

$$U_j^{n+1} = \begin{cases} \frac{1}{2}(U_{j-1}^n + U_j^n) & j > 1 \\ \frac{1}{2}(U_b^n + U_1^n) & j = 1 \end{cases}$$

in time steps with $n+1$ even, and

$$\begin{aligned} U_j^{n+1} &= \frac{1}{2}(U_j^n + U_{j+1}^n) \quad j \geq 1, \\ U_b^{n+1} &= U_1^n \end{aligned} \tag{4.11}$$

in time steps with $n+1$ odd. It is easy to verify that this method is second order accurate on the "modified equation"

$$u_t = h u_{\xi\xi} \tag{4.12}$$

with the boundary condition

$$u_\xi(0,t) = 0. \tag{4.13}$$

This boundary condition results from the choice (4.11) for U_b^{n+1}.

Had we taken

$$U_a^{n+1} = U_b^{n+1} = \frac{1}{2}(U_{-1}^n + U_1^n), \tag{4.14}$$

we would simply have the heat equation (4.12) everywhere giving a first order accurate approximation to the equation $u_t = 0$. The choice (4.11) corresponds to setting $U_{-1}^n = U_1^n$ in (4.14) which models the boundary condition (4.13). If we now take initial data $u(\xi, 0)$ that does not satisfy (4.13), it will rapidly flatten out at $\xi = 0$ as time evolves, introducing an error at this boundary that is bigger than $O(h)$.

Returning to the original linear advection equation on the grid shown in Figure 4.12a, we see that the same effect occurs if $u_x \neq 0$ near the tracked point. As an example, we take

smooth initial data (4.7) with periodic boundary conditions on the left and right boundaries; $x \in [0, 2]$. We run this problem on a time-dependent grid as in Figure 4.12a with various choices of the initial tracked point location x_ξ at time $t = 0.5$. Doing so gives us some indications on how the accuracy is affected by the value of the u_x near the tracked point at this particular time.

Results of this accuracy study are shown in Figures 4.13 and 4.14 using not only the Godunov method, but also the Lax-Wendroff method as for comparison. Figure 4.13 shows the errors and the order of accuracy in the 1-norm and the max-norm. From it, for the Godunov method, the accuracy near the tracked point has been verified to be less than first order accurate in the max-norm, in the case where the boundary condition (4.13) is not satisfied at $x_\xi \neq 0.5$. Since the big error appears only for cells near the tracked point and has relatively small magnitude, the method remains first order accurate in the 1-norm. In Figure 4.14a, we plot the solution in two different cases, for $x_\xi = 0.5$ and $x_\xi = 1$ in that the boundary condition (4.13) is satisfied and not satisfied, respectively. Note that in the latter case the error near the tracked point is clearly seen.

It is interesting to note that for the Lax-Wendroff method on this model problem this loss of accuracy near the tracked point occurs in the case that the boundary condition $u_{xx} = 0$ is not satisfied there. Numerical results shown in Figures 4.13b and 4.13b confirm this.

One way we might try to improve the method is to choose a better initialization of the split cell values $U^n_{i_a}$ and $U^n_{i_b}$ in place of (4.10). We have tried introducing a linear function in cell i at time t_n with slope σ_i and averaging this linear function over the subcells to obtain initial values. For the case considered here, where the cell is split in half, this reduces to taking

$$\begin{aligned} U^n_{i_a} &= U^n_i - \frac{h}{4}\sigma_i, \\ U^n_{i_b} &= U^n_i + \frac{h}{4}\sigma_i. \end{aligned}$$

This procedure maintains conservation. Unfortunately, it seems to give little improvement in the results. For the particular problem considered here, other choices can be found that do restore full accuracy but are either rather artificial for this problem or else do not maintain conservation. Moreover for a nonlinear problem, any choice of $U^n_{i_a}$ and $U^n_{i_b}$ other than (4.10) has a major difficulty in conjunction with our algorithm – the speed of the tracked discontinuity is first computed using values U^n_{i-1} and U^n_i and this determines the location of the tracked point in the next time step. If we now choose a different value for $U^n_{i_a}$, the solution of the Riemann problem between U^n_{i-1} and $U^n_{i_a}$ will have a strong wave traveling at a different speed than the speed used to choose the new tracked point location. This means that the wave will no longer propagate exactly to the cell boundary, defeating one of our main design goals.

Because of this difficulty, we currently use the initialization (4.10) in spite of this possible loss of accuracy. On the other hand, the example shown here seems to be a worst case scenario. The particular grid shown in Figure 4.12 is especially bad due to the regularity of the nonuniformity. Moreover, in the linear advection equation all characteristics are parallel and the error continuously grows near the discontinuity. In a nonlinear problem, characteristics are swept into the shock, reducing the growth of errors. For the Euler

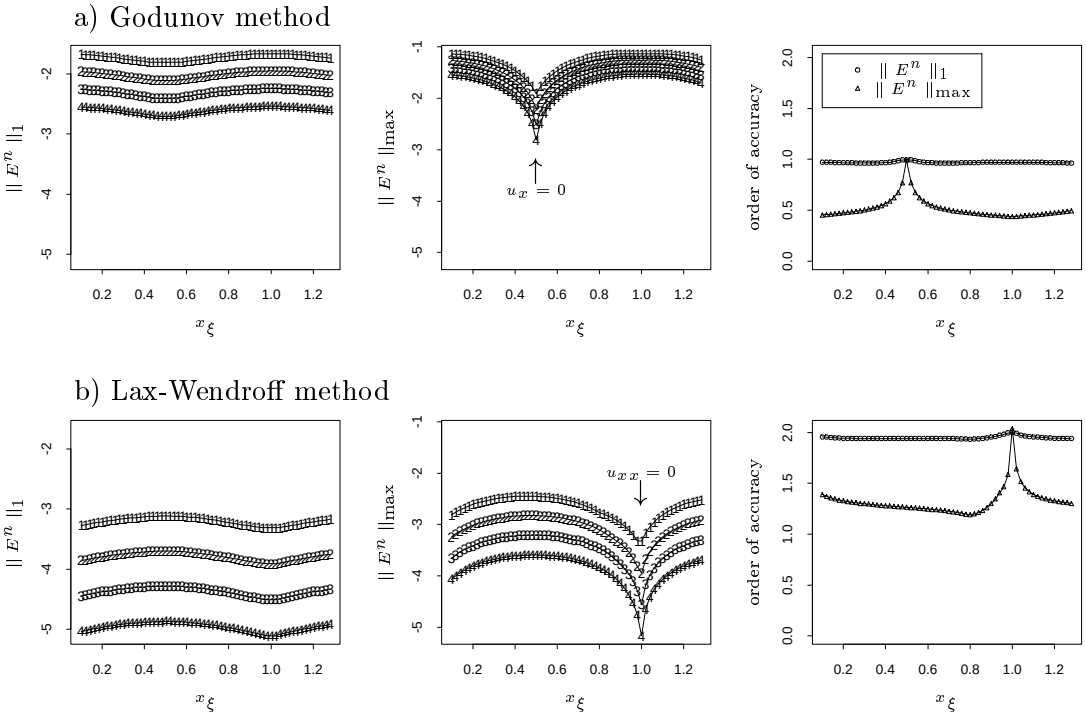


Figure 4.13: An accuracy study of the front tracking algorithm for the linear advection equation (4.4) with smooth initial data (4.7) on nonuniform grids. Results are shown with various choices of the initial tracked point location x_ξ at time $t = 0.5$.

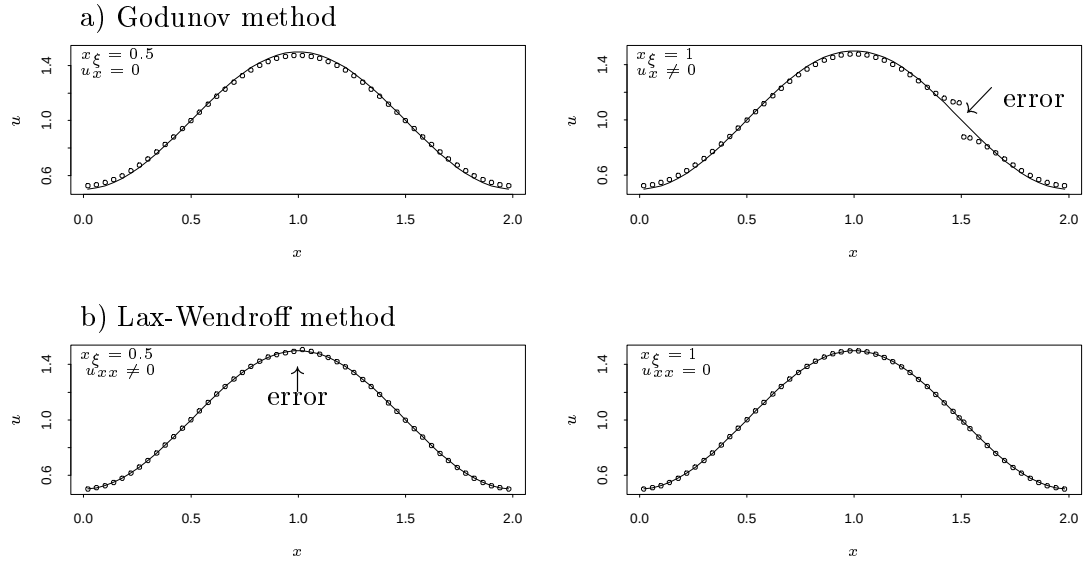


Figure 4.14: Plot of the solution for the accuracy study shown in Figure 4.13 at two different x_ξ. The solid line shown in the figure is the exact solution, while the dotted points are the numerical solution.

equations of gas dynamics, errors of the type seen here would most likely appear near contact discontinuities rather than shocks.

4.4 Nonlinear Wave Interactions and Accuracy

In our front tracking algorithm, we employ a large time step approach that avoids the stringent time step limitation in the presence of small cells created by the tracked points, while maintaining stability of the algorithm. An important element of this approach is to allow waves to pass through one another with no change in speed or magnitude, and continue to have exactly the same effect on cell averages as if the collision had not occurred. That is, the interaction is linearized during the wave propagation process. For a linear problem this is in fact true, but for a nonlinear problem this linearization in general is not valid. The aim of this section is therefore to investigate the error behavior introduced by this procedure for nonlinear problems. For more information on this problem, one may consult [60],[62].

For illustration, Figure 4.15 shows a typical wave interaction for the nonlinear isothermal equations (4.8). In Figure 4.15a, we see the solution of two Riemann problems in the x-t plane. When the 2-wave from the left hand Riemann problem meets the 1-wave from the right hand Riemann problem, we have a new Riemann problem to solve with left state u_l and right state u_r. The solution will again give two waves with some intermediate state $\bar{u}_m$, and will have different wave speeds and jumps across the wave. The locations of these states in the phase plane is shown in Figure 4.15b in relation to the Hugoniot loci of the states u_l and u_r.

In using the large time step method, we are linearizing the wave interaction. For the example considered above, we therefore obtain the wave structure as shown in Figure 4.16 with the intermediate state u_m^*. It can be demonstrated quite easily, using linear theory for the hyperbolic systems, see [62] for example, that the state u_m^* has the value

$$u_m^* = u_l + u_r - u_m.$$

In Figure 4.16b, we plot the location of u_m^* in the phase plane.

Notice that the error made in this approximation, which we might measure by $u_m^* - \bar{u}_m$, depends on the nonlinearity of the problem. In a linear problem, the Hugoniot loci are all parallel to one another, and there is no error. As the nonlinearity increases, these curves diverge more and more. The error also depends on the particular data u_l, u_m, and u_r. Even for a highly nonlinear problem the error will be small if these values are close to each other, since the Hugoniot loci in a small neighborhood of any given point have a nearly linear structure. If $u_l - u_m = O(\epsilon)$ and $u_r - u_m = O(\epsilon)$, then we may expect $u_m^* - \bar{u}_m = O(\epsilon^2)$ as $\epsilon \to 0$. In other words, since the interaction of weak waves is nearly linear anyway, our linearization introduces small error in this case. It is only in approximating the interaction of strong waves or the interaction of strong and weak waves that we might introduce large errors.

In the present context of front tracking, since the strong wave interaction has been dealt with carefully by choosing the time step so that the collision of two strong waves occurs exactly at the end of a time step, there is no error caused by the interaction of strong waves. There are errors, however, arising from the interaction of strong and weak waves. This is clearly seen from the earlier results shown in Figure 4.8 where the interaction of a shock

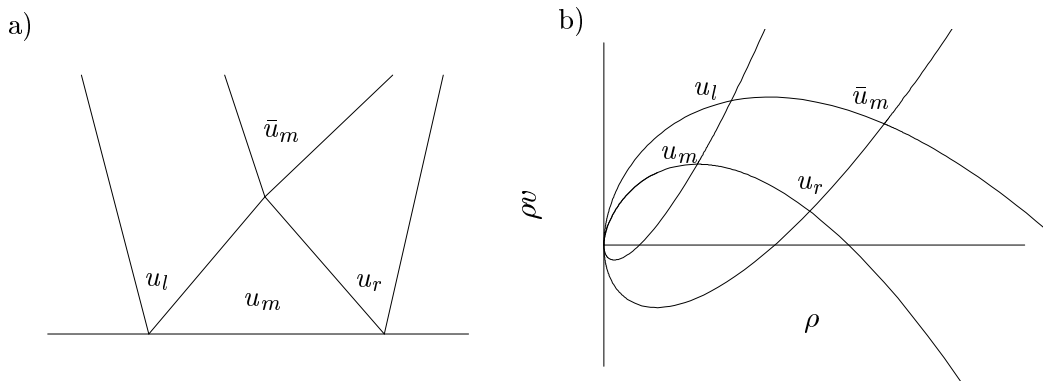


Figure 4.15: The interaction of two waves in a nonlinear problem. a) The x-t plane. b) Location of states in the phase plane.

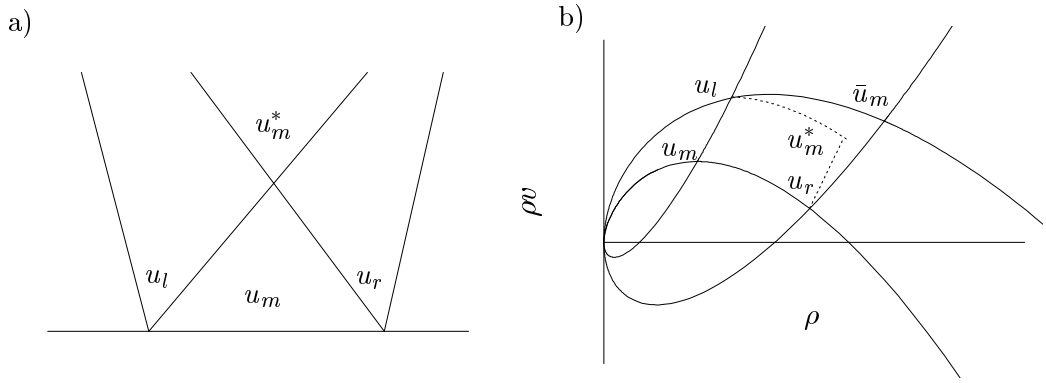


Figure 4.16: The linearized interaction of two waves in a nonlinear problem. a) The x-t plane. b) Location of states in the phase plane.

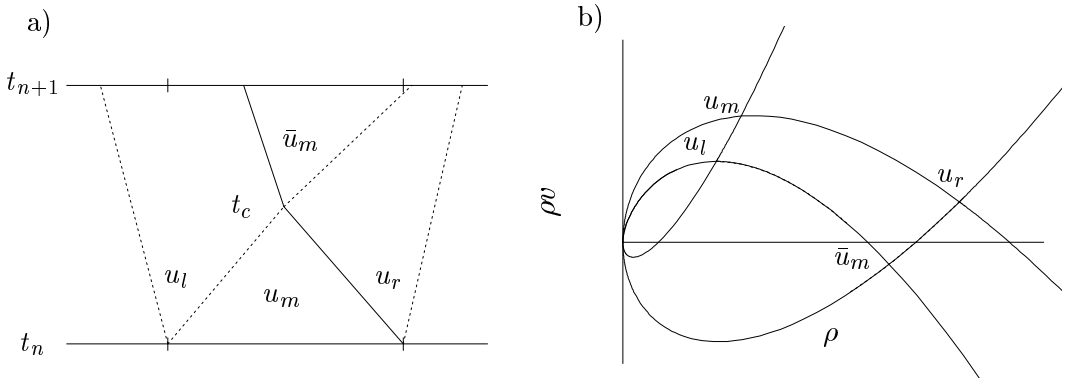


Figure 4.17: The interaction of strong and weak waves is handled "exactly" for a nonlinear problem. a) The x-t plane. The solid lines represent the strong waves, and the dashed lines represent the weak waves. b) Location of states in the phase plane.

and a simple wave for the isothermal equations is considered. Note that since this error occurs locally near the tracked point, the 1-norm error is only slightly affected by this loss of accuracy.

The worst case scenario of linearizing wave interaction appears in dealing with the interaction of the same wave family, *e.g.*, a rarefaction wave overtaking a shock. In this case, if the state variable (*e.g.*, density or pressure) is very small in front of the region where the wave interaction takes place, any small perturbation of the solution in that region caused by linearizing the wave interaction can result in a negative value of the state variable which is nonphysical. A situation like this can be seen in an example considered in Section 5.1.2 where a strong rarefaction wave overtakes a shock that has a very low pressure in front of it. The pressure becomes negative when the rarefaction wave passes through the shock and enters the low pressure region.

One possible approach to improve the method is to compute the interaction of the tracked discontinuity with the weak wave "exactly". This is not a new idea and was introduced by Swartz and Wendroff in their front tracking method[98]. We do this within the time step when the interaction occurs, modifying the strength and speed of these waves over the latter portion of the time step. This is illustrated in Figure 4.17a where the interaction of a shock and a rarefaction wave is handled by first propagating the pair of interacting waves up to the collision time t_c, solving a Riemann problem using the appropriate initial data (u_l and u_r in this case) at the collision point, and then propagating the resulting waves over the remaining portion of the time step $t_{n+1} - t_c$. Note that other waves are not affected by this procedure and are propagated over the time step k as usual. The locations of the state variable in the phase plane is shown in Figure 4.17b.

Using this modification of the method, we can overcome the stability problem mentioned above. It is also interesting to see how the accuracy is improved by using this modification of the wave interaction. This is still under study, however.

Chapter 5

APPLICATIONS

Having analyzed the front tracking algorithm, we now present more numerical results for some sample problems involving shocks and contact discontinuities arising in gas dynamics. Our aims here are to validate our results by comparing them to results (either exact or numerical) which can be found in the literature. We also hope to demonstrate the potential power of using our front tracking algorithm on more complex problems.

The problems we consider are the Woodard-Colella blast wave problem, the steady quasi one-dimensional nozzle flow, and unstable detonation waves. To efficiently perform computations on some of these problems, we first introduce an algorithm that combines front tracking with adaptive mesh refinement to enhance the resolution produced by the front tracking algorithm, particularly for the regions near tracked discontinuities. We then discuss a simple approach to include source terms in the algorithm.

5.1 Front Tracking with Adaptive Mesh Refinement

5.1.1 Algorithm

For simplicity, we describe grid refinement for the case of a single fine grid superimposed on a portion of the coarse grid. The refinement is performed in both space and time over rectangular regions of the space-time grid. Figure 5.1 shows the typical grid system for grid refinement with a mesh refinement ratio $m_r = h_c/h_f = 4$ where h_c and h_f are the coarse and fine grid mesh sizes respectively. If there are several fine grid regions, each fine grid can be handled in the same manner. Further nested levels of fine grids can also be handled.

In general, one would want to do error estimation on the approximate solution in order to determine where fine grids are needed. This can be done using the techniques developed by Berger[9], and should carry over to the front tracking method with little difficulty. Here we demonstrate the potential power of grid refinement coupled with front tracking in two test cases where we know *a priori* the region in which refinement should occur. We study a model combustion problem in which refinement is needed in the neighborhood of the single tracked discontinuity (see Section 5.2.2), and a blast wave interaction problem in which refinement is introduced in the neighborhood of two colliding strong shocks against a background smooth solution in order to better resolve the solution near the interaction (see Section 5.1.2).

For simplicity, we also assume that tracked fronts remain within the fine or coarse grid region and do not cross the interface between fine and coarse grids during a time step. The fine grid region is adaptively adjusted so that there is a sufficient buffer zone that fronts will not leave the refined region. This is accomplished by performing a regridding procedure at certain fixed time intervals. In cells where the new grid overlaps the old fine grid, the old fine grid value is carried over. In cells where a new fine grid is created where there was only coarse grid before, the fine grid cell values are initialized by performing piecewise linear

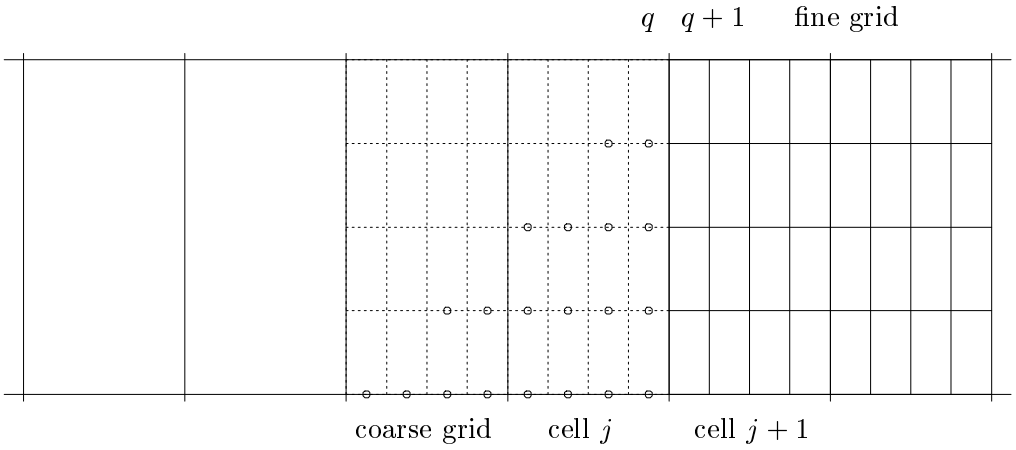


Figure 5.1: Interface between coarse and fine grids. Dashed lines represents the two additional coarse cells that are refined in order to generate fine grid fluxes at the interface. Values at the virtual fine grid points indicated by large dots are calculated using the high resolution method. We can compute only within a triangular region since the stencil of the method requires data from two adjacent cells. This is sufficient to compute fluxes all along the grid interface.

interpolation from the coarse cell values and evaluating this piecewise linear approximation at the cell center of each fine grid cell. Note that this gives a conservative transfer of values between grids. When a fine grid disappears in some region, the coarse grid values are set equal to the average of the fine grid values, again maintaining conservation.

The high resolution front tracking algorithm described in the previous chapters can be used on the fine grid and also on the coarse grid. The difficulty comes at the interface between the grids. Since we are assuming that tracked fronts do not cross this interface, we need only ensure that we maintain conservation and high accuracy with our underlying method at the interface. See Berger[5] for a general discussion of this problem for methods in conservation form with specified numerical flux functions. Although we normally use the "wave propagation" form of our high resolution algorithm because of the ease of dealing with irregular cells, near the grid interface we can reinterpret the method in terms of numerical fluxes as described in Section 2.3. Recall that we can define fluxes $F_{i+1/2}$ at each cell interface in such a way that the wave propagation algorithm is equivalent to the standard flux differencing formula

$$U_i^{n+1} = U_i^n - \frac{k}{h_i}(F_{i+1/2} - F_{i-1/2})$$

with fluxes given by (2.17) for the high resolution method.

To handle the grid interface, we first extend the fine grid to cover two additional coarse grid cells as indicated in Figure 5.1. Initial values at time t_n in these cells are computed using piecewise linear interpolation from the coarse grid values. The interface between cells j and $j+1$ on the coarse grid lies between cells q and $q+1$ on the fine grid, where $q = 2m_r$. Because our high resolution method involves a stencil of two grid cells on each side of an interface, we can compute over a triangular array of cells as indicated by dots in Figure 5.1

without noticing the lack of fine grid points to the left. In the process, we can compute numerical fluxes at the $q+1/2$ interface using the formula (2.17). On the fine grid, we take m_r time steps within the one time step on the coarse grid. We denote the flux for each time step $l = 1, 2, \cdots, m_r$ by $F^{fine}_{q+1/2,l}$. These are the fluxes that have essentially been used to update the fine grid cell $q+1$ to the right of the interface.

On the coarse grid, we can define fluxes $F^{coarse}_{j-1/2}$ and $F^{coarse}_{j+1/2}$ in the process of updating the coarse grid values *via* the high resolution method. We use these updated values as our new values in each of the coarse grid cells that are not overlapped by the fine grid, with the exception of cell j, just at the boundary of the fine grid. Here we replace the provisional value calculated with the coarse grid algorithm by the value

$$U^{n+1}_j = U^n_j - \frac{k}{h_c}(\hat{F}^{coarse}_{j+1/2} - F^{coarse}_{j-1/2})$$

with a modified flux $\hat{F}^{coarse}_{j+1/2}$ defined to be the average of the fine grid fluxes

$$\hat{F}^{coarse}_{j+1/2} = \frac{1}{m_r}\sum_{l=1}^{m_r} F^{fine}_{q+1/2,l}.$$

This ensures that the coarse grid flux at the right boundary of the coarse grid agrees with the total fine grid flux over time step k at the left boundary of the fine grid, giving global conservation.

This prescription for the interface has nothing to do with front tracking. With our front tracking method a new difficulty arises. If two fronts collide then we wish to adjust the time step so that collision occurs at the end of the time step. Since we assume that all tracked waves are within the fine grid and we integrate the fine grid first, we can simply truncate the fine grid time step during which collision occurs and then truncate the coarse grid time step at this same point. Suppose we have taken $m < m_r$ fine grid steps of length k/m_r at this point plus a shorter step of length $\tilde{k} \le k/m_r$. We have corresponding fine grid fluxes $F^{fine}_{q+1/2,l}$, $l = 1, 2, \cdots, m+1$. At this point we take a coarse grid time step of length $k_c = mk/m_r + \tilde{k} \le k$. The interface coarse grid cell j is updated by

$$U^{n+1}_j = U^n_j - \frac{k_c}{h_c}(\hat{F}^{coarse}_{j+1/2} - F^{coarse}_{j-1/2})$$

where $\hat{F}^{coarse}_{j+1/2}$ is now given by the appropriate weighted combination of each fine grid flux, taking into account that the last time step is shorter than the others,

$$\hat{F}^{coarse}_{j+1/2} = \frac{1}{k_c}\left[\frac{k}{m_r}\sum_{l=1}^{m} F^{fine}_{q+1/2,l} + \tilde{k}F^{fine}_{q+1/2,m+1}\right].$$

Since we never take more than m_r fine grid time steps in each coarse step, refinement of an extra two cells bordering the interface is sufficient to generate the fine grid fluxes needed at the interface.

If we allowed tracked fronts on the coarse grid as well, we would need slightly more complicated logic to truncate time steps appropriately if fronts collide on the coarse grid. This is clearly no problem, however.

Fronts moving between the coarse and fine grids could also be handled quite easily by simply truncating the time step when a tracked front hits the interface. Then within a given time step the front would be either on the fine grid or on the coarse grid and appropriate fluxes at the interface could be calculated.

5.1.2 The Woodward-Colella problem

As a first example to demonstrate the capability of front tracking with adaptive mesh refinement, we consider the blast wave interaction problem studied by Woodward and Colella[109],[110]. In this problem, the initial condition consists of three constant states with data

$$\begin{pmatrix} \rho \\ v \\ p \end{pmatrix}_l = \begin{pmatrix} 1 \\ 0 \\ 10^3 \end{pmatrix}, \quad \begin{pmatrix} \rho \\ v \\ p \end{pmatrix}_m = \begin{pmatrix} 1 \\ 0 \\ 10^{-2} \end{pmatrix}, \quad \begin{pmatrix} \rho \\ v \\ p \end{pmatrix}_r = \begin{pmatrix} 1 \\ 0 \\ 10^2 \end{pmatrix},$$

where l, m, and r are the states used for $x \in [0, 0.1)$, $x \in [0.1, 0.9)$, and $x \in [0.9, 1]$ respectively. There are two solid walls at $x = 0$ and $x = 1$.

With this initial condition a shock wave, contact discontinuity, and rarefaction wave develop at each discontinuity individually. The shock waves are moving toward each other and then collide. A new contact discontinuity arises from the collision. Further collisions then occur. A density contour plot in the x-t plane is shown in Figure 5.2 which indicates the complex wave pattern of this problem.

One of the main difficulties for this problem is the very low pressure in the middle state, and because of this any small perturbation caused by numerical error can lead to negative pressures which are nonphysical. Another difficulty involves the proper treatment of the strong wave interactions in a smooth background flow. Therefore this problem provides a severe test of our front tracking algorithm, and especially tests our ability to handle small cells and wave interactions. Furthermore, since complex wave interactions occur after the shock waves' collision, poor resolution will result near the interaction if the grid is not sufficiently fine. For this reason, we have used mesh refinement in addition to front tracking in order to better resolve the solution.

For this problem there is no mesh refinement initially. The mesh refinement is introduced after the shock waves' collision and used thereafter. For convenience, the refinement region is chosen to contain all the tracked fronts within one fine grid with a buffer zone to prevent them from moving onto the coarse grid. For the results shown below, we take coarse grid mesh size $h_c = 1/100$ as our underlying mesh size and use a mesh refinement ratio $m_r = 8$ for the fine grid, so that $h_f = 1/800$. The buffer zone has width $10h_c$, and a regridding step is done for every 16 time steps. Since the density jump is not prominent in this problem, we choose the max-norm of the jump in conservative quantities as our tracking criterion (with tolerance $\varepsilon = 50$). Throughout the test Courant number $\nu = 0.9$ is used, and only results obtained using the high resolution method are shown.

In Figure 5.2a, we show the density contour plot in the x-t plane over both the coarse and fine grids; contour lines were plotted on a logarithmic scale. A blowup of the fine grid solution is shown in Figure 5.2b. Notice the fine wave structure following the interaction between the rightward going shock wave and the leftward going contact discontinuity.

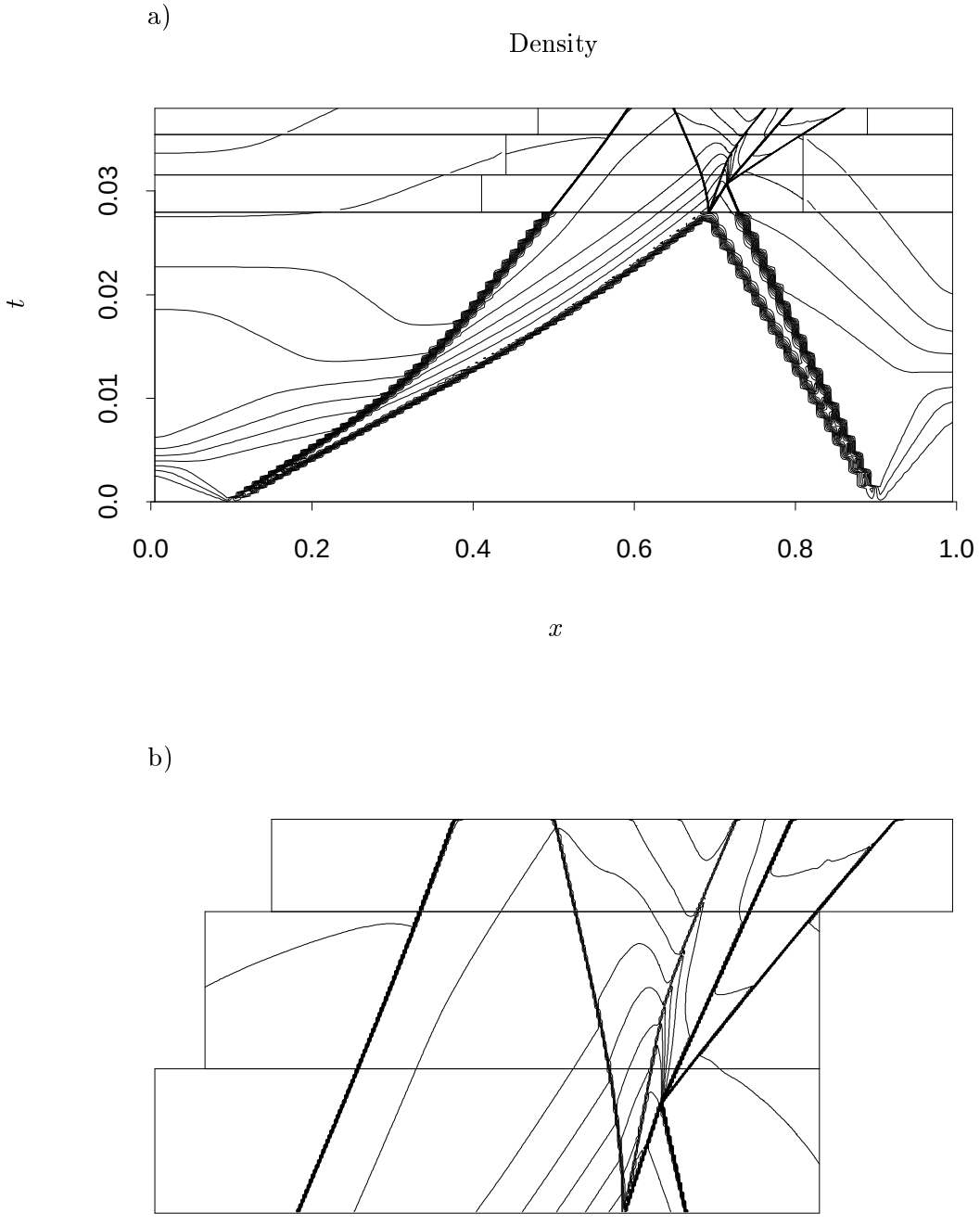


Figure 5.2: Density contour plot in the x-t plane (the contour lines are in the logarithmic scale) for the Woodward-Colella problem up to time $t = 0.038$ using the high resolution front tracking with adaptive mesh refinement algorithm with $h_c = 1/100$ and $m_r = 8$. a) Combined plot for both the coarse and fine grids. b) Blowup of the fine grid region.

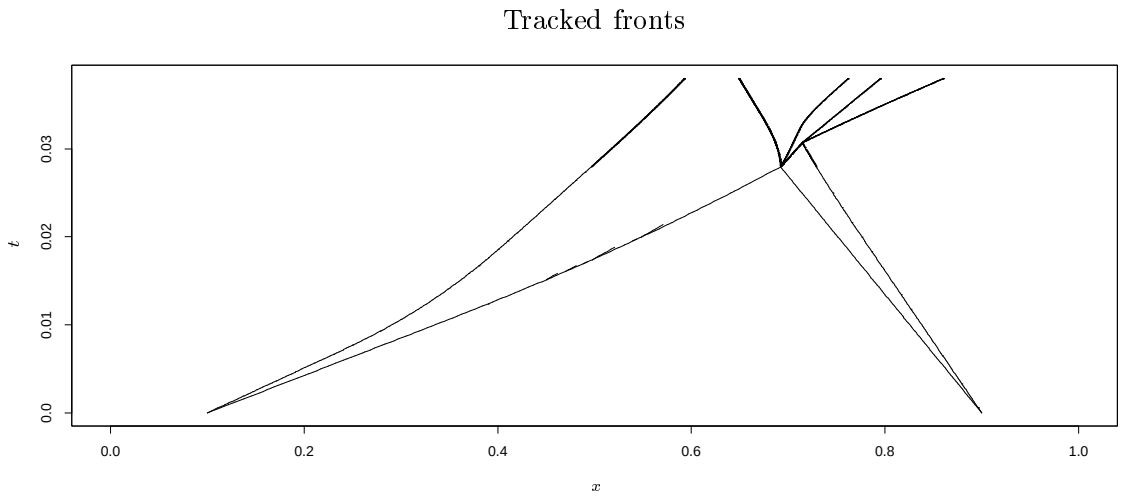


Figure 5.3: Tracked fronts for the Woodward-Colella problem.

Without mesh refinement, this wave pattern would not be clearly seen. Tracked points are shown in Figure 5.3.

To investigate the accuracy, we show plots of the state variables ρ and v versus a finer grid ("true") solution, computed using $h_c = 1/800$ in the time before refinement is introduced, and then $h_c = 1/200$, $m_r = 8$ for refinement. Results at three different times are shown in Figure 5.4. Note that they are plotted using the most accurate cell values at each point. That is, if a grid cell is in the fine grid region, we use the fine grid solution; otherwise we use the coarse grid solution. We see good agreement between the two solutions. Notice the smooth transition between the coarse and fine grids. This indicates that our treatment of the coarse-fine grid interfaces is working in a satisfactory way.

As mentioned above, this is a difficult problem due to the very low pressure in the middle state. A rarefaction arising from the smooth flow behind the shock may move faster than the tracked shock, carrying a negative jump in pressure into the low pressure region that is of sufficient magnitude to result in a negative pressure. This is due to the linearization of the interaction between waves.

We currently deal with this problem by computing the interaction of the rarefaction wave with the strong shock wave exactly rather than using the wave linearization that is used elsewhere. This has been discussed fully in Section 4.4. This leads to some complication of the algorithm, but avoids the need to further restrict the time step and eliminates the difficulties.

Naturally it would be preferable to find a more robust solution to this problem and work is continuing in this direction. We note, however, that this is a particularly difficult problem and that many production codes contain *ad hoc* procedures such as resetting negative pressures to positive values in order to deal with such problems. This is not a difficulty that arises solely from our front tracking methodology. On the contrary, our approach has the advantage that it allows one to recognize these difficulties and deal with the interaction correctly and conservatively. (See [29] for an interesting discussion of this problem.)

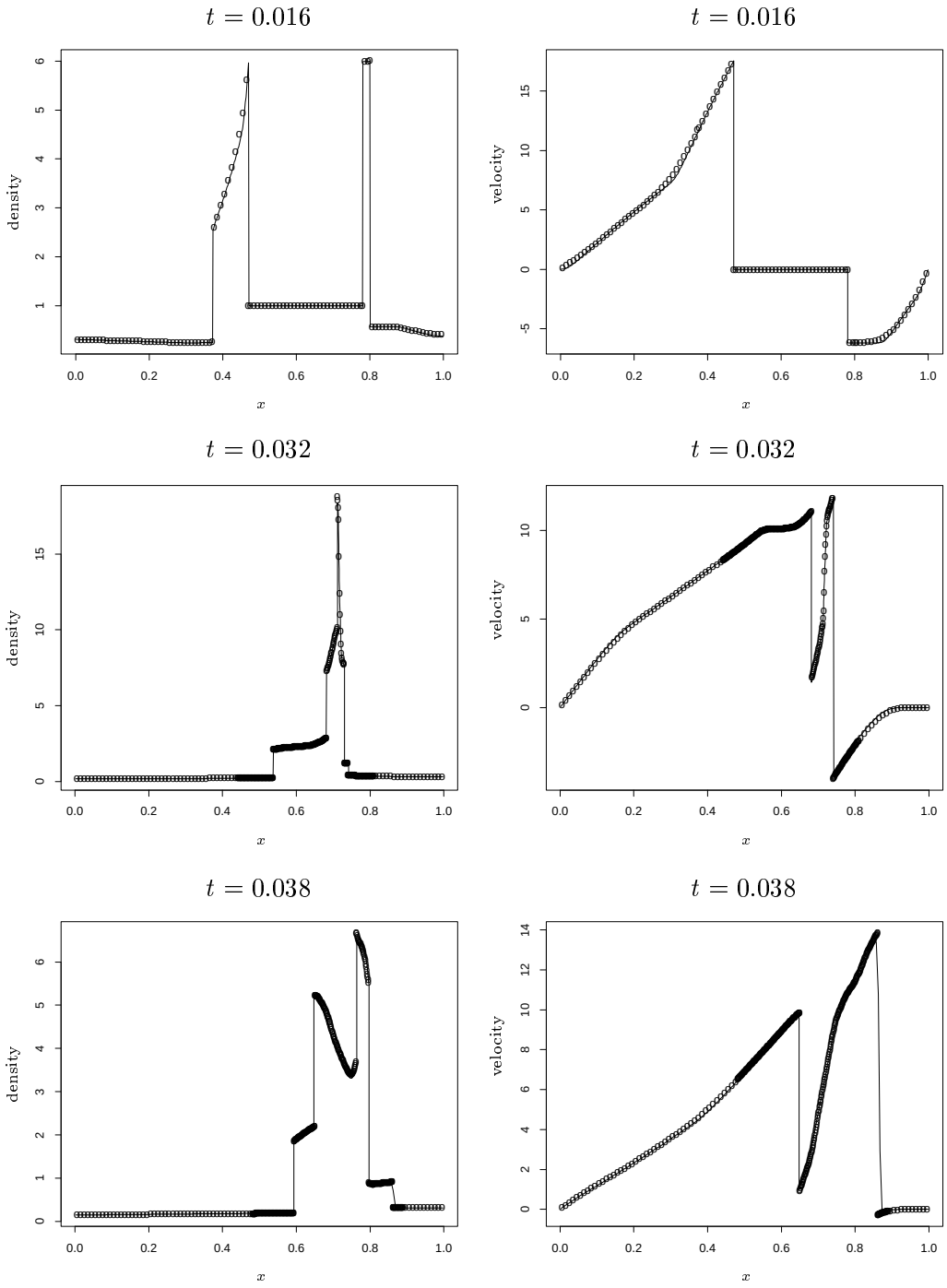


Figure 5.4: Comparison plots for the Woodward-Colella problem at three different times. In each figure, the solid line is the fine grid solution computed by $h_f = 1/800$ in the time when no refinement is used, and $h_c = 1/200$, $m_r = 8$ when the refinement is used. The points show the solution with $h_c = 1/100$ and $m_r = 8$. Density and velocity are shown.

5.2 Source Terms

A wide variety of numerical methods have been developed for conservation laws with source terms $u_t + f(u)_x = \psi(u)$, *e.g.*, [4],[37],[90],[96]. Here we will only consider one popular approach, a time-splitting method in which we alternate between solving the homogeneous conservation laws with no source terms

$$u_t + f(u)_x = 0, \tag{5.1}$$

and solving the ordinary differential equations

$$u_t = \psi(u) \tag{5.2}$$

within each cell.

We use a "Strang splitting" [97], which is second order accurate for smooth solutions. In the present context with front tracking, this consists of the following steps:

1. Take a half time step by solving the ODEs (5.2) in the old grid cells.

2. Take a full time step with the homogeneous equations (5.1) using the front tracking algorithm. This generates new grid cells.

3. Take a half time step again with the ODEs (5.2) in the new grid cells after removing the old tracked points.

Two examples will be given to demonstrate the ability to handle source terms by using this splitting procedures together with the front tracking algorithm.

5.2.1 Quasi one-dimensional nozzle flow

As a first example with source terms, we consider the quasi one-dimensional nozzle flow. The Euler equations now have the form

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho A \\ \rho v A \\ \rho E A \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho v A \\ (\rho v^2 + p)A \\ (\rho E + p)vA \end{pmatrix} = A' \begin{pmatrix} 0 \\ p \\ 0 \end{pmatrix} \tag{5.3}$$

where $A \equiv A(x)$ is the cross section of area, $A' \equiv dA(x)/dx$. The conservative variables u, flux functions $f(u)$, and source terms $\psi(u)$ are defined in the obvious way. In this example, $\psi(u)$ are called the "geometric" source terms since they result from the geometrical simplification to a one-dimensional problem, see, *e.g.*, [69] for more detail.

For incorporation into the front tracking algorithm, we rewrite these equations in another form by moving all the area terms in (5.3) to the right hand side,

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho v \\ \rho E \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho v \\ (\rho v^2 + p) \\ (\rho E + p)v \end{pmatrix} = -\frac{A'}{A}\begin{pmatrix} \rho v \\ \rho v^2 \\ (\rho E + p)v \end{pmatrix}. \tag{5.4}$$

Although these two sets of equations have different state variables and flux functions, they share the same Rankine-Hugoniot jump conditions. This is simply due to the fact that the variable area is continuous at the discontinuity, and so the area term drops out from each side of the jump condition. In practice, it is interesting to see how numerical results are affected by using these two different equations with the same numerical method. (See [107] for an interesting example of such a comparison.)

In the following, we perform a standard test problem for the quasi one-dimensional nozzle flow in the form (5.4), a steady state calculation[20]. Take a divergent nozzle with area

$$A(x) = 1.398 + 0.347 \tanh(8x - 4), \qquad 0 \le x \le 1.$$

Consider a supersonic inflow boundary condition at $x = 0$ with $\rho_{in} = 0.502$, $v_{in} = 1.299$, and $p_{in} = 0.3809$ (Mach number = 1.26), and a subsonic outflow boundary condition at $x = 1$ with $\rho_{out} = 0.776$. The steady state solution under these boundary conditions consists of a stationary shock at $x = 0.481991$ with steady smooth flow in front and in back of this stationary shock, see [95] for the detail on the construction of the exact solution.

To start the computation, we must also specify initial data in the interior region. For the test we present here, the velocity and pressure in the interior region were initialized to the inflow velocity and pressure respectively, and the density was initialized to be linearly varying from the inflow boundary to the outflow boundary. At the outflow boundary the unknown v_{out} and p_{out} are calculated using the numerical characteristic boundary conditions.

The numerical characteristic boundary conditions can be described briefly as follows. Suppose the outflow is subsonic, given ρ_{out}. Let cell j be the closest cell to the outflow boundary. The density, velocity, and pressure for the jth cell are denoted by ρ_j, v_j, and p_j respectively. Then since entropy is constant along the particle path $dx/dt = v_j$, the outflow pressure p_{out} can be computed as

$$p_{out} = \rho_{out}^{\gamma}(p_j/\rho_j^{\gamma}).$$

Let R_- and R_+ be the Riemann invariants for the Euler equations, *i.e.*,

$$R_- = v - \frac{2}{\gamma - 1}c = \text{constant} \qquad \text{along } \lambda_1 = v - c \text{ family},$$

and

$$R_+ = v + \frac{2}{\gamma - 1}c = \text{constant} \qquad \text{along } \lambda_3 = v + c \text{ family},$$

where $c = \sqrt{\gamma p/\rho}$ is the speed of sound. Using the constant Riemann invariant R_+ along $dx/dt = v_j + c_j$, we have

$$v_{out} + \frac{2}{\gamma - 1}c_{out} = v_j + \frac{2}{\gamma - 1}c_j,$$

and then combining this equation with ρ_{out} and p_{out} we can compute v_{out}. In each time step, the numerical characteristic boundary conditions are used at the outflow boundary to update v_{out} and p_{out}.

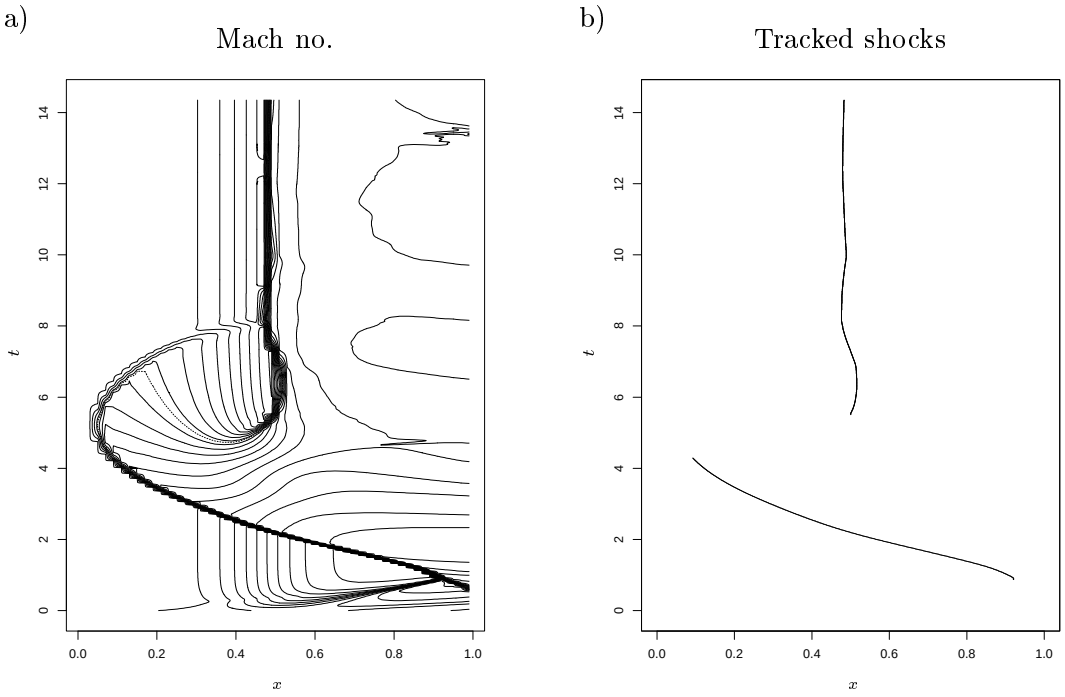


Figure 5.5: Results for the quasi one-dimensional nozzle flow. a) Mach number contour plot in the x-t plane up to time $t = 14.35$. b) Tracked shocks.

Results for this test using $h = 1/50$ and $\varepsilon = 0.3$ in density jump for tracking are shown in Figures 5.5 and 5.6. Figure 5.5a and b show the Mach number contour plot in the x-t plane and the tracked shocks, respectively. Figure 5.6 shows the converged numerical solution together with the exact steady state solution. Here our converged numerical stationary shock location is at $x = 0.481714 \pm 10^{-6}$, with an error of roughly 0.06% relative to the exact location. The numerical result (Mach number) agrees well with the exact steady state solution also.

It is interesting to note how our front tracking algorithm handles tracked waves for this problem. For this test, there is no tracked shock in the beginning. When a shock is formed, it is tracked. Since this tracked shock is not the stationary shock, the strength of the shock begins to decay due to the geometric effect and boundary conditions. After sufficient decay, it is no longer tracked. Meanwhile, a second shock forms which converges to the correct stationary shock. A similar test is performed in [34] for their front tracking method.

We should note that we have made no attempt to accelerate convergence to steady state in this code, since the current version is designed primarily for time-dependent calculations.

5.2.2 Unstable detonation waves

As a second example with source terms, we consider a simplified chemically reacting flow in which viscosity, heat conduction, diffusion, and radiation effects are ignored. We consider a model problem for combustion in which there are only two chemical species: "burnt gas" and "unburnt gas", and the unburnt gas is converted to burnt gas *via* a simple decay process

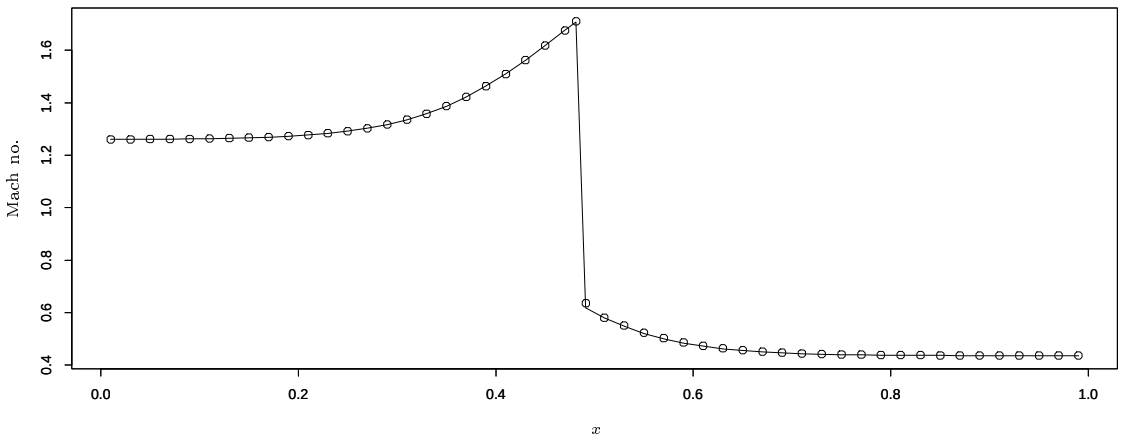


Figure 5.6: The converged numerical solution for Mach number with the exact steady state solution.

of the form

$$\text{unburnt gas} \xrightarrow{K(T)} \text{burnt gas}$$

where $K(T)$ represents the reaction rate of the burning process. This model has been extensively studied in the past, *e.g.*, [12],[22],[23],[25],[31].

In general, the reaction rate depends on the temperature T *via* some Arrhenius relation

$$K(T) = K_0 T^{\alpha} e^{-E^+/T} \tag{5.5}$$

where K_0 is the rate multiplier, E^+ is the activation energy, and α is the order of the reaction. Typically the reaction rate is very large when T is sufficiently high but negligible for small T.

For this combustion model, the Euler equations in one space dimension take the form

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho v \\ \rho E \\ \rho Z \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho v \\ \rho v^2 + p \\ (\rho E + p)v \\ \rho Z v \end{pmatrix} = -\begin{pmatrix} 0 \\ 0 \\ 0 \\ K(T)\rho Z \end{pmatrix} \tag{5.6}$$

where Z is the mass fraction of the unburnt gas ($Z = 1$ for the unburnt gas and $Z = 0$ for the burnt gas). For simplicity we assume that both the unburnt gas and burnt gas are ideal gases with the same ratio of specific heats γ. Then by the ideal gas law, the temperature is given by

$$T = p/\rho R$$

where R is the universal gas constant. The equation of state is modified by the fact that the unburnt gas contains chemical energy that is released as heat in the process of burning. The total energy per unit mass takes the form

$$E = \frac{1}{\gamma - 1} p/\rho + \frac{1}{2} v^2 + q_0 Z \tag{5.7}$$

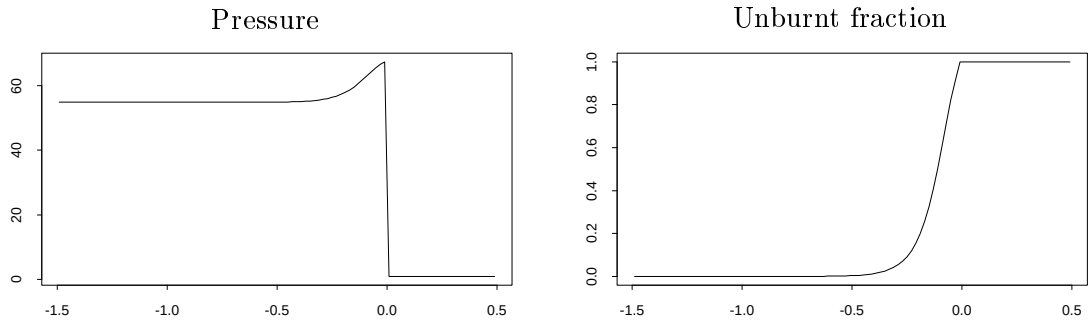


Figure 5.7: Typical ZND structures for the reaction wave in the combustion model equations (5.6) with Arrhenius rate relation (5.5).

where q_0 is the heat release.

There are two distinct types of reaction waves for this combustion model, the "detonation wave" and the "deflagration wave". For a detonation wave, the pressure and density across the wave jump to higher values, and the wave travels at supersonic speed relative to the unburnt gas in front of it. For a deflagration wave, the pressure and density across the wave jump to lower values, and the wave travels at subsonic speed relative to the unburnt gas.

With the Arrhenius rate relation (5.5), the typical structure of the detonation wave consists of an ordinary fluid dynamic shock followed by a finite length chemical reaction zone, which gives the so-called ZND (Zel'dovich-von Neumann-Döring) structure (see Figure 5.7). The steady ZND structures can be computed by dropping the time derivative terms in (5.6) and integrating the first three equations explicitly in space for a given initial state. From them and Equation (5.7), the density, velocity and pressure can be expressed in term of the mass fraction Z at any point of the ZND structure. The fourth equation, after expanding $(\rho Z v)_x$ and canceling the $Z(\rho v)_x$ term, now gives a nonlinear ODE

$$Z_x = \frac{-K(T)Z}{v}$$

for the mass fraction Z with respect to x, which can be solved numerically for Z as a function of x. Note that $K(T)$ is a function of Z also since the temperature T depends only on ρ and p. Having obtained the mass fraction Z for a given x, the remaining state variables, ρ, v, and p can then be calculated.

This steady ZND structure is uniquely determined if the speed of the ZND structure is specified[31]. In fact, for each given unburnt state, there is a minimal shock speed s_{CJ}, the speed of the Chapman-Jouguet detonation, which moves with the speed of sound with respect to the burnt gas[25], and hence in this case a ZND structure can also be determined without specifying any particular ZND speed.

A well-known difficulty in the detonation wave computation is that incorrect detonation wave speeds can arise from numerical effects. This behavior is observed by Colella, Majda, and Roytburd [22],[23] where a time-splitting method is used for the model Equations (5.6). They assert that if the chemistry is not fully resolved due to the insufficiently fine grids, incorrect detonation wave speeds will be obtained. Similar experiments have been reported

in [62] and [67].

Another difficulty of modeling detonation waves is noticed by Bourlioux, Majda, and Roytburd[12] in which classical one-dimensional stable and unstable detonation waves are tested. They conclude that false predictions of stability in the regime of physical instability as well as drastic predictions of instability for a physically stable detonation wave can be obtained with standard shock capturing methods if the grid is not sufficiently fine. In their paper, a front tracking method with adaptive mesh refinement is proposed for the detonation wave computation, and by using this method they have obtained good results. (Their method combines the piecewise-parabolic method[110] with conservative front tracking[16] and adaptive mesh refinement[6].) This provides a good comparison problem for our front tracking approach.

Let s be a speed of the given ZND structure. Then the parameter $f = (s/s_{\rm CJ})^2$ measures the degree of overdrive of the detonation wave and satisfies $f \geq 1$. Now the problem of interest is to study the large time behavior of the overdriven detonation wave for a given ZND structure under small perturbation. For comparison purposes, we choose the test cases as used in [12], *i.e.*, we take $\gamma = 1.2$, $R = 1$ (the universal gas constant), $\alpha = 0$, $q_0 = 50$, and $E^+ = 50$. With these parameters, according to the linear stability results[11], this detonation wave is unstable if the degree of overdrive f is greater than the critical value $f_c = 1.73$, for it is stable otherwise. Here we choose $f = 1.8$ for the stable detonation computation and $f = 1.6$ for the unstable detonation computation. In the tests shown below, a steady ZND structure is used as the initial data with the unburnt state $\rho = 1$, $v = 0$, $p = 1$, $Z = 1$, and degree of overdrive f. Note that by specifying the unburnt state, the minimal shock speed $s_{\rm CJ}$ can be calculated (see [100]). Then the speed of the ZND structure $s = s_{CJ}\sqrt{f}$ can be computed for a given f, and so this ZND structure is uniquely defined. The destabilizing perturbation for each test is provided automatically by the truncation error of the numerical method.

There are two characteristic length scales for this problem. They are the half reaction length $L_{1/2}$ and the half reaction time $t_{1/2}$, where $L_{1/2}$ is the distance required for half the mass fraction to be released in the ZND structure and $t_{1/2}$ is the undergoing time required for such a process to complete. These two values can be computed by evaluating the following integrals numerically:

$$L_{1/2} = -\int_{\frac{1}{2}}^{1} \frac{v\, dZ}{K(T)Z}, \tag{5.8}$$

$$t_{1/2} = -\int_{\frac{1}{2}}^{1} \frac{dZ}{K(T)Z}. \tag{5.9}$$

For the purpose of studying the grid effect on the numerical solutions, we normalize the length scale x by choosing K_0 so that $L_{1/2} = 1$. We find that $K_0 = 231.16$ and $t_{1/2} = 0.891$ for $f = 1.6$, while $K_0 = 145.69$ and $t_{1/2} = 0.856$ for $f = 1.8$.

Following [12] and [31], we monitor the shock front pressure, the pressure right behind the shock wave, as time evolves. This shock front pressure history will give us a clear indication on the stability of a given ZND structure under small perturbation. Here to investigate the grid effect on the numerical solutions, a convergence study for the shock front pressure history with three different coarse-fine grid spacings is performed for each stable and unstable case. The coarse-fine grid spacings we used are as follows:

1. coarse mesh $h_c = 1$ point $/L_{1/2}$, fine mesh $h_f = 4$ points$/L_{1/2}$,
2. coarse mesh $h_c = 2$ points$/L_{1/2}$, fine mesh $h_f = 8$ points$/L_{1/2}$,
3. coarse mesh $h_c = 4$ points$/L_{1/2}$, fine mesh $h_f = 16$ points$/L_{1/2}$.

Since there is only one tracked shock in this problem, the refinement region is chosen by going out $20L_{1/2}$ on each side of the tracked shock. Courant number $\nu = 0.5$ is used for all the test cases, and only results obtained using the high resolution method are shown. Tolerance $\varepsilon = 3$ in density jump is used for shock tracking.

Figures 5.8 and 5.9 show results for the convergence study up to time $t = 100$ in the computational domain $0 \leq x \leq 1000$. After analyzing the results, our solution converges to the initial steady state profile with about 0.008% oscillation in the shock front pressure for the stable detonation case. For the unstable detonation case our solution converges to a detonation wave with period 7.383 ± 0.110 (about $8.284t_{1/2}$) and peak pressure 99.83 ± 0.2. Note that the unperturbed shock front pressure for the unstable detonation wave is 67.355, and so the shock front pressure is magnified to a value nearly 50% higher than the initial value. Our solutions agree very well with values taken from the figures in [12].

To show the spatial resolution for the unstable detonation wave problem, we plot the pressure at six different times within one complete pressure front oscillation cycle as illustrated in the second plot of Figure 5.9, where the large dots indicate the plotting time. The results are shown in Figure 5.10 for both the coarse and fine grid solutions, where a region between the dashed lines above the x-axis is the mesh refinement region. A blow-up of the solution in the fine grid region is shown in Figure 5.11. Note that an oscillatory wave structure appears behind the shock. This is not seen in the stable detonation problem.

Figure 5.12 shows the shock speed as a function of time in the unstable case, also showing periodic oscillatory behavior. An earlier results given by Fickett and Wood[31] shows that under certain assumptions the averaged shock speed $\bar{s}$, in both stable and unstable detonation waves, should essentially remain the same as the steady-solution shock speed. In our calculations, we have observed that for the unstable detonation wave problem the time-averaged shock speed $\bar{s}$ is equal to 8.655, while the steady-solution shock speed is 8.613. So there is about 0.5% discrepancy from the predicted value. On the other hand, for the stable detonation wave problem $\bar{s}$ is 9.1369, while the steady-solution shock speed is 9.1359.

Finally, Figure 5.13 shows the result in the regime of transition to instability. We also observe good agreement with the linear stability result. Recall that for the parameters we used here $f = 1.73$ is the critical value for stability of detonation waves.

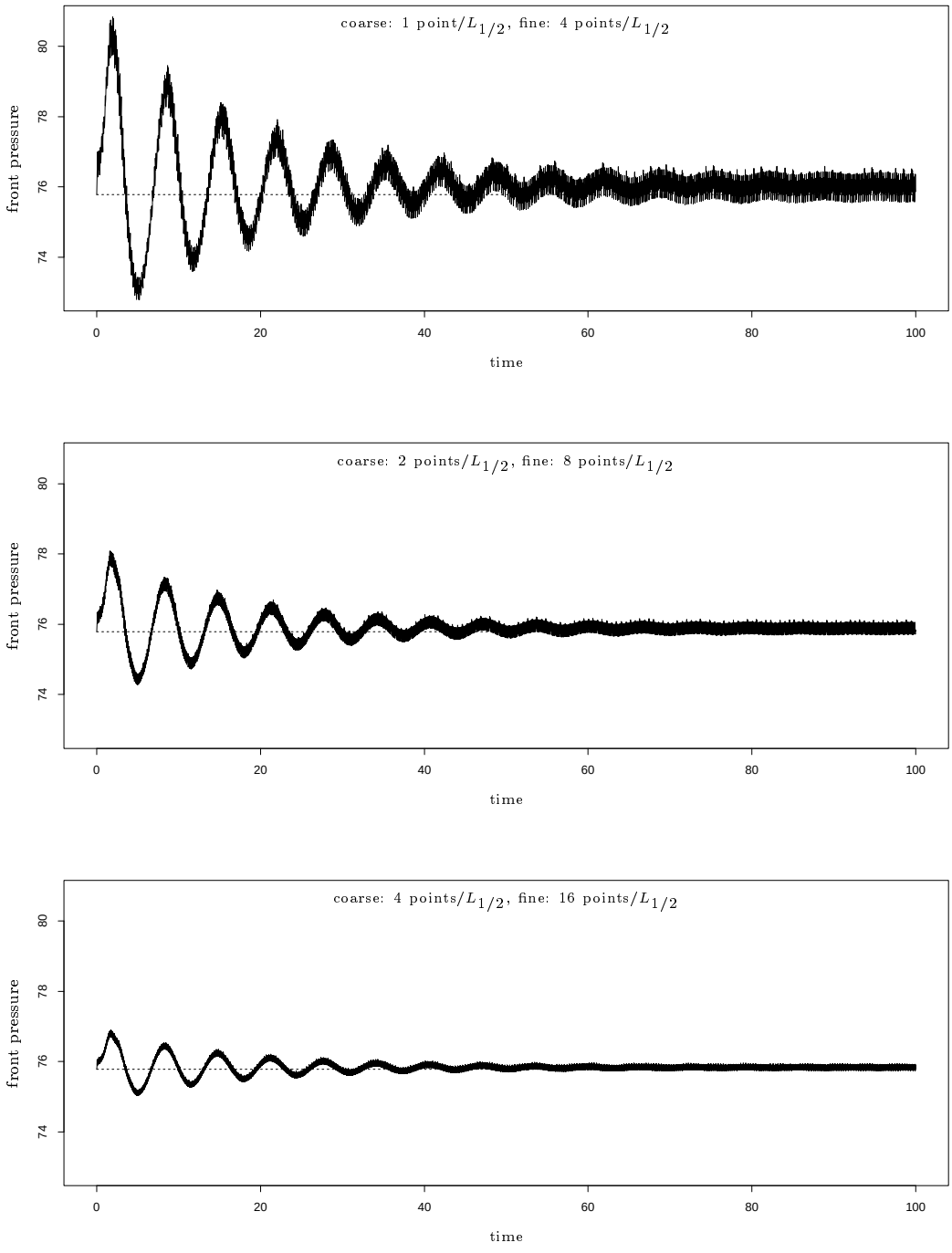


Figure 5.8: A convergence study for the shock front pressure history on the stable detonation wave problem ($f = 1.8$) using the high resolution front tracking with adaptive mesh refinement algorithm. The dashed line shown in the figure is the front pressure of the steady ZND solution.

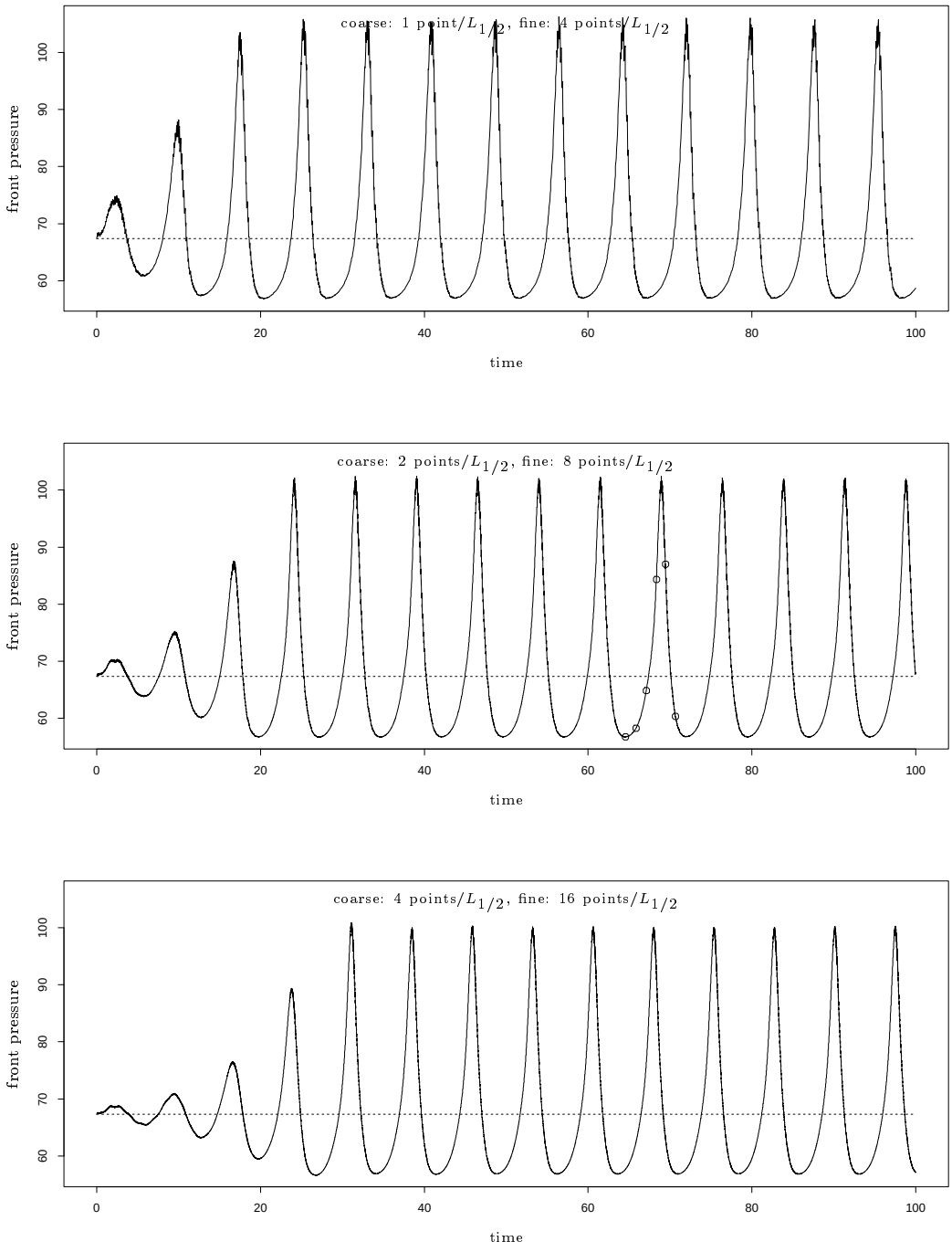


Figure 5.9: A convergence study for the shock front pressure history on the unstable detonation wave problem ($f = 1.6$) using the high resolution front tracking with adaptive mesh refinement algorithm. The dashed line shown in the figure is the front pressure of the steady ZND solution.

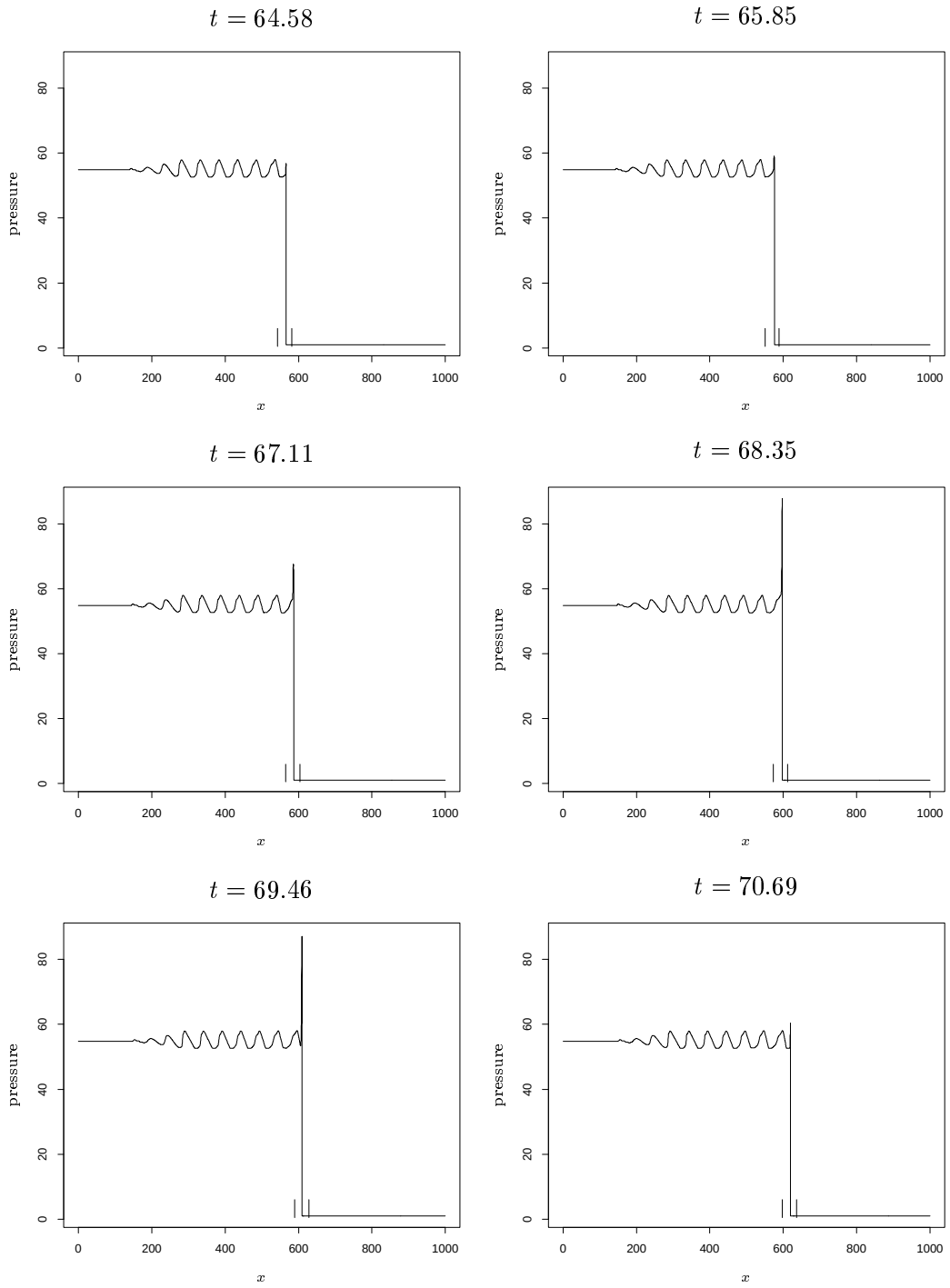


Figure 5.10: Spatial resolution for the unstable detonation problem at six different times (pressure is shown) using the high resolution method with mesh size $h_c = 2$ points/$L_{1/2}$, $h_f = 8$ points/$L_{1/2}$. Note that a region between the dashed line above the x-axis is the refinement region.

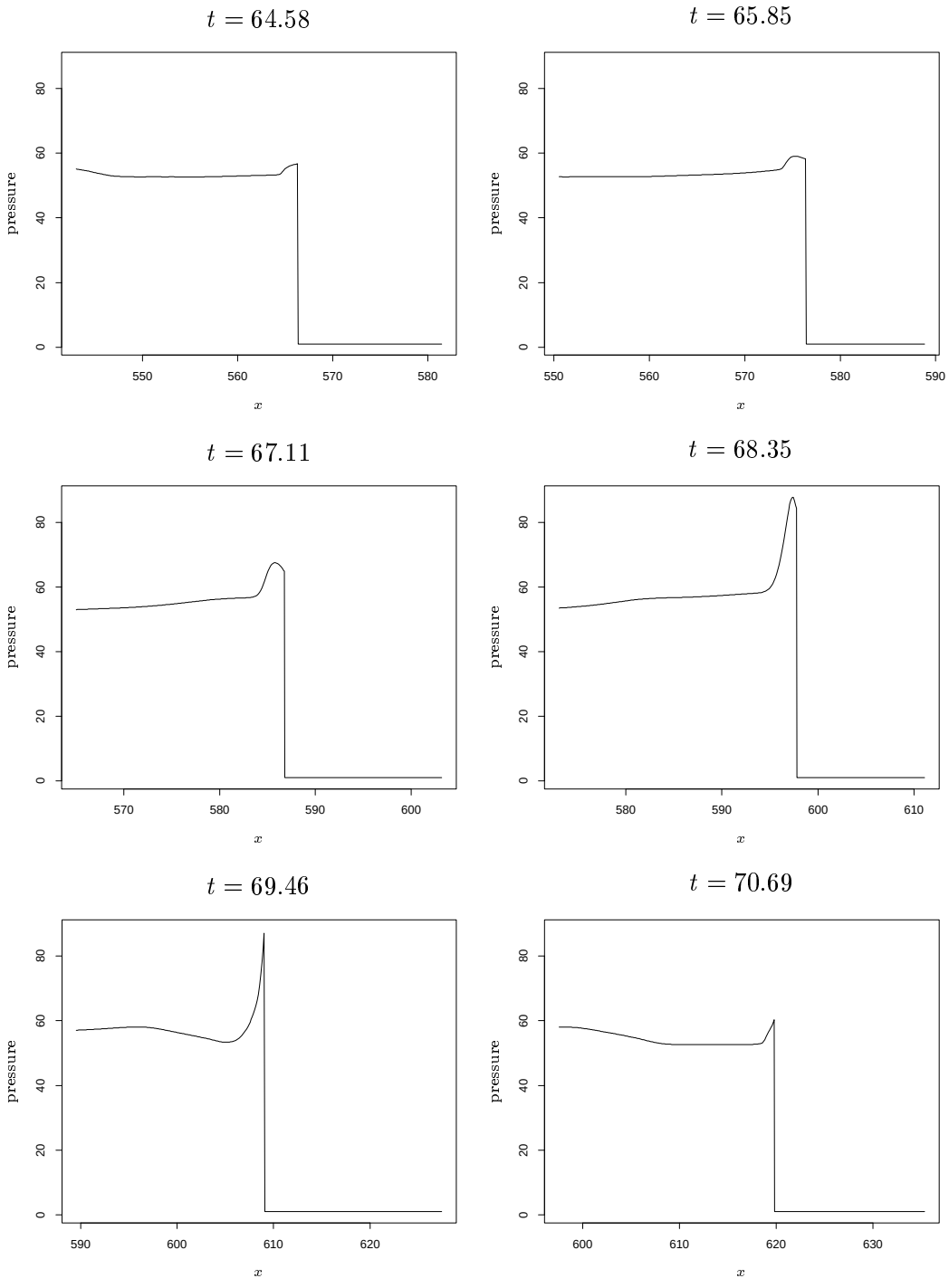


Figure 5.11: Blow up of the fine grid region for the solution shown in Figure 5.10.

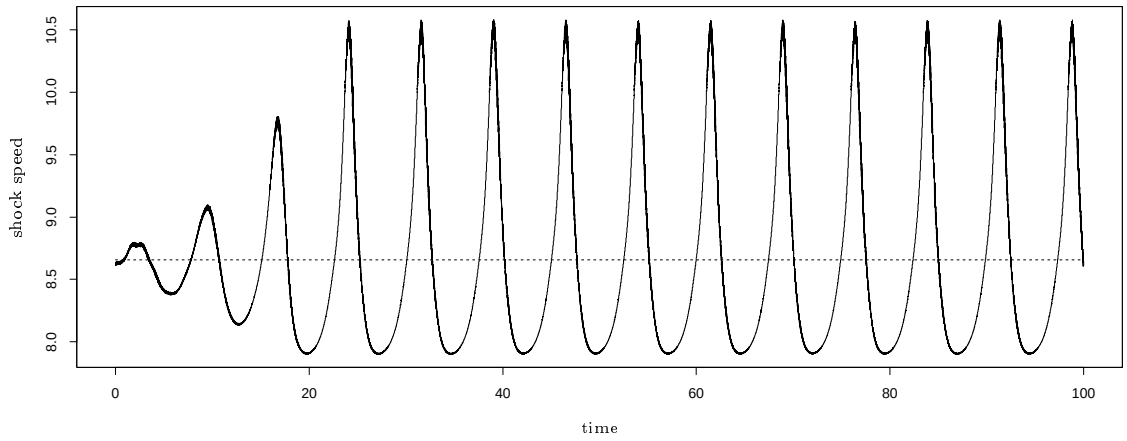


Figure 5.12: Shock speed history for the unstable detonation wave problem using the coarse-fine grid spacing: $h_c = 2$ points/$L_{1/2}$, $h_f = 8$ points/$L_{1/2}$, where the dashed line is the time-averaged shock speed.

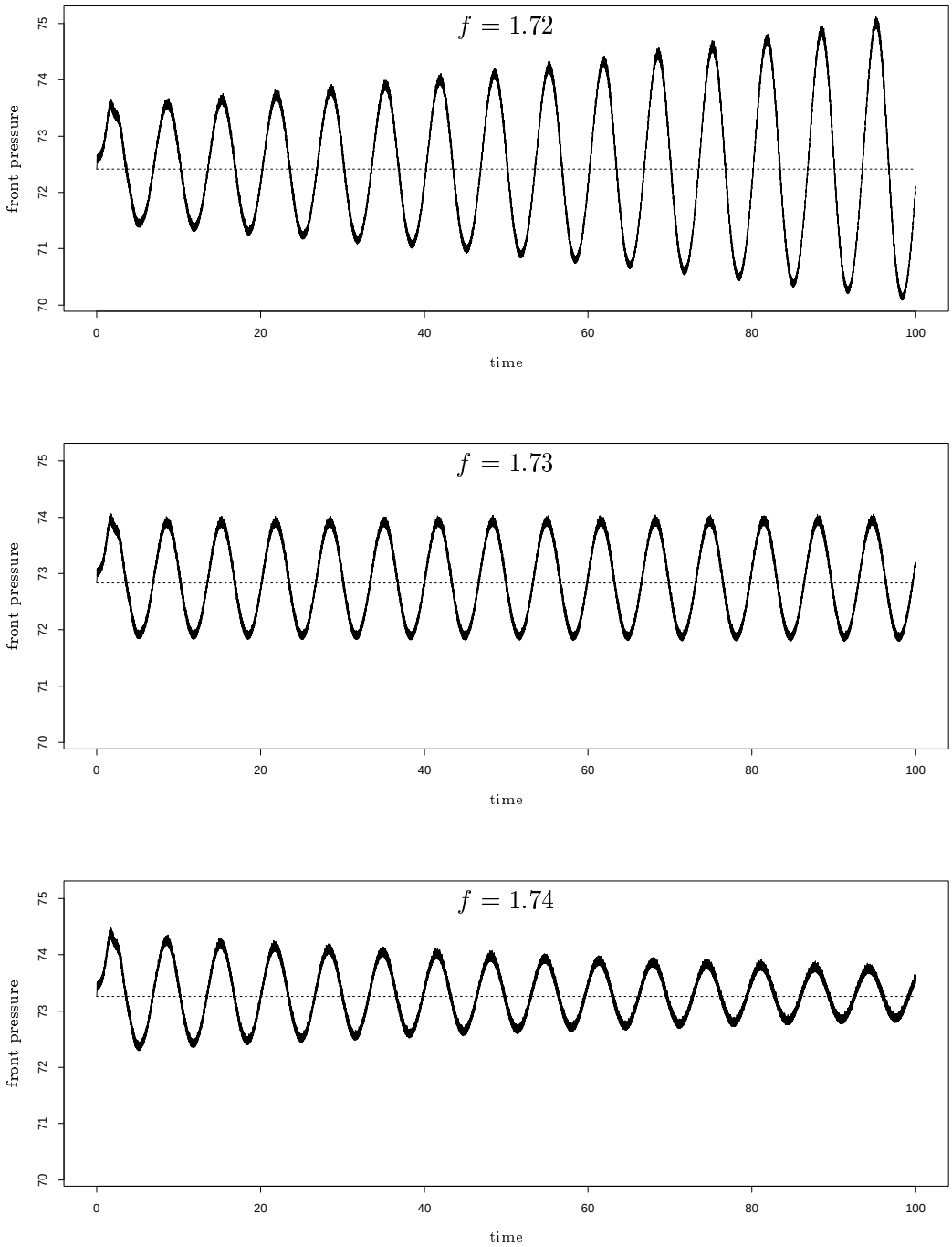


Figure 5.13: A numerical study for the detonation wave problem in the regime of transition to instability. In the case we considered here $f = 1.73$ is the critical value for stability of detonation waves.

Part II

Two Space Dimensions

Chapter 6

FINITE VOLUME WAVE PROPAGATION METHODS

Here we begin by considering numerical methods that can be used to compute the solution for the homogeneous conservation laws

$$u_t + f(u)_x + g(u)_y = 0 \tag{6.1}$$

on the grid determined by the front tracking algorithm. We again describe methods based on the wave propagation approach, because of the ease of dealing with small cells and boundary conditions in a stable manner. The approach we present here follows closely ideas from the one-dimensional wave propagation approach discussed in the previous chapter, though some modification of the methods will be considered to take into account the two-dimensional effects, particularly for the high resolution methods. In the discussions, special attention will be given to the treatments of the irregular cells, and a rotated Godunov method will also be explained.

6.1 Preliminaries

We describe the methods on a very special grid as illustrated in Figure 6.1, in which additional cell interfaces are introduced for the tracked discontinuities in a uniform underlying Cartesian grid subdividing some cells into pieces. Let C_j denote each grid cell, and let I_{ij} denote the cell interface between cells C_i and C_j. We use a finite-volume formulation in which the value U_j^n approximates the cell average of the solution over the grid cell C_j at time t_n,

$$U_j^n \approx \frac{1}{A_j} \int_{C_j} u(x, y, t_n)\, dx\, dy$$

where A_j is the area of the cell C_j. For convenience, the additional cell interface is called an *irregular cell interface* to distinguish it from the regular cell interfaces, and the cell it subdivides is called an *irregular cell* to distinguish it from the regular cells.

The methods we use are based on solving one-dimensional Riemann problems at each cell interface. Consider, for example, the one-dimensional Riemann problem normal to the interface separating cells C_j and C_l in Figure 6.1. Define the new variables ξ (normal to the interface) and η (tangential to the interface) by

$$\xi = \alpha x + \beta y, \qquad \eta = -\beta x + \alpha y \tag{6.2}$$

with $\alpha = \cos\theta$, $\beta = \sin\theta$, where θ is the angle of the cell interface. Then the conservation laws (6.1) can be written as

$$u_t + \hat{f}(u)_\xi + \hat{g}(u)_\eta = 0 \tag{6.3}$$

with

$$\hat{f}(u) = \alpha f(u) + \beta g(u), \qquad \hat{g}(u) = -\beta f(u) + \alpha g(u).$$

Assume that in the new coordinates u is constant in η, and so (6.3) reduces to a one-dimensional Riemann problem

$$u_t + \hat{f}(u)_\xi = 0 \tag{6.4}$$

with left and right states U_j and U_l. For rotationally invariant equations such as the Euler equations this is particularly simple since, after a change of dependent variables to rotate the velocity field, the form of $\hat{f}(u)$ agrees with $f(u)$ and hence a single Riemann solver suffices for all angles θ.

As in the one-dimensional method, we use Roe's approximate Riemann solver and obtain a set of waves traveling with speeds $\lambda_1, \lambda_2, \cdots, \lambda_m$ in the ξ-direction. As before, we denote the jumps in u across these waves by the vectors $r_1, r_2, \cdots, r_m$, so that

$$U_j - U_l = \sum_{p=1}^{m} r_p.$$

Finite volume wave propagation methods are based on using these propagating discontinuities to update the cell averages in the cells neighboring each interface.

6.2 Godunov Method

In a standard finite volume method, fluxes across the cell interfaces are defined and used to update the cell values on either side of the interface. In particular, in the Godunov method, fluxes are computed based on solving the Riemann problems at each interface in the direction normal to the interface over a time step of length k. Then a conservative flux-differencing method is used to obtain the solution at the next time step U^{n+1}, see [63] for more general discussion on the conservative flux-differencing methods.

Following convention, the normal direction for a regular cell interface is defined in the usual manner pointing to the positive x- or y-direction. At an irregular cell interface it is defined by the following rules: if the irregular interface represents a boundary segment, the normal direction is chosen pointing toward the interior region, while if it represents a tracked discontinuity, the normal direction is chosen pointing to a state with lower density or other physically meaningful quantity, such as entropy.

A first order accurate version of the finite volume wave propagation method is a variant of the Godunov method, with the Roe Riemann solver, on a nonuniform grid. That is to say, we solve the Riemann problems at each interface in the direction normal to each interface as well, but now waves which result from solving the Riemann problems are propagated over the time step k to update whichever cell values they affect. Since this approach has been discussed fully in the past [61],[62], here we only briefly describe the method.

Figure 6.1a shows an example in which waves are propagated from the regular and irregular cell interfaces. Then in the method the cell average U_j^n is updated by

$$U_j^{n+1} = U_j^n - \left(\frac{h_{ij} |\lambda_p| k}{A_j} \right) r_p$$

a) b)

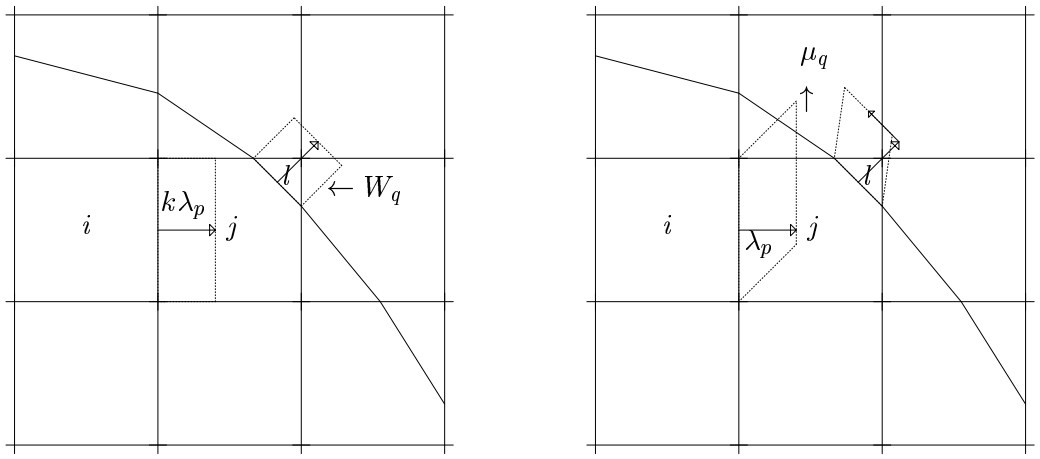


Figure 6.1: a) Godunov method. Waves obtained from solving the normal Riemann problem at each interface is used to update the cell values (only one family of waves is shown). b) Godunov method with tangential splitting. Waves shown in a) are split into subwaves in the tangential direction, and used to update the cell values (only one subwave is shown for each wave family).

since $h_{ij}|\lambda_p|k$ is the area the wave sweeps out over the time step k, and so the value should be changed by the jump r_p in this portion of the cell. Similarly, due to the q-wave, the cell average U_l^n should be updated by

$$U_l^{n+1} = U_l^n - \left(\frac{\text{Area}(W_q \cap C_l)}{A_l}\right) r_q \tag{6.5}$$

where W_q is the region that the q-wave affects (the rectangular region in the figure), and $\text{Area}(W_q \cap C_l)$ is the area of the intersection between W_q and cell C_l. Note that other cells which are affected by the q-wave should also be modified by the appropriate amount. By computing the effect of each wave on the cell average, we obtain the finite volume wave propagation Godunov method.

As in the one-dimensional method, it is also possible to reformulate this method in a standard conservation form by computing fluxes appropriately at each interface. In fact, if we do so, on a uniform grid the resulting method is equivalent to the standard Godunov method. On a nonuniform grid, however, the form of the method becomes very complicated due to the fact that waves may cross many cell interfaces as seen in Figure 6.1a. This is also the case in dealing with the boundary conditions. Because of the complications we do not discuss this flux formulation here.

Regarding stability, it is well known that on a uniform grid in two space dimensions the Godunov method has a Courant number restriction $\nu_0 \leq 1/2$ where

$$\nu_0 = \frac{k}{h_0} \max_{p,j} |\lambda_{pj}|; \tag{6.6}$$

$h_0 = \min(h_x, h_y)$ is the minimum mesh size of the uniform grid cell in both the x- and y-directions. On a nonuniform grid, the Courant number is defined by

$$\nu = \frac{k}{A_{\min}} \max_{p,j} |\lambda_{pj}| \tag{6.7}$$

where $A_{\min} = \min_j A_j$. Note that because of the presence of irregular cells, the time step k is severely reduced if $A_{\min} \ll h_0$. From numerical experience, because the waves are allowed to affect more than one neighboring cell, the wave propagation version of the Godunov method is more stable than the standard flux version Godunov method, and $\nu \leq 1/2$ is not required on the nonuniform grid. Instead we can use the Courant number $\nu_0 \leq 1/2$ with mesh size based on the underlying uniform grid.

6.3 Godunov Method with Tangential Splitting

Another variant of the Godunov method is the Godunov method with tangential splitting introduced by LeVeque[61],[65] in which the waves obtained from solving the normal Riemann problem at a cell interface are split into subwaves in the tangential direction with appropriate tangential speeds. In this approach, a tangential Riemann problem is solved for each wave using the data on the left and on the right of the wave. Since this approach has been discussed in more detail in [61], here we only briefly describe the method.

Figure 6.1b shows an example in which we split the p-wave from the I_{ij} interface. For the equation, we take the tangential portion of (6.1), *i.e.*, the portion of equation in the y-direction:

$$u_t + g(u)_y = 0, \tag{6.8}$$

and for the initial data we take

$$U_i^n + \sum_{q<p} r_q$$

as the left state and

$$U_j^n - \sum_{q>p} r_q$$

as the right state. Taking these as the left and right states for the equation (6.8) gives a discontinuity of magnitude r_p, and the solution will be resolved into waves $w_1, w_2, \cdots, w_m$ propagating in the y-direction with speeds $\mu_1, \mu_2, \cdots, \mu_m$ as illustrated in Figure 6.1b. The splitting of waves in the x-direction can also be dealt with in the similar manner. By computing the effect of each tangential wave on the cell average, we obtain the Godunov method with tangential splitting.

Note that with this tangential splitting the method remains conservative because the total contribution of the subwaves satisfies

$$\sum_{q=1}^{m} w_q = r_p, \tag{6.9}$$

and the area swept out by each subwave is the same as the area of the original wave. Moreover, doing so approximates the transverse derivative $(BAu_x)_y$, since

$$A(U_j^n - U_i^n)/h \approx Au_x(x_{i+1/2}, y_j, t_n)$$

and splitting waves in the tangential direction gives an approximation to

$$BAu_{xy}(x_{i+1/2}, y_{j+1/2}, t_n)$$

which is a linearized version of the term. Analogously, the splitting of waves in the x-direction gives an approximation of the $(ABu_y)_x$ term. We will see in Section 6.5 that the ability of handling the $(BAu_x)_y$ and $(ABu_y)_x$ terms is an essential step toward achieving high resolution.

It is not difficult to verify that for the linear advection equation

$$u_t + au_x + bu_y = 0 \tag{6.10}$$

this method gives exact propagation of waves for any time step k, except for the error introduced by the averaging process. For linear hyperbolic systems

$$u_t + Au_x + Bu_y = 0 \tag{6.11}$$

this is also the case, if A and B have identical eigenvectors and hence are simultaneously diagonalizable in which case A and B commute, $AB = BA$. In this instance, the equations can be decoupled into different characteristic fields, and by employing the wave propagation approach, this method can be viewed as the method of characteristics since here each wave family is propagated exactly for any time step k and only averaging error is introduced in the method. If the matrices A and B do not commute, or for nonlinear problems, this method, in general, will not produce results that are stable for arbitrarily large time steps. However, all of our numerical results indicate that the method is stable for the Courant number ν_0 up to 1.

6.4 Rotated Godunov Method

In a rotated Godunov method, we solve the Riemann problems in some physically relevant directions rather than the directions normal to the grid interfaces. Various approaches have been introduced in the past that determine the rotation direction as well as the way that fluxes are computed in the method. Typical examples of the rotation direction are: the flow direction, the pressure-gradient direction, the velocity-magnitude-gradient direction, and the direction of the irregular cell interface[8],[27],[68]. In the flux version of the rotated method, it is quite often the case that the Riemann data is obtained from some form of interpolation of the cell values to maintain stability and achieve high accuracy of the method, see [7],[68] for examples.

The idea of the rotated method is best explained by considering the linear advection equation (6.10), and describing the method based on the wave propagation approach on a uniform grid. As the first step in the method, we need to choose the rotation direction θ and transform (6.10) to the new ξ-η coordinate system using (6.2), which leads to

$$u_t + \hat{a}u_\xi + \hat{b}u_\eta = 0 \tag{6.12}$$

a) b)

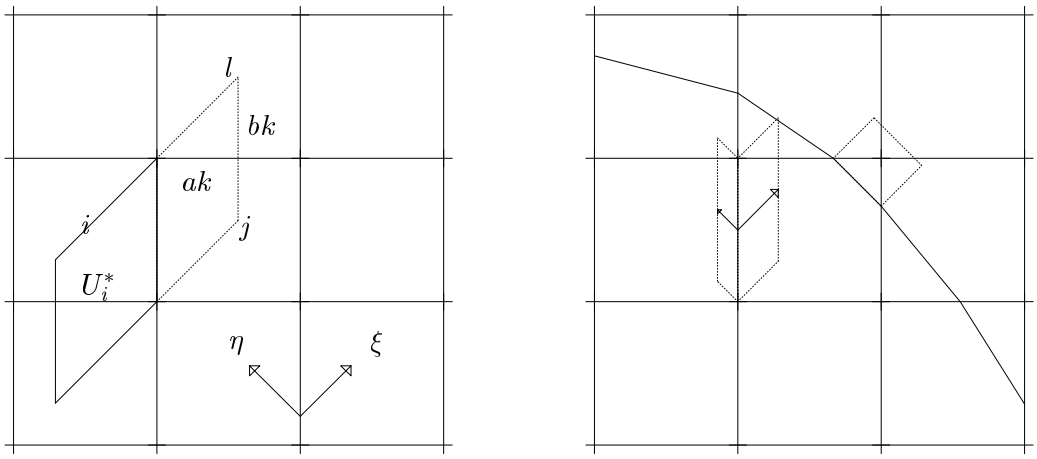


Figure 6.2: Rotated Godunov method. a) Waves obtained from solving the Riemann problem in the flow direction (a, b) of the linear advection equation (6.10) is used to update the cell values. An interpolated value U_i^* obtained from taking averages of cell values over the parallelogram box, is used in a flux version rotated scheme. b) On the irregular cell, the rotation direction based on the direction of the irregular cell interface is used in solving the Riemann problem for the conservation laws (6.1) (only one family of waves is shown).

with $\hat{a} = \alpha a + \beta b$ and $\hat{b} = -\beta a + \alpha b$. (Recall that $\alpha = \cos\theta$, $\beta = \sin\theta$.) For this model equation (6.10), it is clear that we should use the flow direction (a, b), or the solution gradient (u_x, u_y), in the normalized form as the rotation direction in the method. Suppose that we take the normalized flow direction as the rotation direction for the Riemann problem, (6.12) simply reduces to the one-dimensional linear advection equation in the flow direction (ξ-direction):

$$u_t + \hat{a} u_\xi = 0 \tag{6.13}$$

with $\hat{a} = \sqrt{a^2 + b^2}$. Then in the second step of the method, we solve the Riemann problem for (6.13) at each interface using the values from adjacent cells as data. Doing so results in waves propagating with speed $\hat{a}$ in the ξ-direction with a shape shown in Figure 6.2a. The cells it overlaps are updated during the wave propagation.

Figure 6.2a shows an example in which the cell average U_j^n is updated by

$$U_j^{n+1} = U_j^n - \frac{ak}{h^2}(h - bk/2)(U_j^n - U_i^n),$$

while the cell average U_l^n is updated by

$$U_l^{n+1} = U_l^n - \frac{abk^2}{2h^2}(U_j^n - U_i^n)$$

where h is the mesh side in both the x- and y-directions. By computing the effect of each wave on the cell average, we obtain the rotated Godunov method *via* wave propagation. It

is easy to see that this method gives exact propagation of waves for any time step k, except for the error introduced by the averaging process. In fact, for this model problem, the result obtained from this method is identical to that obtained from the Godunov method with tangential splitting.

It is worth noting that for this model problem, in the standard flux version rotated Godunov method, the flux at the I_{ij} interface is computed as

$$F(U_i^n, U_j^n) = aU_i^n \tag{6.14}$$

if the values from adjacent cells are used as Riemann data, whereas the flux is computed as

$$F(U_i^*, U_j^*) = aU_i^* \tag{6.15}$$

if the values from some interpolation method are used as Riemann data, see Figure 6.2a for illustration. It is easy to see that flux (6.14) is simply the flux obtained from the unrotated Godunov method, while flux (6.15) is the modified version of the flux (6.14) which takes more information from the upwind direction. Evidently, no matter which flux is used in practice, the resulting flux differencing method will not produce results that give the exact propagation for any time step k as in our wave propagation method.

On uniform grids, the extension of the rotated wave propagation method to linear hyperbolic systems (6.11) is straightforward if A and B commute, since we can take each characteristic direction as the rotation direction for the Riemann problem and propagate the resulting wave as the way we did for the linear advection equation. If A and B do not commute or for nonlinear problems, this method is still applicable, but now the rotation direction should be chosen on a problem by problem basis and will vary from cell to cell.

The extension of the method to nonuniform grids can be made quite easily also, but now a formula similar to (6.5) should be used to update the cell values, see Figure 6.2b for illustration. It can be demonstrated that by propagating waves exactly and allowing waves to affect the neighboring cells, this method is not only stable in the presence of small cells, but also with Courant number ν_0 up to 1, for most of the equations of practical interest.

It should be noted that, in general, for conservation the cell values should be updated not only based on the solutions of the rotated Riemann problem in the ξ-direction, but also based on the Riemann solutions in the η-direction as well.

6.5 High Resolution Godunov Method

Here we discuss the high resolution modifications of the Godunov method. To illustrate the idea, let us look at the derivation of the Lax-Wendroff method for the conservation laws (6.1) on a uniform grid. In deriving the Lax-Wendroff method, we start with the Taylor series expansion

$$u(x, y, t + k) = u(x, y, t) + ku_t(x, y, t) + \frac{k^2}{2}u_{tt}(x, y, t) + \cdots. \tag{6.16}$$

From the governing equation (6.1) we can compute

$$u_t = -f(u)_x - g(u)_y = -Au_x - Bu_y$$

and

$$\begin{aligned} u_{tt} &= (-f(u)_x - g(u)_y)_t = -f(u)_{tx} - g(u)_{ty} = -(Au_t)_x - (Bu_t)_y \\ &= (A^2u_x)_x + (ABu_y)_x + (BAu_x)_y + (B^2u_y)_y, \end{aligned}$$

where $A = \partial f(u)/\partial u$ and $B = \partial g(u)/\partial u$, so that (6.16) becomes

$$\begin{aligned} u(x,y,t+k) &= u(x,y,t) - k(Au_x + Bu_y)(x,y,t) + \frac{k^2}{2}((A^2u_x)_x + \\ &\quad (ABu_y)_x + (BAu_x)_y + (B^2u_y)_y)(x,y,t) + \cdots \end{aligned} \tag{6.17}$$

The Lax-Wendroff method then results from retaining all the terms up to $O(k^2)$ and using centered difference approximation for the derivatives appearing there.

From (6.17), it is clear that to achieve second order accuracy we need to deal with the second order derivative terms $(A^2u_x)_x$, $(ABu_y)_x$, $(BAu_x)_y$, and $(B^2u_y)_y$. Here the approach we use follows ideas from the previous work of LeVeque on wave propagation methods[61] in that we introduce piecewise linear approximations to the solution in place of the piecewise constant functions in Godunov's method and handle transverse derivatives by splitting waves in the direction tangential to the cell interface. On uniform grids, this method is in fact very similar to the unsplit multi-dimensional upwind method of Colella[21] as discussed in [65]. The approach we employ here, however, has the advantage of easy extension in dealing with the irregular cells.

We begin our method by solving the Riemann problems in the direction normal to the cell interface as before, using the piecewise constant data. The resulting waves are then split into subwaves using the tangential splitting approach discussed in Section 6.3. As mentioned previously, doing so gives an approximation of the transverse derivative terms $(ABu_y)_x$ and $(BAu_x)_y$.

To handle the $(A^2u_x)_x$ and $(B^2u_y)_y$ terms, we use the approach similar to our one-dimensional high resolution method in that a slope is introduced for each wave and used to construct the piecewise linear wave in place of the piecewise constant wave.

On the regular cells, slopes and piecewise linear waves can be defined quite easily in both the x- and y-directions. Let σ_{pi} be the slope vector used in the pth family over the C_i cell. In this case, as in the one-dimensional method, σ_{pi} can be obtained easily from using either the unlimited slope (2.7) or a slope limiter (2.8). Then, with this slope σ_{pi} the piecewise linear wave, moving in the x-direction, is now made as a three-dimensional profile that is constant in y and piecewise linear in x over the grid cell C_i, as illustrated in Figure 6.3a.

To modify the cell values, this piecewise linear wave is advanced with speed λ_p, obtained from solving the normal Riemann problem at the I_{ij} interface, over the time step k, and the cells it overlaps are updated. For example, cells U_i^n and U_j^n are updated by

$$\begin{aligned} U_i^{n+1} &:= U_i^{n+1} - \left(\frac{|\lambda_p|k(h - |\lambda_p|k)h}{2A_i}\right)\sigma_{pi}, \\ U_j^{n+1} &:= U_j^{n+1} + \left(\frac{|\lambda_p|k(h - |\lambda_p|k)h}{2A_j}\right)\sigma_{pi}, \end{aligned}$$

where $\frac{1}{2}|\lambda_p|k(h - |\lambda_p|k)\sigma_{pi}h$ is the volumetric region that the piecewise linear wave overlaps the grid cell. Piecewise linear wave propagation in the y-direction can be handled in an

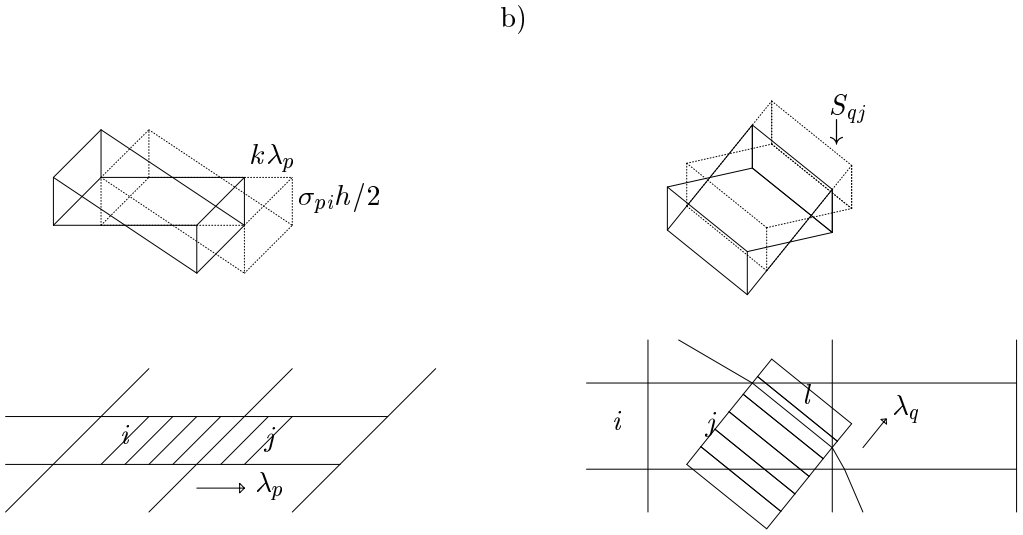


Figure 6.3: Piecewise linear wave propagation. a) Wave arising from the interface between the regular cells C_i and C_j in the x-direction. b) Wave arising from the interface between the irregular cells C_j and C_l in the ξ-direction.

analogous manner. It is easy to check that doing so gives an approximation to the linearized version of the $(A^2u_x)_x$ and $(B^2u_y)_y$ terms, and the method remains conservative with any choice of slopes.

On the irregular cells, say for the C_j cell shown in Figure 6.3b, one simple choice of slope is to take the unlimited slope

$$\sigma_{qj} = r_q/h_* \tag{6.18}$$

where h_* is some measure of the normal distance between two cells, such as the difference in the ξ-coordinate (normal to the irregular cell interface) of the centers of mass of these two neighboring cells sharing the same interface. Then a piecewise linear wave with this slope in the ξ-direction and total integral zero can be made and propagated in the ξ-direction with velocity λ_q. This is illustrated in Figure 6.3b where a piecewise linear wave is constructed for the C_j cell of the q-wave arising from the interface between cells C_j and C_l. Similarly, for waves arising from the other side of the irregular cell, piecewise linear waves can also be constructed.

More generally, if we want to use slopes based on a slope limiter, we may need to employ an interpolation scheme that determines another slope to compare with in the slope limiter (2.8). For instance, for the example shown in Figure 6.3b, one possible approach is to first compute cell values by averaging the neighboring cell values on two artificial cells of size $h \times h_*$ each, solve the Riemann problem in the ξ-direction using this data, and then take the resulting jump in the pth family divided by h as the slope used for the limiter. Note that special attention needs to be taken to avoid using information from the opposite side of the irregular interface, *i.e.*, across the shock, and so, as in one dimension, we can do slope limiting based on a one-sided differencing formula.

As usual, cell values on the irregular cells are updated by the propagation of piecewise

linear waves. Since this amounts to computing the intersection between three-dimensional wave structures and grid cells, we write the formula for updating the cell values in a descriptive form

$$U_j^{n+1} := U_j^{n+1} + \frac{\text{Volume}(S_{qj} \cap C_j)}{A_j} \tag{6.19}$$

where S_{qj} is the piecewise linear wave in the q-family for the C_j cell, and Volume$(S_{qj} \cap C_j)$ is the volume of the intersection between S_{qj} and C_j. For the example shown in Figure 6.3b, this formula should be applied to not only the two neighboring cells of the wave, C_j and C_l, but to other cells where the wave S_{qj} is affected. Notice that the work involved in the method increases a great deal by introducing this piecewise linear approximations to the irregular cells. Naturally, it would be desirable to find a better way to do this. The performance of the method and other approaches are still under investigation.

Chapter 7

FRONT TRACKING ALGORITHM

Having described the numerical methods that can be used on a grid which contains tracked discontinuities for the conservation laws (6.1), we now discuss the front tracking algorithm for this system. We will see from the discussion that this algorithm is in spirit similar to our one-dimensional front tracking algorithm, and is also very simple and robust. Here we will focus our attention on describing ideas of advancing tracked fronts from one time step to the next. Some possible approaches to setting up the data structure will also be discussed. Results obtained using this front tracking algorithm for radially symmetric shock waves will be presented for the Euler equations, and the implementation of the solid wall boundary conditions for this model system will be discussed. For ease of comparison, the format of this chapter is organized analogous to the one-dimensional counterpart Chapter 3.

7.1 Algorithm

As in the one-dimensional front tracking algorithm, our grid consists of two parts. We choose a uniform underlying grid that remains fixed for all time, and we also introduce tracked interfaces which vary from step to step for the discontinuities in the flow field. These tracked interfaces subdivide some regular cells into two or more subcells, creating some irregular cells. We then view the union of the regular cells and irregular cells as our global grid. In each grid cell, the cell average is denoted by U_j^n.

For the representation of the tracked interfaces, we use the simplest piecewise linear approach in which an interface is represented by a straight line within each cell that is formed by connecting two *point*s lying on the underlying fixed grid; a *point* is an (x, y) location in the computational domain. In addition, the interfaces belonging to the same discontinuity are joined together into a continuous piecewise linear curve as shown in Figure 7.2, for example. We assume that the curve does not cross itself or the other curves.

In each time step our front tracking algorithm consists of the following steps:

Algorithm 7.1

1) Determine the new location of the tracked interfaces at the next time step.

2) Insert these new tracked interfaces into the grid. Some cells will be subdivided and the values in each subcell must be initialized.

3) Take a time step on this nonuniform grid using a finite volume method described in Chapter 6 to update the cell averages.

4) Delete the old tracked interfaces from the previous time step. Some subcells will be combined, and a value in the combined cell must be determined from the subcell values.

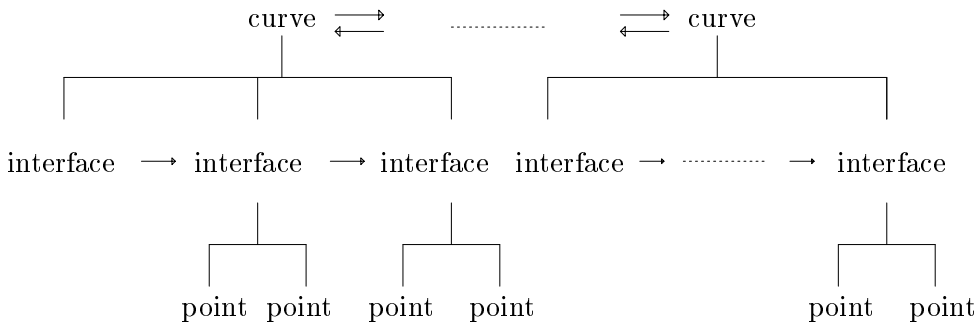


Figure 7.1: Data structure of the tracked interfaces in the front tracking code.

From the algorithm, it is clear that we have to deal with both the grid and tracked interfaces to a great extent. So before describing each of these steps in more detail, we first discuss some possible approaches to setting up the data structure. Some terms will be defined accordingly and used later on.

For the data structure of the grid, we can use an approach analogous to our one-dimensional front tracking code by employing a standard representation for the fixed grid together with a flag for each grid cell that indicates whether the grid cell is subdivided by one or more tracked interfaces. For subdivided cells, this flag is a pointer to another data structure containing information on each subcell.

There are, however, two possibilities in defining the data structure of the subdivided cells. The first possibility is to view the old and new tracked interfaces as being located in the same grid cell. There is no distinction between the grid system before and after inserting the new tracked interfaces. This is a reasonable approach as motivated by our one-dimensional algorithm, and in practice this approach works fine for problems consisting of well separated old and new tracked interfaces. But it turns out that this grid setup is of limited use because of the fact that the old and new tracked interfaces might cross each other as seen in many problems, *e.g.*, Figures 8.6 and 8.7, causing an unnecessary complication of the data structure.

The second, more reliable, approach is to view the old and new tracked interfaces as being located in two different grid systems; the old grid system and the new grid system. The old grid system contains only the old tracked interfaces, and the new grid system contains only the new tracked interfaces. The global information of the grid can still be recovered by maintaining a flag for the grid cell which encloses both the old and new tracked interfaces. Given these two distinct grid systems, there is no problem dealing with old and new tracked interfaces which cross.

Concerning the data structure of the tracked interfaces, one simple approach is to use a tree-like structure as presented in Figure 7.1. On the top level, we have a structure *curve* which includes a pointer to another data structure indicating the first element of the next level. On the next level, we have a structure *interface* which consists of two next level structure *point*s (recall a *point* is an (x, y) location in the computational domain), its beginning and its end, and a pointer to the next *interface*. Then a doubly linked list is used for the *curve* to maintain the overall information on the tracked front. Note that the *interface*s are linked for each *curve* individually. Since we need to keep track of which

tracked interfaces must be deleted in Step 4, we would also maintain a flag for each *curve* that tells whether it is an old *curve* or a new *curve*. In addition, it would be very useful in Step 1 to include a flag for each *curve* that indicates the physical type of curve, *e.g.*, shock, interface, or boundary *curve*.

We now discuss each step of Algorithm 7.1 in more detail.

Step 1: We begin our algorithm by solving a one-dimensional Riemann problem in a direction normal to each tracked interface using the values from the adjacent cells as data and obtain a set of waves traveling with speeds $\lambda_{1l}, \lambda_{2l}, \cdots, \lambda_{ml}$ and jumps $r_{1l}, r_{2l}, \cdots, r_{ml}$. Here the first and second subscripts on the speeds and jumps stand for the wave family and the index of the interface respectively. We expect the solution to this Riemann problem to consist of only one strong wave, corresponding to the shock or interface being tracked, and $m-1$ weaker waves. The strong wave is used to help choose the new interface location.

To be more precise, we discuss one simple approach in more detail. (Other ways to advancing fronts may be found in [3],[16],[17].) Let (x_l^*, y_l^*), $* = 1, 2$, be *point*s of the *interface* l. Assume that the strong wave is in the pth wave family, and so λ_{pl} is the speed of the strong wave on the *interface* l. Then the new location $(\widehat{x_l^*}, \widehat{y_l^*})$ of the *point* (x_l^*, y_l^*), under the current time step k, can be calculated by simply using the formula

$$\begin{pmatrix} \widehat{x_l^*} \\ \widehat{y_l^*} \end{pmatrix} = \begin{pmatrix} x_l^* \\ y_l^* \end{pmatrix} + \lambda_{pl}\, k \begin{pmatrix} \alpha_l \\ \beta_l \end{pmatrix} \tag{7.1}$$

where $(\alpha_l, \beta_l)^T$ is the normal direction to the *interface* l, $* = 1, 2$. Performing the same calculation as in (7.1) on all the *interfaces* in a given *curve*, we obtain an ordered set of points $\{((\widehat{x_l^*}, \widehat{y_l^*}), * = 1, 2), l = 1, 2, \cdots, n\}$ where n is the number of *interface*s in a *curve*. Here we assume that the original set of points, $\{((x_l^*, y_l^*), * = 1, 2), l = 1, 2, \cdots, n\}$, is an ordered set.

Note that in many problems, *e.g.*, when there is strong shear layer flow along the discontinuities, the tracked interfaces should be advanced not only in the normal direction to the interface as illustrated in Figure 7.2a, but also in the tangential direction. This can be done quite easily by moving points tangential to the interface using, for example, an average tangential velocity from the data of the normal Riemann problem. It is easy to show that doing so gives exact front propagation for linear advection equations and for simultaneously diagonalizable linear hyperbolic systems. For general linear hyperbolic systems, or for nonlinear equations, this front moving procedure gives a good approximation to the front motion.

Connecting each pair of points $(\widehat{x_l^1}, \widehat{y_l^1})$ and $(\widehat{x_l^2}, \widehat{y_l^2})$ with a straight line using (7.1) or the modified locations which take account of the tangential effect of the flow, for $l = 1, 2, \cdots, n$, we then obtain the new location of the tracked interfaces at the next time step. Notice that in general these new interfaces would not join together into a continuous curve as seen in Figure 7.2a.

To form a continuous curve, one simple approach is to take an average of two neighboring points $(\widehat{x_l^2}, \widehat{y_l^2})$ and $(\widehat{x_m^1}, \widehat{y_m^1})$, where m is the *interface* next to the *interface* l, for $l = 2, 3, \cdots, n-1$, and collect the set of averaged points together with appropriate endpoints to form a new ordered set. Let $(\bar{x}_l, \bar{y}_l)$ be the averaged point location. This results in the following set:

$$\{(\widehat{x_1^1}, \widehat{y_1^1}), (\bar{x}_1, \bar{y}_1), (\bar{x}_2, \bar{y}_2), \cdots, (\bar{x}_{n-1}, \bar{y}_{n-1}), (\widehat{x_n^2}, \widehat{y_n^2})\}. \tag{7.2}$$

a) b)

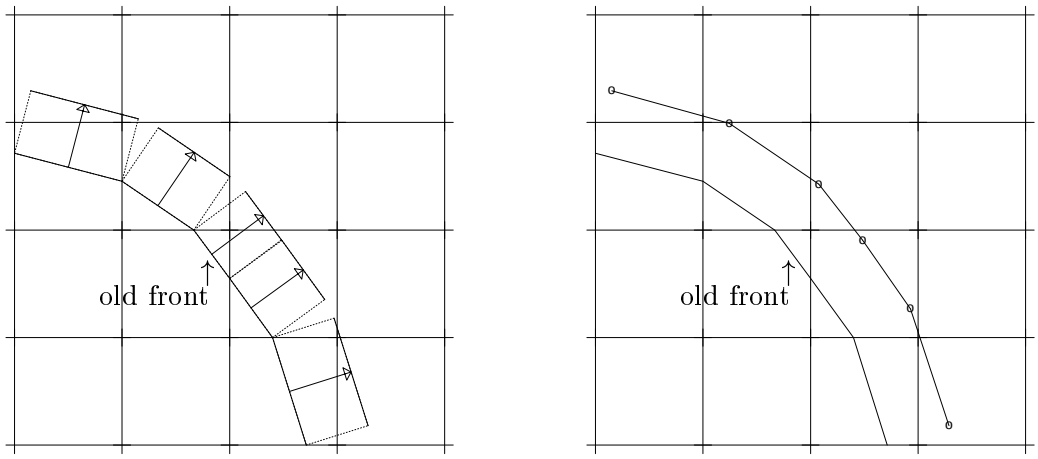


Figure 7.2: Front propagation. a) Tracked interfaces after propagating the original interfaces using the strong wave speeds obtained from the normal Riemann problems under the current time step. b) New tracked interfaces after taking an average of two neighboring points on the new interface location shown in a), and connecting the resulting points (indicated by large dots) consecutively by straight lines.

For simplicity, we write the set (7.2) as

$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \cdots, (x_n, y_n), (x_{n+1}, y_{n+1})\}. \tag{7.3}$$

Finally, a continuous piecewise linear curve, the new location of the tracked interface, is obtained by connecting the points in set (7.3) consecutively by straight lines as shown in Figure 7.2b.

Mathematically, this piecewise linear curve is represented by a parametric form

$$\mathcal{P}(s) = (\mathcal{X}(s), \mathcal{Y}(s)) \tag{7.4}$$

where $x = \mathcal{X}(s)$ and $y = \mathcal{Y}(s)$ are piecewise linear polynomials, and s is the parameter along the curve. Assume that s is in $[0, 1]$. To assign the parametric value s to each point (x_k, y_k), we use the simplest approach by choosing a uniform mesh size, $\triangle s = 1/n$, and setting $s_k = (k-1) \triangle s$. Then from points (x_k, y_k) in (7.3) and the parametric variables s_k, the piecewise linear polynomials take the form

$$\begin{array}{rl} \mathcal{X}_k(s) = & a_k + b_k s \\ \mathcal{Y}_k(s) = & c_k + d_k s \end{array} \tag{7.5}$$

where

$$\begin{aligned} a_k &= \frac{x_k s_{k+1} - x_{k+1} s_k}{s_{k+1} - s_k} \\ b_k &= \frac{x_{k+1} - x_k}{s_{k+1} - s_k} \end{aligned}$$

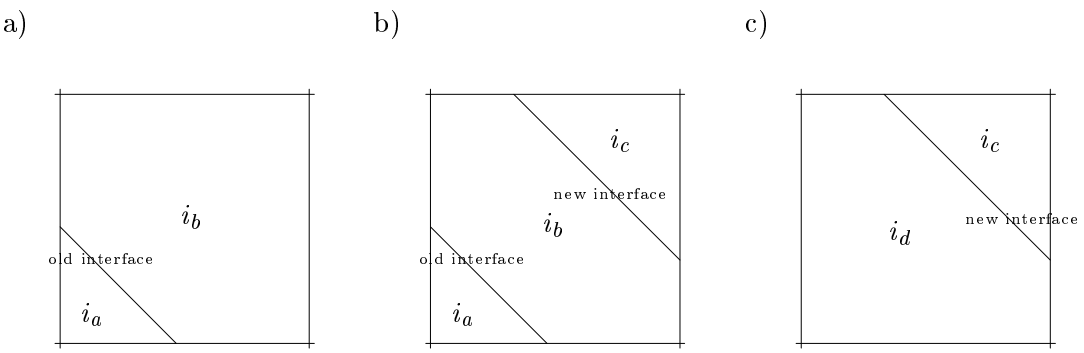


Figure 7.3: A new tracked interface is created in the front propagation procedure, in Step 2, that leads to a subdivision of cells. a) Grid at time step n before inserting the new tracked interface to the grid. b) One-grid approach. In time step n, we split C_{i_b} in two, setting $U^n_{i_c} = U^n_{i_b}$. In time step $n+1$, we eliminate the old tracked interface, and merge the C_{i_a} and C_{i_b} to C_{i_d} using (7.7). c) Two-grids approach. In time step n, we split the original regular cell i in two, setting $U^n_{i_d}$ using (7.7) and $U^n_{i_c} = U^n_{i_b}$. Our grid system consists of both the old grid in a) and the new grid in c).

$$\begin{aligned} c_k &= \frac{y_k s_{k+1} - y_{k+1} s_k}{s_{k+1} - s_k} \\ d_k &= \frac{y_{k+1} - y_k}{s_{k+1} - s_k} \end{aligned} \tag{7.6}$$

for $s_k \le s \le s_{k+1}$ and $k = 1, 2, \cdots, n$, and hence we get the parametric representation of the piecewise linear curve $\mathcal{P}(s) = (\mathcal{X}(s), \mathcal{Y}(s))$ with

$$\begin{aligned} x &= \mathcal{X}(s) = \{\mathcal{X}_k(s), k = 1, 2, \cdots, n\} \\ y &= \mathcal{Y}(s) = \{\mathcal{Y}_k(s), k = 1, 2, \cdots, n\}. \end{aligned}$$

More generally, based on the data (x_k, y_k, s_k) we could use some sort of curve fitting procedures or the reconstruction technique of the ENO (essentially nonoscillatory) method[48] to construct a smoother parametric curve $\mathcal{P}(s)$ to any desired order. The possibility of using this higher order representation of the tracked front, in particular constructed by the ENO method, will be discussed further in the next chapter.

Step 2: Having gotten the new location of the tracked front $\mathcal{P}(s)$ at the next time step, we then insert it into the underlying grid. This can be done quite easily by marching along the parametric curve $\mathcal{P}(s)$ from s_1 to s_{n+1} and looking for the intersections of each of the piecewise linear polynomials $(\mathcal{X}_k(s), \mathcal{Y}_k(s))$ with the underlying fixed grid. This determines *point*s for *interface*. Connecting the resulting *point*s by a straight line in an orderly way, we obtain the *curve* and also the new grid at the next time step.

Now since each new tracked interface subdivides some cell into two subcells, we must assign a cell value to each of these subcells. As mentioned earlier, there are two approaches to setting up the data structure for the subdivided cell, and hence there are two ways to assign the subcell values. If we adopt the first approach, *i.e.*, we treat the old and new tracked interfaces as being in the same grid cell, as in the one-dimensional front tracking

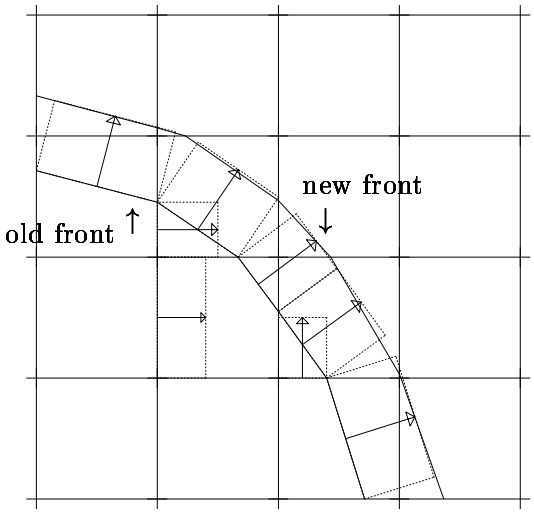


Figure 7.4: Wave propagation in Step 3 (only some of the waves are drawn). Each wave is propagated independently. For waves passing through each other the interaction is linearized. Note that the tracked waves are propagated close to the tracked interface introduced in Step 2.

algorithm, the simplest way to do this is to assign the previous cell value to each subcell, see Figure 7.3b. Whereas, if we adopt the second approach, treating the old and new tracked interfaces as being in two different grid cells, the old cell and the new cell, each subcell would be initialized by a value based on the appropriate weighted combination of the old cell values, see Figure 7.3c.

Step 3: Once the new grid is constructed, we then update the cell average U_j^n by applying the finite volume wave propagation methods described in Chapter 6, see Figure 7.4 for illustration. As in the one-dimensional algorithm, because of the carefully chosen grid, the tracked discontinuity is propagated close to the tracked interfaces. There is little or no smearing of the tracked wave during the averaging process. Smooth flow is captured as usual. Again, near tracked interfaces, waves may propagate through several cells due to the fact that we have created small subcells. In the next chapter, we perform error estimation to study stability properties of this wave propagation method using various finite volume approaches. The error behavior and accuracy near the tracked interface will also be examined.

It should be mentioned that, in principle, we can use any finite volume method to update the solution on this nonuniform grid created by the front tracking algorithm. The best way to do this that achieves higher order of accuracy, even for cells near the tracked interfaces, is still under study. We will show some preliminary results in the next chapter that give some indications of what one should do to accomplish this, however.

Step 4: We now delete the old tracked interfaces from the grid system. Again, we have to discuss two different situations. First, in the one-grid approach, to delete the old tracked interfaces would correspond to merging two subcells into one, and the cell value in the combined cell would be calculated by the appropriate weighted combination of these

two deleted subcells to maintain the correct cell average. For example, in Figure 7.3b the old tracked interface is deleted from the grid cell. Let C_{i_d} denote the cell after deletion. Then the cell average of the C_{i_d} cell becomes

$$U_{i_d}^{n+1} = \frac{A_{i_a}}{A_{i_d}} U_{i_a}^{n+1} + \frac{A_{i_b}}{A_{i_d}} U_{i_b}^{n+1} \tag{7.7}$$

where $U_{i_a}^{n+1}$, $U_{i_b}^{n+1}$ are the cell averages in the C_{i_a} and C_{i_b} cells respectively; A_{i_a} and A_{i_b} are the corresponding areas of the subcell, and A_{i_d} is the area of the C_{i_d} cell.

Alternatively with the two-grids approach, only grid cells which contain exclusively old tracked interfaces need to apply the above averaging procedure, because on grid cells which consist of both old and new tracked interfaces this procedure has already been used in Step 3 to assign new subcell values.

7.2 The Euler Equations and Boundary Conditions

Before presenting numerical results with this front tracking algorithm, we introduce the two-dimensional version of the Euler equations of gas dynamics and discuss the implementation of the solid wall boundary conditions for this system.

The inviscid Euler equations of gas dynamics in two dimensions have the form

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (\rho E + p)u \end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (\rho E + p)v \end{pmatrix} = 0, \tag{7.8}$$

where ρ, u, v, p, E are the density, velocity in the x-direction, velocity in the y-direction, pressure, and total energy of gas per unit mass, respectively. We again assume the equation of state satisfies the $\gamma-$law; so the internal energy is $e = \frac{1}{\gamma-1} p/\rho$ and the total energy of gas per unit mass is $E = e + \frac{1}{2}(u^2 + v^2)$.

As in one space dimension, the wave propagation approach is very easy to apply for various boundary conditions. Nonreflecting-outflow boundaries and periodic boundaries can be handled in a manner quite similar to the one-dimensional case. Here we devote our discussion solely to the most interesting case, the solid wall boundary.

At a solid wall boundary, the proper boundary condition for the Euler equations is zero normal velocity. Now consider the grid configuration shown in Figure 7.5a where a Cartesian grid is cut off by an irregular boundary. In the wave propagation approach, waves resulting from solving one-dimensional Riemann problems at the cell boundaries are used to update cell values. To achieve the solid wall boundary condition, waves which leave at the boundary are now reflected to the interior domain, as in the one-dimensional case, see Section 3.2.

For example, Figure 7.5a shows a wave originating from the Riemann problem between cells (i, j) and $(i + 1, j)$ that passes all the way through the irregular cell $(i + 1, j)$. The portion of this wave that lies beyond the boundary is then reflected normal to the boundary segment and back into the computational domain, as shown in Figure 7.5b. This reflected wave carries a reflected jump $\bar{r}_p$ and is used to update cell averages that overlap with the reflected wave, in this case cells $(i+1, j)$ and $(i+1, j-1)$. The relation between the reflected

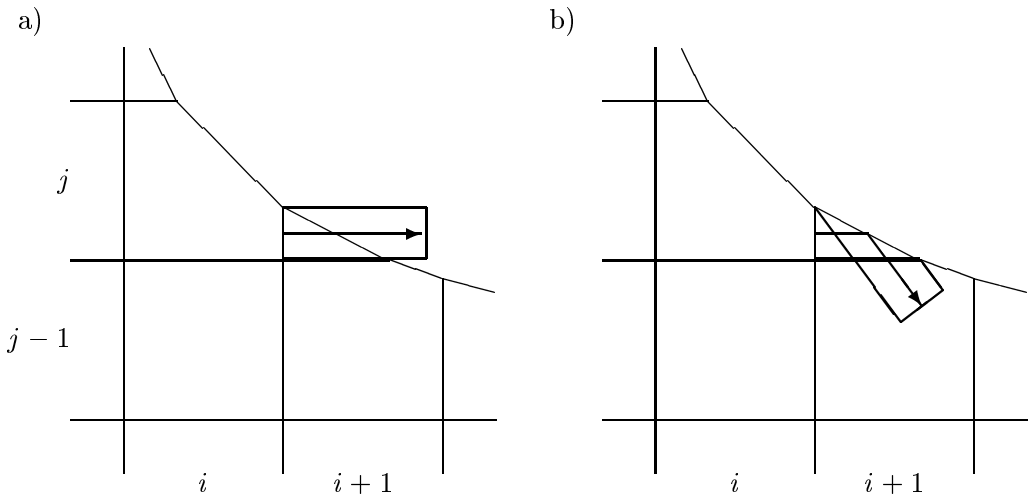


Figure 7.5: a) Wave propagating through a small boundary cell and out of computational domain. b) Reflected wave actually used.

jump $\bar{r}_p$ and the outgoing jump r_p that fulfills the solid wall boundary condition can be obtained by first rotating r_p to the ξ-η (normal-tangential) coordinates at the boundary, negating the normal velocity, and then rotating the resulting jumps back to the Cartesian coordinates[61],[62].

For convenience we use an operator Θ, called the rotation operator, to denote the coordinate transformation of the velocity field from the Cartesian coordinates to the ξ-η coordinates. For the Euler equations (7.8),

$$\Theta = \begin{pmatrix} 1 & & & \\ & \alpha & \beta & \\ & -\beta & \alpha & \\ & & & 1 \end{pmatrix}. \tag{7.9}$$

Recall that $\alpha = \cos\theta$, $\beta = \sin\theta$. It is also convenient to define the inverse of the rotation operator, Θ^{-1}, which maps the velocities in the ξ-η coordinates back to the original coordinates. With these notations, we may simply write

$$\bar{r}_p = -\mathcal{R}(r_p)$$

to express the above wave reflection procedure, where $\mathcal{R}$ is an operator of the form

$$\mathcal{R} = \Theta^{-1} \begin{pmatrix} 1 & & & \\ & -1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \Theta. \tag{7.10}$$

Similarly, the reflected slope $\bar{\sigma}_p$ is related to the outgoing slope σ_p by

$$\bar{\sigma}_p = -\mathcal{R}(\sigma_p).$$

a) b)

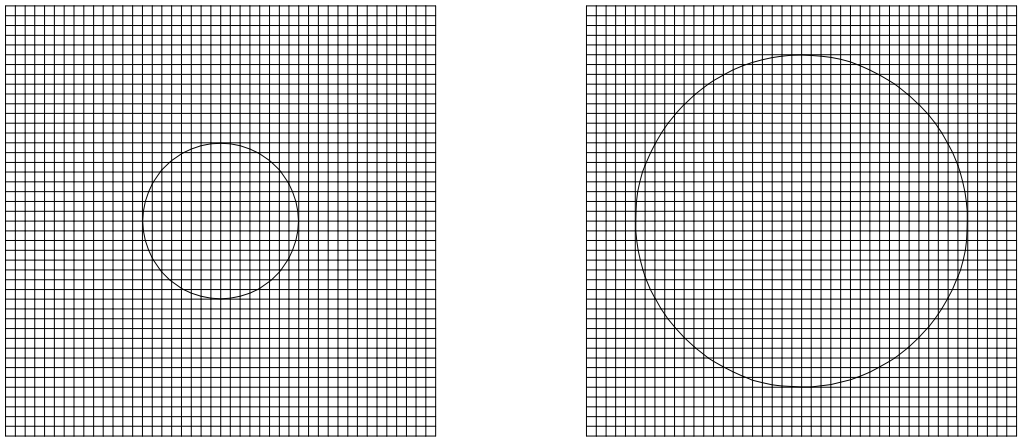


Figure 7.6: Grid used for the radially symmetric expanding shock wave. a) Grid for the initial data. b) Final grid after 14 time steps (time $t = 0.1$).

In addition to reflecting waves, we need to solve a one-dimensional Riemann problem normal to the boundary with data $u_r = \Theta(\tilde{U})$ and $u_l = \mathcal{R}(u_r)$, where $\tilde{U}$ can be obtained either from data in the adjacent cell or from some interpolation method[7]. With this initial data there is only one incoming wave that affects the cell values. Of the other three waves, the two contact discontinuities will have zero strength and zero velocity, while the outgoing image of the incoming wave is ignored.

7.3 Radially Symmetric Shock Waves

We now show results obtained using this front tracking algorithm. As a first example, we consider a radially symmetric expanding shock wave. Outside of a circle of radius $r_0 = 0.2$, we set

$$\rho = 1.4, \quad u = 0, \quad v = 0, \quad p = 1.$$

Inside the circle, the initial data is:

$$\begin{aligned} \rho(x, y, 0) &= 5.143204 \\ u(x, y, 0) &= 2.045108\ (x - x_0)r/r_0^2 \\ v(x, y, 0) &= 2.045108\ (y - y_0)r/r_0^2 \\ p(x, y, 0) &= 9.045462 \end{aligned}$$

where $r^2 = (x - x_0)^2 + (y - y_0)^2$ is the distance from the center $(x_0, y_0) = (0.5, 0.5)$. The initial grid is shown in Figure 7.6a where the initial shock is inserted as an interface that subdivides some cells in the underlying 40×40 grid.

After 14 time steps (time $t = 0.1$ and Courant number $\nu = 0.9$), we obtain the results shown in Figure 7.7 on the grid shown in Figure 7.6b. Notice that the tracked shock

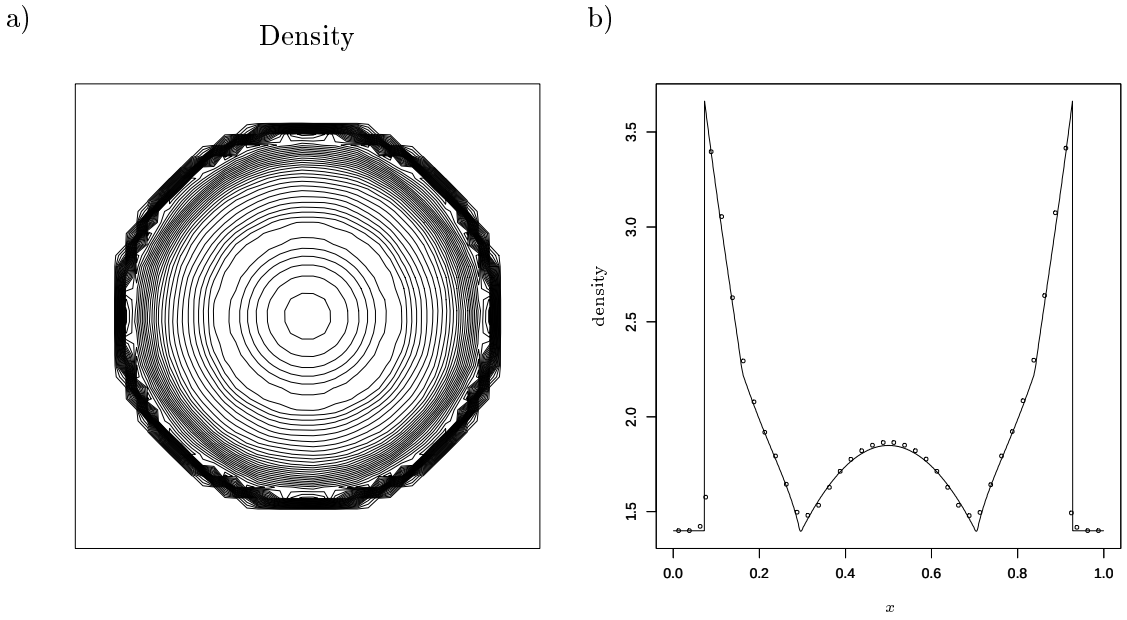


Figure 7.7: Results for the radially symmetric expanding shock wave. a) Density contours at time $t = 0.1$. b) Cross section of density along line $y = 0.5$. The solid line is the "true" solution obtained from solving the system $u_t + f(u)_r = \psi(u)$ with appropriate source terms for the radial symmetry using the one-dimensional front tracking algorithm. The dotted points are the two-dimensional result.

remains smooth and circular and appears to be very well located. The density contour plot in Figure 7.7a is not very sharp due to the graphics routine which plots the solution projected onto a uniform grid. The cross-section along $y = 0.5$ shown in Figure 7.7b shows the sharpness of our result (dotted points) much more clearly.

Note the solid line in this figure is the "true" solution as calculated with our one-dimensional front tracking algorithm on the system $u_t + f(u)_r = \psi(u)$ with appropriate source terms for the radial symmetry, using $h = 0.001$. The two-dimensional results shown above were obtained using the high resolution Godunov method on the regular cells and the Godunov method with tangential splitting on the irregular cells, with $\gamma = 1.4$ on a unit square domain ($[0, 1] \times [0, 1]$). No slope is introduced for the irregular cells.

Next, we consider a radially symmetric converging shock wave. The initial data now consists of two circular regions. Inside of a circle of radius $r_0 = 0.36$, we have density 1.4, zero velocity, and pressure $p = 1$. Outside the circle of radius r_0 and inside a circle of radius $r_1 = 0.46$, the initial data is:

$$\begin{aligned}
\rho(x, y, 0) &= 5.143204 \\
u(x, y, 0) &= -2.045108\ ((r - r_1)/(r_0 - r_1))^2 (x - x_0)/r \\
v(x, y, 0) &= -2.045108\ ((r - r_1)/(r_0 - r_1))^2 (y - y_0)/r \\
p(x, y, 0) &= 9.045462
\end{aligned}$$

The outer circle is introduced to maintain the radial symmetry of the flow. Results are shown in Figure 7.8. We again observe good agreement of the results. In this calculation,

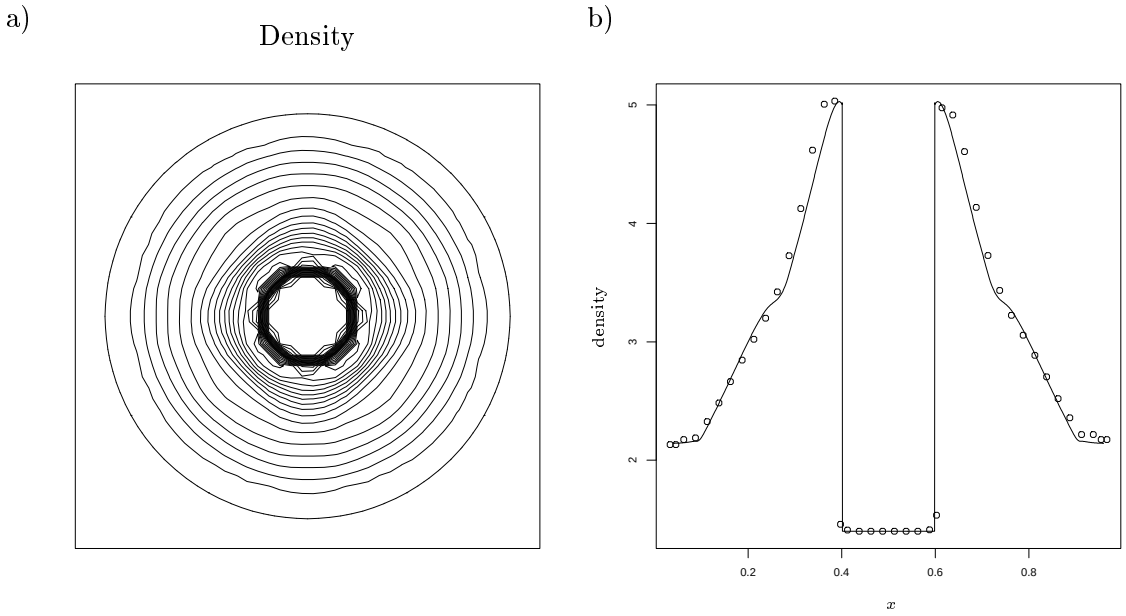


Figure 7.8: Results for the radially symmetric converging shock wave. a) Density contours at time $t = 0.1$. The outer circle shown in the figure is a solid wall boundary. b) Cross section of density along line $y = 0.5$.

the high resolution method used in the previous example was employed to update the cell values, and the wave reflection procedure, described in Section 7.2, was used to handle the solid wall boundary at the outer circular boundary.

Chapter 8

ERROR ANALYSIS

As in our one-dimensional front tracking algorithm, we use a high resolution method that is essentially second order accurate away from the tracked interfaces. We use front tracking in order to resolve discontinuities properly, and so our method does not suffer the standard loss of accuracy due to smearing that a shock capturing method would suffer. Nevertheless, there can be some loss of accuracy near the discontinuity due to the nonuniformity of the grid and the piecewise linear representation of the tracked discontinuity. A planar isolated discontinuity separating two constant states is tracked perfectly, but in a more realistic situation the discontinuity may have some curved structure and interact with some smooth background flow. There are several factors that can then lead to loss of accuracy near the tracked discontinuity, such as the piecewise linear representation of the discontinuity, loss of accuracy due to the use of the nonuniform and time-dependent grid, the choice of slopes in neighboring cells, and the linearization of the interaction between the tracked discontinuity and weak waves from the neighboring cell interfaces. Since it is difficult to analyze errors arising in some of these instances, here we will only examine the first two problems in some detail, and leave other problems as future work. To begin, we report results on the order of accuracy for some sample problems where exact or "true" solutions are available.

8.1 Preliminaries

As in the one-dimensional case, see Section 4.1, we use E_j^n to denote the *global error* of the grid cell j at time t_n. We assume that the global error in some particular norm, $\| E^n \|$, can be expressed in relation to $O(h^p)$, in which the largest real number p is called the *order of accuracy* of a method as h approaches zero for all $t_n \geq 0$. Here h is the mesh size in both the x- and y-directions of the underlying uniform grid cells. The norms we use here are: the 1-norm,

$$\| E^n \|_1 = \sum_j A_j |U_j^n - u_j^n|,$$

where A_j is the area of the jth grid cell, and the max-norm,

$$\| E^n \|_{\max} = \max_j |U_j^n - u_j^n|.$$

In addition to computing errors over the entire grid cells using the above norms, we also compute errors for cells near the tracked discontinuity. This is done using the following 1-norm,

$$\| E_{\chi^\pm}^n \|_1 = \frac{1}{h} \sum_{k \in \chi^\pm} A_k |U_k^n - u_k^n|,$$

where now the sum is over the set of irregular cells in either the state behind the discontinuity χ^-, or the state ahead the discontinuity χ^+. The order of accuracy of a method is computed in a manner similar to what is described in Section 4.1.

We now consider some sample problems and investigate the order of accuracy that is achieved by using our front tracking algorithm.

Example 8.1. We first consider a scalar linear problem consisting of the linear advection equation

$$u_t + au_x + bu_y = 0 \qquad \text{for} \quad 0 \le x \le 1,\ 0 \le y \le 1 \tag{8.1}$$

with initial data

$$u(x,y,0) = \begin{cases} 2 + 1.5e^{20(\xi - 0.32)} & \xi < 0.32 \\ 1 + 0.5\tanh(6\pi(0.36 - \xi)) & \text{otherwise} \end{cases} \tag{8.2}$$

where $a = \cos 5^o$, $b = \sin 5^o$, and $\xi = ax + by$. This initial data gives an oblique discontinuity at an angle $\theta = 5^o$ to the y-coordinate with an extreme point just behind the discontinuity. The exact solution for this problem can be obtained by simply shifting this initial profile in the flow direction $(\cos 5^o, \sin 5^o)$ with speed 1. Note that if we view this problem in the direction of (a, b), this is essentially a one-dimensional problem; the same one as we have studied previously in Example 4.1, but now the problem is solved on a two-dimensional grid with an oblique profile.

As in the one-dimensional error estimation performed in Section 4.1, we examine the error behavior of the method as time evolves and as the mesh is refined. For this problem, we perform error estimation up to time $t = 0.1$ at 5 different times (at every integer multiple of the time interval $k = 0.02$) with a mesh refinement sequence $\{h_l = 2^{1-l}/25, k_l = h_l/2, l = 1, 2, 3\}$. The result is shown in Figure 8.1 where the errors and order of accuracy in the 1-norm and max-norm are presented for the Godunov method, the rotated Godunov method, and the high resolution Godunov method. From the figure, we observe the poor order of accuracy of the methods we employed here, particularly, in the max-norm. This is also the case for the one-dimensional test as seen in Figure 4.2, and has been discussed in some detail previously, see Chapter 4. Notice that there is little distinction between the results obtained by using the Godunov method and the rotated Godunov method. For convenience in reading, we again plot the errors in the logarithmic scale with base 10. (This is also the case for other figures shown below relating to errors of a method.)

It should be mentioned that because the work increases a great deal by introducing slopes for the irregular cells, for simplicity, in the experiments performed here we do not incorporate the slope information for the high resolution method on these cells. We use the "MUSCL" limiter (2.12) to determine slopes of the regular cells.

Example 8.2. Next, we consider a radially symmetric problem arising from the Euler equations (7.8). As initial conditions, we take the data from the example of an expanding shock wave discussed in Section 7.3. For this problem, we compute the "true" solution by first applying the one-dimensional front tracking algorithm to the system $u_t + f(u)_r = \psi(u)$ with the appropriate source terms for the radial symmetry, using $h = 0.001$, and then interpolating this one-dimensional result on a two-dimensional grid.

Table 8.1 shows results in density of an accuracy study up to time $t = 0.1$ using the Godunov method, the rotated Godunov method, the Godunov method with tangential

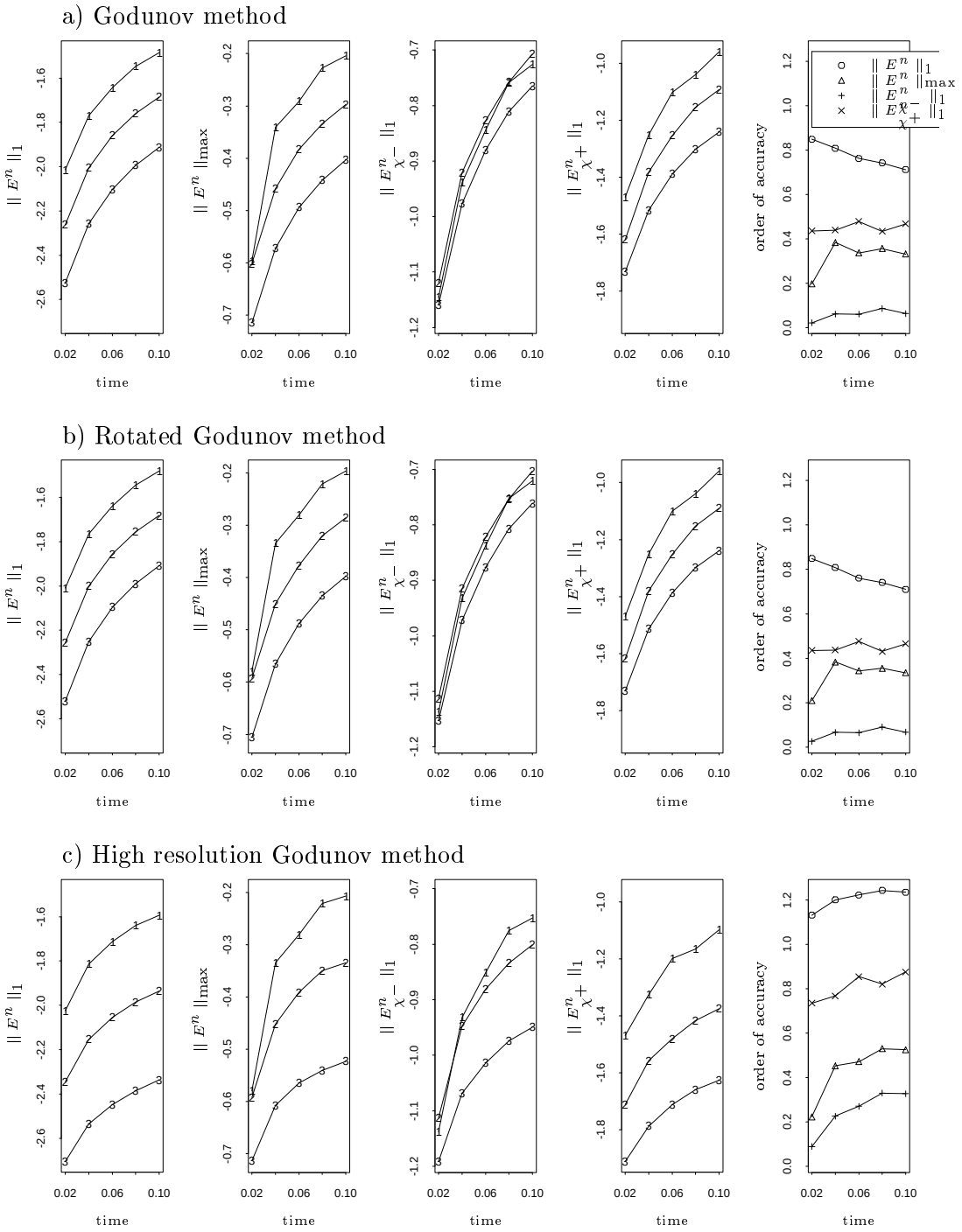


Figure 8.1: An accuracy study of the front tracking algorithm for the linear advection equation (8.1) with initial data (8.2) up to time $t = 0.1$. All the errors shown in the figure are plotted in the logarithmic scale with base 10. Error estimation is performed at 5 different times with a mesh refinement sequence $\{h_l = 2^{1-l}/25, l = 1, 2, 3\}$.

splitting, and the high resolution Godunov method. From it, we see that in the 1-norm the Godunov method and the rotated Godunov method are first order accurate, the Godunov method with tangential splitting is order $p = 0.92$, and the high resolution Godunov method is order $p = 1.65$. In the max-norm, the order of accuracy for the methods used here is not as good as we might hope to obtain. This is expected, however, because unlike the previous linear problem where the grid is exact, here the error on the grid due to the discrepancy between the tracked interfaces and the exact shock location contributes a source of error. Note in the table the number in the parenthesis represents the exponent of the base 10, for example, $9.9149(-2)$ stands for 9.9149×10^{-2}.

In the same table, we also show the errors and accuracy for the irregular cells and the tracked shock position. We observe reasonable results using our front tracking algorithm. For example, for the high resolution method we have order $p = 1.43$ in the post-shock density, order $p = 0.84$ in the pre-shock density, and order $p = 1.72$ in the shock position. Note that the error for the tracked front location at time t_n is defined by

$$E^n_{\text{front}} = r^n_{\text{true}} - \bar{r}^n_{\text{computed}},$$

where r^n_{true} is the "true" shock location in the radial direction from the center $(x_0, y_0) = (0.5, 0.5)$, and $\bar{r}^n_{\text{computed}}$ is the averaged shock location obtained by averaging the radial distances of the *point*s on the tracked shock.

To examine more closely the error behavior of the irregular cells and the tracked shock, in Figures 8.2, 8.3, and 8.4, we plot the data that is used to compute the errors and accuracy shown in the table. It is clearly seen that our solutions converge to the "true" solution as the mesh is refined, and the solution varies from angle to angle in an oscillatory way. Notice that there are big spikes appearing in some of the figures, particularly, in the pre-shock state in Figure 8.3. One of the reasons for the occurrence of the spikes is due to the fact that we use the large time step approach on a grid which contains the *approximate* location of the tracked discontinuity, see Figure 7.4. Because of this, it is unavoidable to have some numerical diffusion of the solution. Even though the amount of diffusion is small, when the grid cell is tiny the contribution of this to the cell value will be significant. This causes a big spike of the error. Despite this fact, our numerical result is still convergent with a reasonable rate. No stability problem has been observed for this test.

For comparison, we have also done experiments using the standard shock capturing methods. The results are shown in Table 8.2. It is clear that our front tracking result is superior to that obtained from shock capturing. For this problem, Courant number $\nu_0 = 0.5$ was used in the experiments.

8.2 Tracked Front Representation and Accuracy

In the above accuracy study, we used the piecewise linear parametric curve $\mathcal{P}(s)$ in (7.5) to approximate the tracked front position after propagating the front in Step 1 of the front tracking algorithm 7.1. We find the intersections of this curve $\mathcal{P}(s)$ with the underlying grid, and use piecewise linear segments that connect the resulting intersecting points to make the tracked front and also the grid at the next time step, Step 2 of the algorithm 7.1. For planar discontinuities, this is a good approximation, but more generally for curved discontinuities this piecewise linear approximation of the front is less desirable. In this section, we consider

Table 8.1: An accuracy study in density of the front tracking algorithm for a radially symmetric expanding shock wave.

a) Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$	$\| E^n_{\chi^-} \|_1$	$\| E^n_{\chi^+} \|_1$	$\lvert E^n_{\text{front}} \rvert$
0.04	9.9149(-2)	6.3076(-1)	8.8315(-1)	1.1592(-1)	1.1127(-2)
0.02	4.5955(-2)	4.2825(-1)	4.3967(-1)	6.1691(-2)	6.3594(-3)
0.01	2.3739(-2)	4.5908(-1)	2.1463(-1)	4.3657(-2)	3.5389(-3)
order p	1.03	0.23	1.02	0.70	0.83

b) Rotated Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$	$\| E^n_{\chi^-} \|_1$	$\| E^n_{\chi^+} \|_1$	$\lvert E^n_{\text{front}} \rvert$
0.04	1.1368(-1)	5.9339(-1)	9.1022(-1)	1.4420(-1)	9.9987(-3)
0.02	5.1797(-2)	1.3328(0)	5.0736(-1)	8.8274(-2)	6.3172(-3)
0.01	2.6670(-2)	4.1488(-1)	2.2343(-1)	4.5132(-2)	3.8698(-3)
order p	1.05	0.26	1.01	0.84	0.68

c) Godunov method with tangential splitting

h	$\| E^n \|_1$	$\| E^n \|_{\max}$	$\| E^n_{\chi^-} \|_1$	$\| E^n_{\chi^+} \|_1$	$\lvert E^n_{\text{front}} \rvert$
0.04	1.0565(-1)	5.3912(-1)	8.2659(-1)	1.2996(-1)	1.3031(-2)
0.02	5.5341(-2)	3.1151(-1)	4.9793(-1)	7.3319(-2)	7.9232(-3)
0.01	2.9540(-2)	9.6968(-1)	2.1376(-1)	3.8686(-2)	4.6351(-3)
order p	0.92	–	0.98	0.87	0.75

d) High resolution Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$	$\| E^n_{\chi^-} \|_1$	$\| E^n_{\chi^+} \|_1$	$\lvert E^n_{\text{front}} \rvert$
0.04	9.0823(-2)	4.5326(-1)	7.3904(-1)	1.2855(-1)	1.0123(-2)
0.02	2.7692(-2)	4.5839(-1)	2.4489(-1)	7.8437(-2)	3.6884(-3)
0.01	9.2866(-3)	1.4665(-1)	1.0243(-1)	3.9976(-2)	9.3533(-4)
order p	1.65	0.81	1.43	0.84	1.72

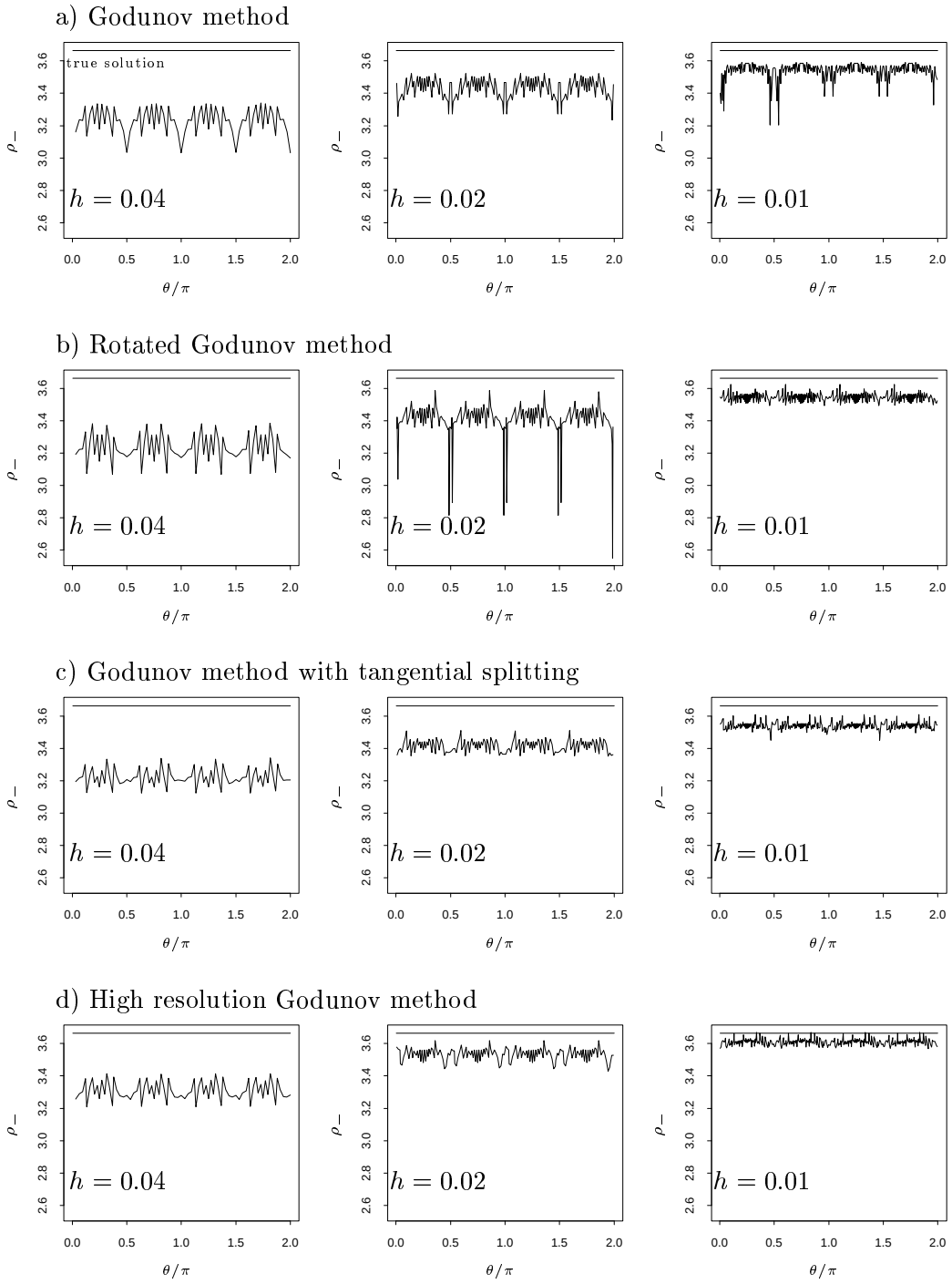


Figure 8.2: A comparison of the density in the post-shock irregular cells, ρ_-, for a radially symmetric expanding shock wave. The straight line shown in the figure is the "true" solution.

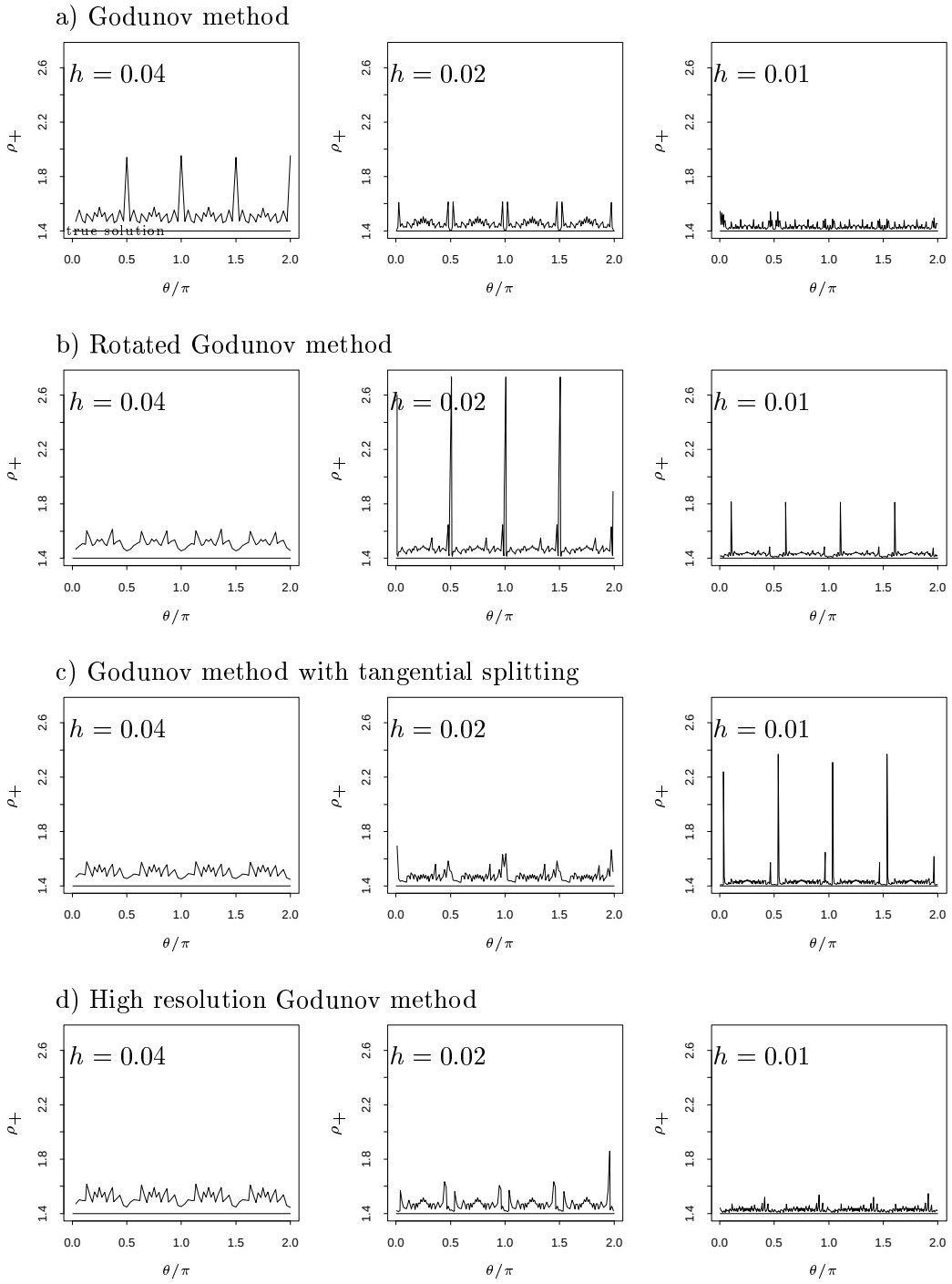


Figure 8.3: A comparison of the density in the pre-shock irregular cells, ρ_+, for a radially symmetric expanding shock wave. The straight line shown in the figure is the "true" solution.

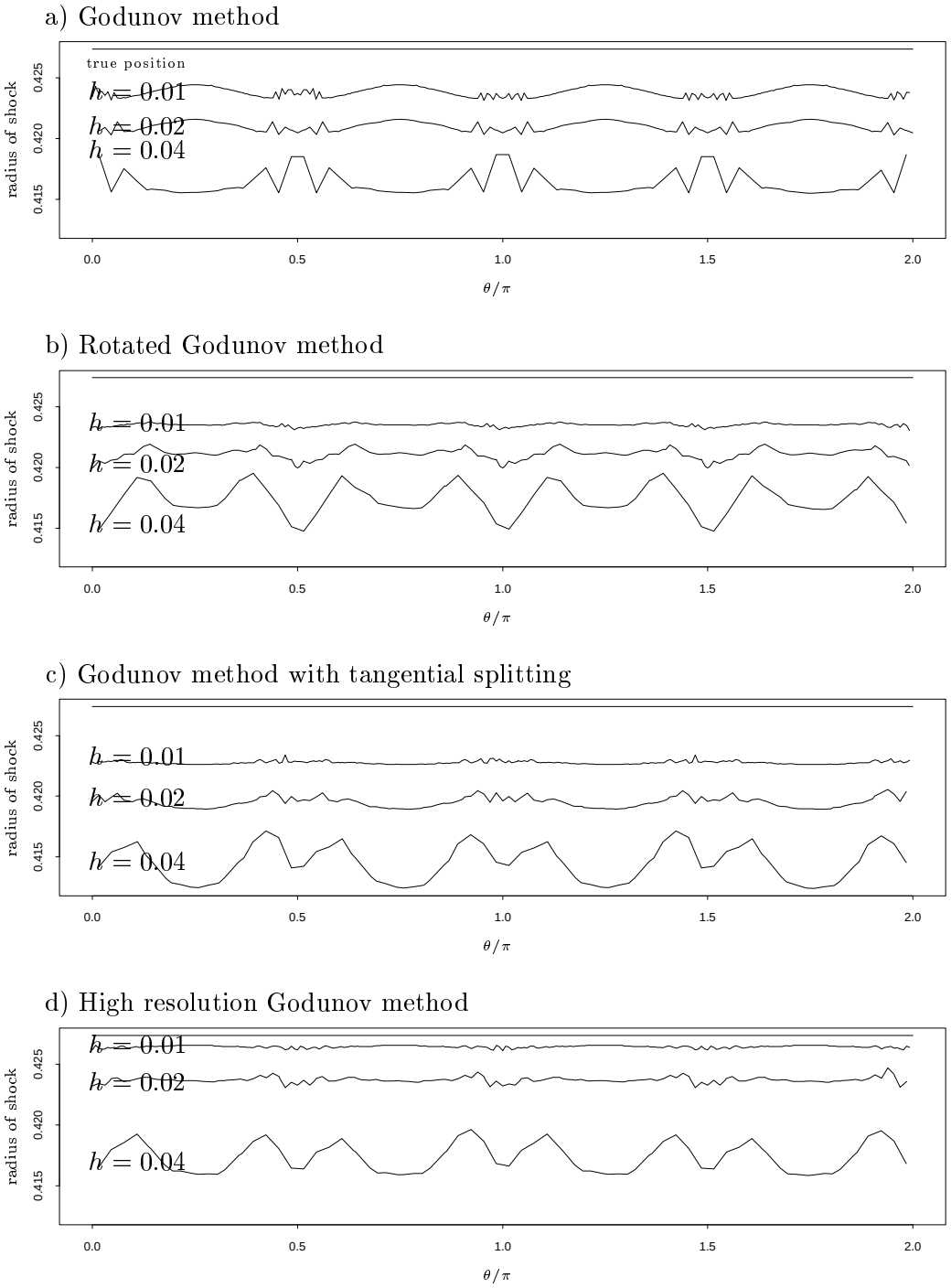


Figure 8.4: A comparison of the radius of the tracked shock for a radially symmetric expanding shock wave. The straight line shown in the figure is the "true" solution.

Table 8.2: An accuracy study in density of the shock capturing method for a radially symmetric expanding shock wave.

a) Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$
0.04	2.1519(-1)	1.3109(0)
0.02	1.2980(-1)	1.2179(0)
0.01	7.0001(-2)	1.3261(0)
order p	0.81	–

b) Rotated Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$
0.04	2.6508(-1)	1.2737(0)
0.02	1.5829(-1)	1.1269(0)
0.01	9.1466(-2)	1.1853(0)
order p	0.77	–

c) Godunov with tangential splitting

h	$\| E^n \|_1$	$\| E^n \|_{\max}$
0.04	2.2844(-1)	1.3172(0)
0.02	1.4063(-1)	1.1829(0)
0.01	7.6326(-2)	1.2346(0)
order p	0.79	–

d) High resolution Godunov method

h	$\| E^n \|_1$	$\| E^n \|_{\max}$
0.04	1.5172(-1)	1.2577(0)
0.02	8.3530(-2)	1.2973(0)
0.01	4.1517(-2)	1.3778(0)
order p	0.93	–

one simple modification of the algorithm that uses a piecewise *quadratic* parametric curve $\mathcal{P}(s)$ for the front position. Doing so should give us a more accurate front position, and so a more accurate grid when this curve is inserted into the grid. For simplicity, we still use piecewise linear segments as the grid interfaces. We will compare results obtained using these two different parametric representation of the fronts.

Recall that in Step 1 of the front tracking algorithm, we move each tracked interface in directions normal and tangential to the interface. We apply an interpolation scheme that determines a set of points (x_k, y_k, s_k) for the new tracked front position. In Section 7.1, these points were used to construct the piecewise linear parametric curve $\mathcal{P}(s)$. Here, instead, we use this set of data to make a piecewise quadratic parametric curve. In particular, we employ the reconstruction technique of the ENO method[48] to do this in which we make a divided difference table from the data and use the adaptive stencils which only select the smallest values from the divided difference table to form $\mathcal{P}(s)$ in Newton form.

Let $x[s_j, s_{j+1}, \cdots, s_{j+k}]$ be the divided difference of order k. Then by the above ENO construction, we get the piecewise quadratic polynomial

$$\mathcal{X}_k(s) = x_k + a_k(s - s_k) + b_k(s - s_i)(s - s_j) \tag{8.3}$$

where

$$\begin{aligned} a_k &= x[s_k, s_{k+1}] \\ b_k &= \phi(\theta_k) x[s_{k-1}, s_k, s_{k+1}] \end{aligned}$$

with

$$\theta_k = x[s_k, s_{k+1}, s_{k+2}] / x[s_{k-1}, s_k, s_{k+1}],$$

and ϕ is some limiter function, say the "minmod" limiter (2.10). The stencils i and j in (8.3) are chosen to interpolate points which have the smallest values of the second order divided difference from the neighboring points. Note that the divided difference $x[s_j, s_{j+1}, \cdots, s_{j+k+1}]$ of order $(k+1)$ is related to the divided differences $x[s_j, s_{j+1}, \cdots, s_{j+k}]$ and $x[s_{j+1}, s_{j+2}, \cdots, s_{j+k+1}]$ of order k by the equation

$$x[s_j, s_{j+1}, \cdots, s_{j+k+1}] = \frac{x[s_{j+1}, s_{j+2}, \cdots, s_{j+k+1}] - x[s_j, s_{j+1}, \cdots, s_{j+k}]}{s_{j+k+1} - s_j},$$

see Powell[85]. Similarly, $\mathcal{Y}_k(s)$ can be constructed in the same manner. Hence we get the piecewise quadratic curve $\mathcal{P}(s)$.

We now consider two examples and examine the front accuracy of our front tracking algorithm. As a first example, we consider evolving a circular front

$$(x - \frac{1}{4})^2 + (y - \frac{1}{4})^2 = (\frac{1}{5})^2$$

in the constant velocity field $(u, v) = (1, 1)$ on a unit square domain. For this problem, in each time step, we get the exact front position after Step 1 of the front tracking algorithm. There are some errors introduced in the front position, however, after Step 2 of the algorithm where we insert the new front position into the underlying grid. This makes the grid that is used in Step 3 to update the cell values.

Results of an accuracy study in the front position up to time $t = 0.4$ is shown in Figure 8.5 where the piecewise linear and piecewise quadratic curves $\mathcal{P}(s)$ are used in the test. It is interesting to see that the results are indistinguishable from these two different representations of the tracked front; they all converge roughly at the same rate with the same error magnitude. Notice that in each case the error grows as time evolves which yields the reduction of the order of accuracy. This is expected, however, because in each time step the tracked front is inserted into the grid, and that tends to clip the front, see Figure 8.6 where the grids constructed in the front tracking algorithm are shown. Here the time step is chosen by $k = h_l/2$ where $h_l = 2^{1-l}/25$, $l = 1, 2, 3$. Note that for this problem, it is possible to improve the front accuracy by taking a larger time step since doing so reduces the number of time steps in the experiments and hence the errors due to inserting the tracked front into the grid.

Our next example of examining the front accuracy concerns evolving an elliptical front

$$(x - \frac{1}{2})^2 + \frac{3}{2}\, y^2 = (\frac{1}{4})^2$$

in a rotating velocity field, $(u, v) = (-y, x)$ on a square domain, $[-1, 1] \times [-1, 1]$. This front rotates counterclockwise about the origin.

Figure 8.7 shows the evolution of the tracked fronts up to time $t = 5.375$ using both the piecewise linear and piecewise quadratic representation of the fronts. We now observe some errors of the tracked front; the size of the interface shrinks, and the shape becomes circular. This result is expected because, as in the previous circular front problem, the insertion of the front into the grid leads to some loss of accuracy. In addition, there are errors introduced in the front-moving procedure, in which the speed of the tracked front is obtained *via* some interpolation method. Here we use one simple approach that takes the

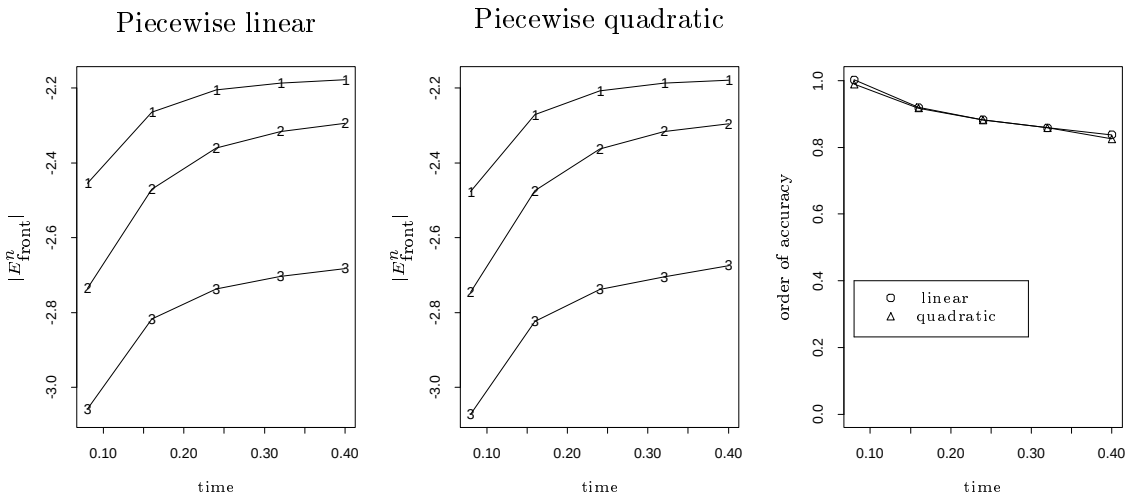


Figure 8.5: An accuracy study in the front position of the front tracking algorithm using the piecewise linear and piecewise quadratic curves $\mathcal{P}(s)$. Results for advancing a circular front in the constant velocity field $(u, v) = (1, 1)$ are shown.

average speed on the two sides of the tracked front as the front propagation speed. (It is possible to obtain a more accurate result if we employ a more sophisticated interpolation method for the front speed.) Note that there is little distinction between the results for these two different representations of the tracked front.

Based on the results shown in this section, we conclude that the use of a higher order representation of the front in Step 1 of the algorithm is not enough to improve the accuracy of the tracked front location. Work is in progress to find an efficient way to do this. It should be mentioned that even with the simplest piecewise linear approach, we still get reasonable front structure for many complicated problems, see Chapters 9 and 10.

8.3 Nonuniform Grids and Accuracy

As we have seen from the previous examples, the grid used in our front tracking algorithm is nonuniform and varies with time. Since in this instance it is difficult to do theoretical analysis of the tracking algorithm, here we perform error estimation and demonstrate a potential problem of loss of accuracy near the tracked interfaces due to the use of nonuniform and time-dependent grids. Our aim is to identify one possible source of error arising from our tracking algorithm, and hopefully pave the way for future algorithm development.

As an example, we consider the linear advection equation

$$u_t + u_x + u_y = 0 \qquad \text{for} \quad 0 \le x \le 1,\ 0 \le y \le 1 \tag{8.4}$$

with smooth initial data

$$u(x, y, 0) = 1 + 0.5 \sin(2\pi x) \sin(2\pi y) \tag{8.5}$$

and periodic boundary conditions. We ran this problem on a time-dependent grid where a circular interface is inserted as an interface in the underlying uniform grid, and advanced in the flow direction $(1, 1)$ with speed $\sqrt{2}$, see Figures 8.6 and 8.8.

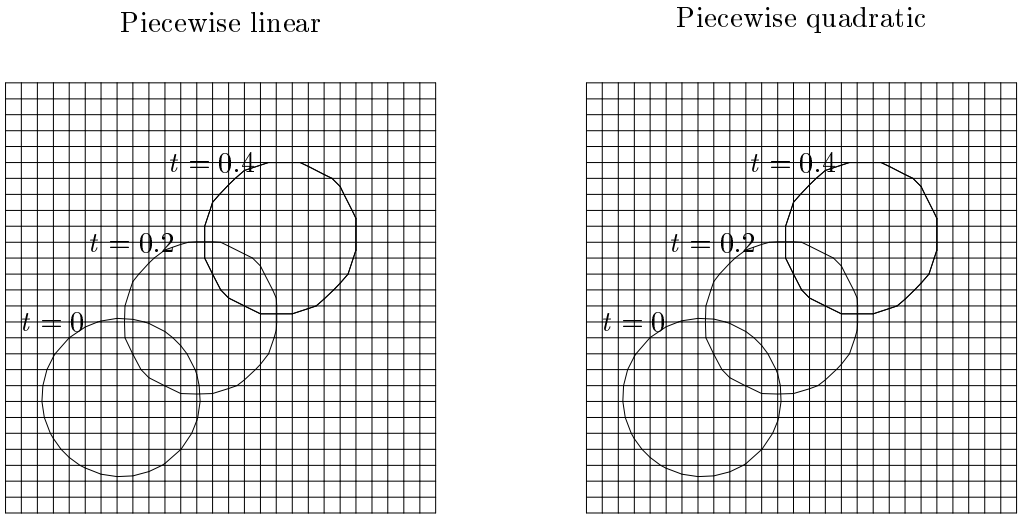


Figure 8.6: Grids constructed in the front tracking algorithm for the evolution of a circular front in the constant velocity field $(u, v) = (1, 1)$ using the piecewise linear and piecewise quadratic curves $\mathcal{P}(s)$. The underlying uniform grid is 25×25.

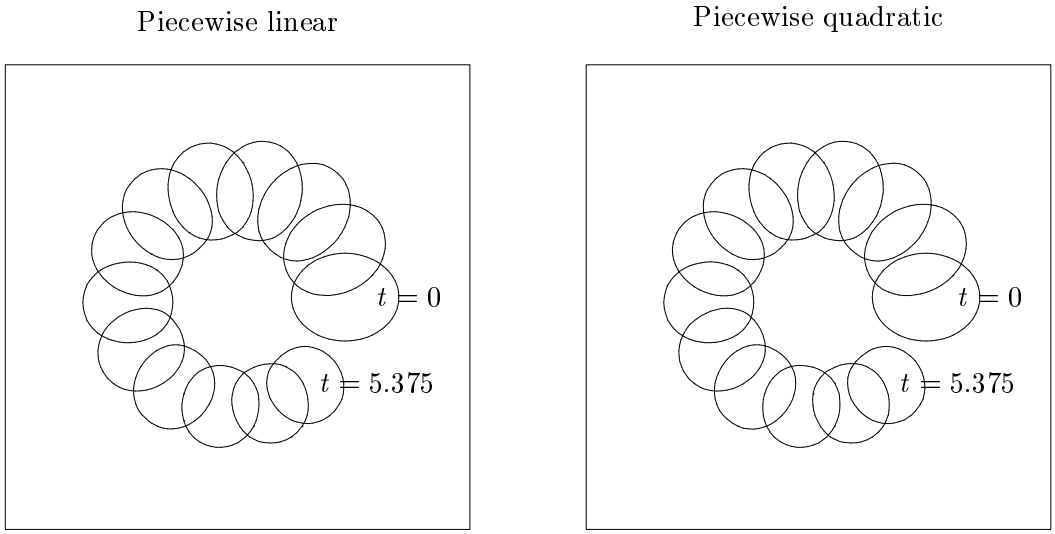


Figure 8.7: Evolution of an elliptical front in a rotating velocity field $(u, v) = (-y, x)$ using the front tracking algorithm with the piecewise linear and piecewise quadratic curves $\mathcal{P}(s)$. The underlying uniform grid is 80×80.

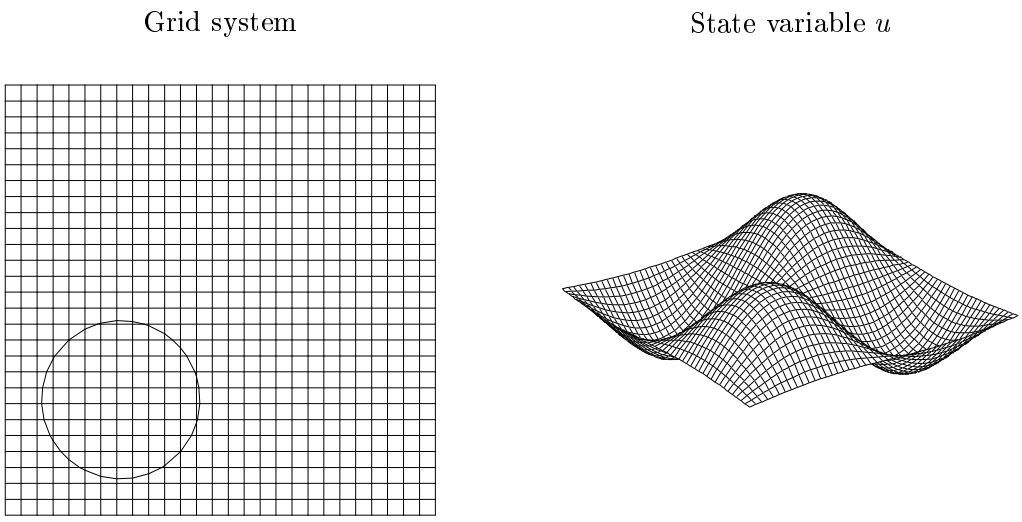


Figure 8.8: Initial conditions for an accuracy study of the linear advection equation (8.4) on nonuniform grids.

Results of an accuracy study in the state variable u up to time $t = 0.4$ are shown in Figure 8.9 where various finite volume methods are tested. From the figure, we observe some loss of accuracy in the max-norm, and for cells near the tracked interfaces. Because of this, we see some reduction of the order of accuracy in the 1-norm. It is not significant, however. Figure 8.10 shows the true solution and the snap shot of the solutions for the accuracy study shown in Figure 8.9 at time $t = 0.4$, using $h = 0.02$. Big errors near the tracked interfaces are clearly seen using the Godunov and rotated Godunov methods. This is expected, however, because for this problem the tracked interface is moving with exactly the same speed as the flow, and so the solution on each side of the interface is independent of the data on the other side, except for small errors due to the grid constructed in the front tracking algorithm. This situation is similar to what we have seen in the one-dimensional case, see Section 4.3, Figure 4.14.

Notice that the error for the high resolution method is somewhat smaller than for the Godunov methods, even though we used piecewise constant functions for the irregular cells, and piecewise linear functions only for the uniform cells. We would expect to obtain better results if slopes are introduced also in the irregular cells.

For comparison, we also ran this problem using the shock capturing methods on uniform grids and fixed nonuniform grids (the grid shown in Figure 8.8). Figure 8.11 shows results for the uniform grids, in which we observe first order accuracy in the 1-norm and max-norm for the Godunov and rotated Godunov methods, and second order accuracy for the Lax-Wendroff method.

Figure 8.12 shows results on a nonuniform grids where the location of the "front" is fixed rather than moving at the advection velocity, so that the smooth solution moves through the grid irregularity rather than moving with it. There is still some loss of accuracy relative to the uniform grid but not as bad as what was seen with the moving irregularity, as would

be expected.

These results show that nonuniformities in the grid can cause a loss of accuracy in the smooth structure of the solution near the interface. The results seen here look particularly bad for two reasons. First, since there is only smooth flow and no discontinuities the errors in the smooth flow are quite obvious and much worse than what is obtained on a uniform grid. For a problem with discontinuities across the interface (which is always the case in practice) the uniform grid method introduces huge errors near the discontinuity which are not incurred with the front-tracking method, so that the balance shifts. Second, the problems illustrated here are for the linear advection equation which is much less forgiving of errors than a nonlinear problem with shocks.

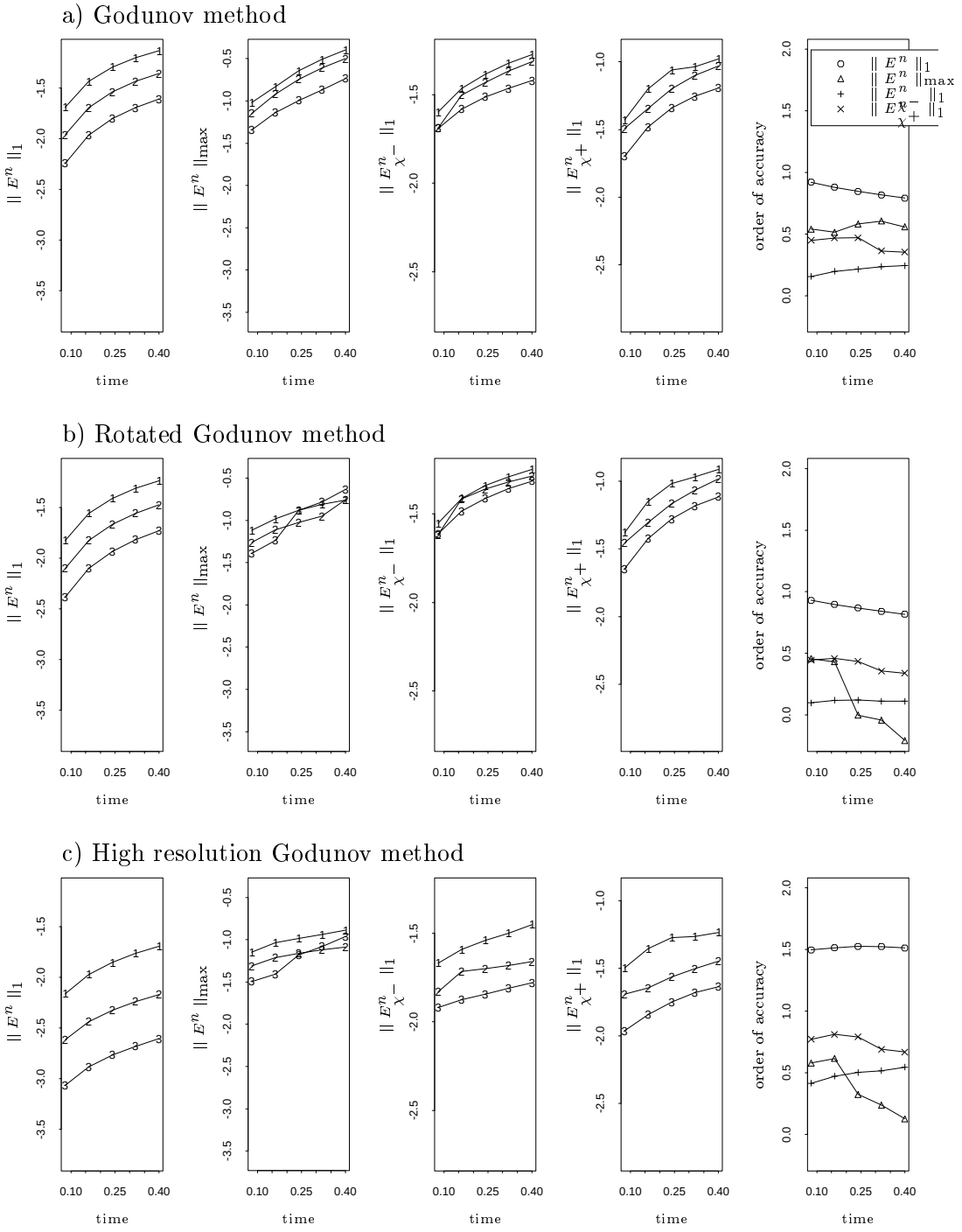


Figure 8.9: An accuracy study of the front tracking algorithm for the linear advection equation (8.4) with smooth initial data (8.5) on *time-dependent* nonuniform grids.

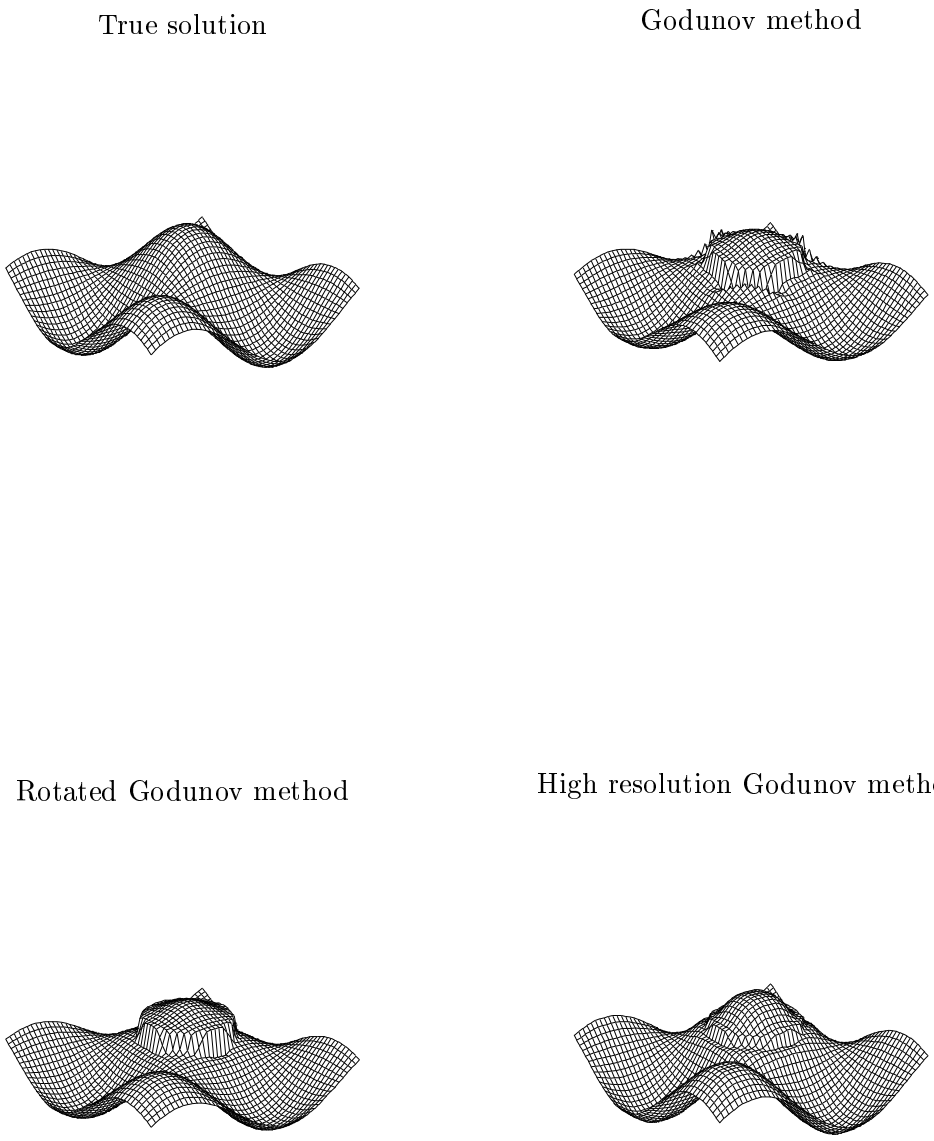


Figure 8.10: Plot of the true solution and the snap shot of the solution for the accuracy study shown in Figure 8.9 at time $t = 0.4$, using $h = 0.02$.

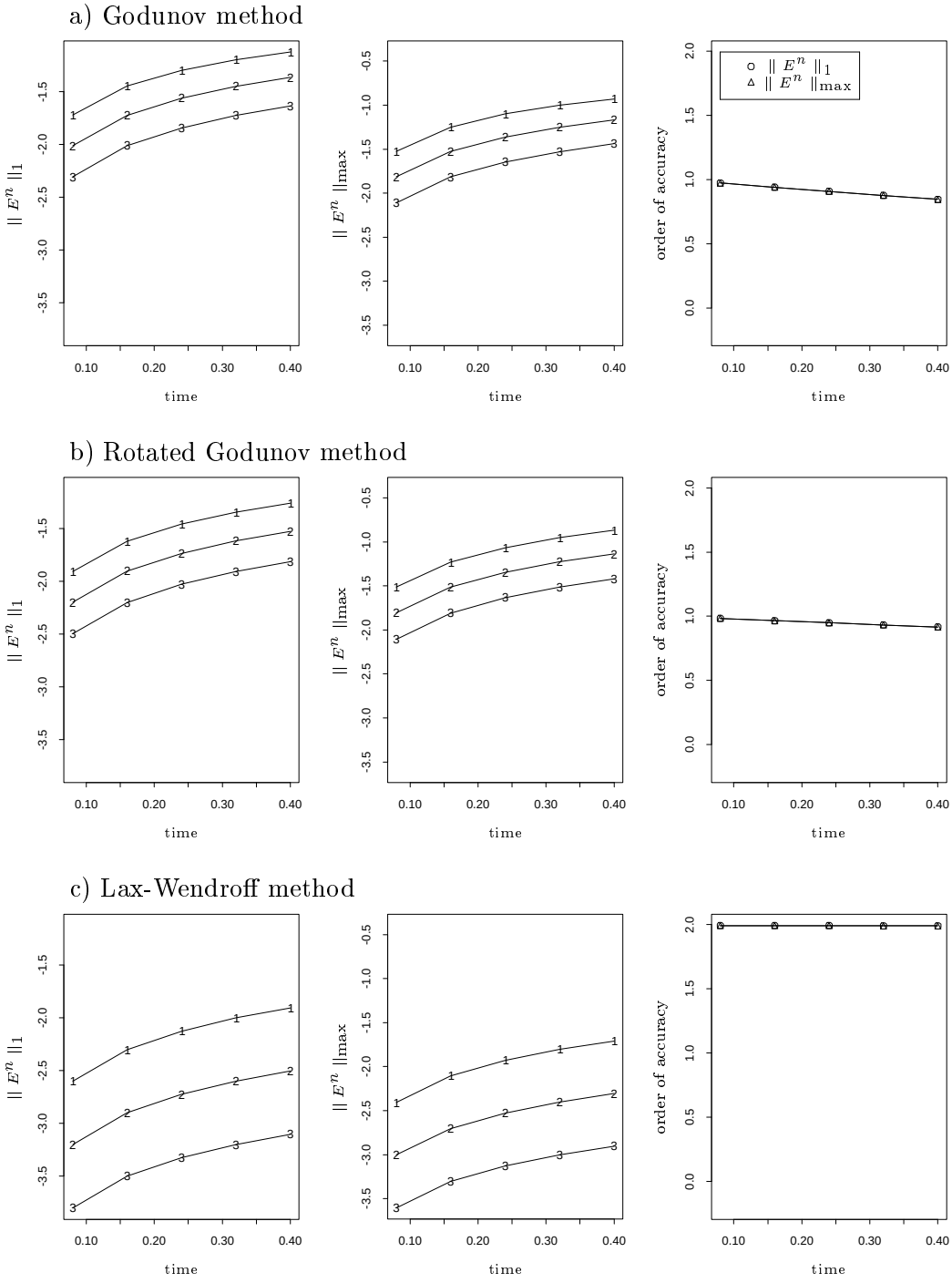


Figure 8.11: An accuracy study of the shock capturing method for the linear advection equation (8.4) with initial data (8.5) on uniform grids.

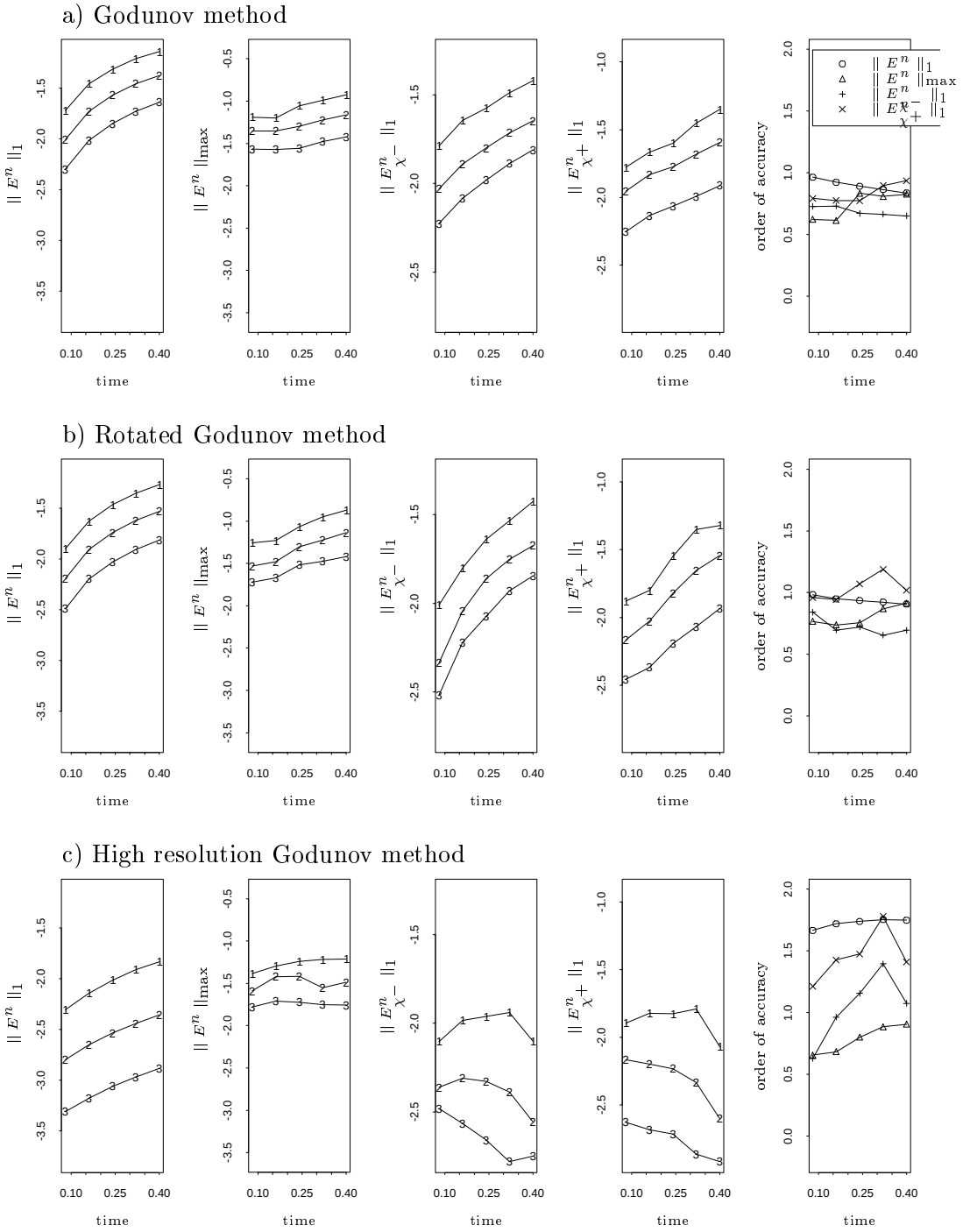


Figure 8.12: An accuracy study of the shock capturing method for the linear advection equation (8.4) with initial data (8.5) on *fixed* nonuniform grids.

Chapter 9

APPLICATIONS

Having analyzed the front tracking algorithm and finite volume approaches on the nonuniform grid, we now present more numerical results for some sample problems involving shocks and interfaces arising in gas dynamics. As in the one-dimensional tests performed in Chapter 5, our aims here are to validate our results by comparing them to results (either exact, numerical or experimental) which have already appeared in the literature. We also hope to demonstrate the potential power of using our front tracking algorithm on more complex problems.

The problems we consider are a shock-vortex interaction and a shock-ramp interaction for shocks, and the Kelvin-Helmholtz and Rayleigh-Taylor instabilities for interfaces. In addition, we show one preliminary result for a steady state problem. The finite volume method we use in these calculations (except for the steady steady problem) is the high resolution Godunov method on the regular cells, and the Godunov method with tangential splitting on the irregular cells; no slope is introduced for the irregular cells. This method gives the overall best performance in the error estimations performed in the previous chapter.

9.1 Shock Diffraction

In many applications, shock waves undergo complicated physical processes and display rich shock diffraction phenomena. Some interesting diffraction structures have been observed and documented in some instances from both laboratory experiments and numerical simulations, *e.g.*, for shock-bubble interaction[45],[83] and shock-ramp interaction[35],[36],[103]. Here we consider two typical examples: the shock-vortex interaction and the shock-ramp interaction, and demonstrate the usefulness of using front tracking for investigating the shock diffraction structure.

9.1.1 Shock-vortex interaction problem

As a first example to examine shock diffraction using our front tracking algorithm, we consider a shock wave interacting with a vortex pair. As noted in [30], this problem and the related subject have been an area of active research for many years. Most of the work was motivated by an interest in the noise produced by rockets and high-speed aircrafts, and therefore the research emphasized the generation of acoustic waves. For these problems, the interaction of shocks with turbulent flows is a significant source of noise[115]. This shock-vortex system is an important element of these more complex processes, see [28],[30], and references therein for more detail.

As initial conditions, we take a planar rightward moving Mach 1.5 shock at $x = 0.175$ with density $\rho = 1.4$, zero velocity, and pressure $p = 1$ in the pre-shock state, and in addition we put a pair of counter-rotating isothermal composite vortices in the pre-shock

state. Since this shock is approaching the vortices, interaction occurs subsequently, see Figure 9.2.

The composite vortex we use is a vortex with velocity field

$$v_\theta = \begin{cases} v_0 r/r_1 & 0 < r \le r_1 \\ Ar + B/r & r_1 < r \le r_2 \end{cases} \tag{9.1}$$

where v_0 is a constant which characterizes the strength and rotation direction of the vortex, $r^2 = (x - x_0)^2 + (y - y_0)^2$ is the distance from the vortex center (x_0, y_0), and A and B are constants so that the velocities are continuous at $r = r_1$ and $r = r_2$. Inside the vortex, the pressure field is specified so that the pressure gradient balances the centripetal force

$$\frac{dp}{dr} = \rho \frac{v_\theta^2}{r}. \tag{9.2}$$

Notice that since the vortex is assumed to be isothermal, the density ρ inside the vortex only differs from the pressure p by a constant, *i.e.*, $\rho = p/T_0$ where T_0 is the constant temperature in the pre-shock state; assuming the universal gas constant $R = 1$. So the above pressure equation (9.2) can be integrated, and hence p can be obtained explicitly.

The parameters we use for the vortex pair are given by:

upper vortex: $(x_0, y_0) = (0.4, 0.7)$, $v_0 = 0.6944$, $A = -2.3148$, $B = 0.09259$

lower vortex: $(x_0, y_0) = (0.4, 0.3)$, $v_0 = -0.3472$, $A = 1.1574$, $B = -0.04629$

and $r_1 = 0.1$ and $r_2 = 0.2$ in each case. Following the naming used in [30], a vortex is called a "strong" vortex if the maximum velocity v_0 in the vortex core is exactly equal to the flow velocity v_f behind the shock, and is called a "weak" vortex if $v_0 \ll v_f$. Here the above parameters are chosen so that the upper vortex is a "strong" vortex while the lower vortex is a "weak" vortex. The upper vortex is rotating in a counter-clockwise manner, whereas the lower vortex is rotating clockwise.

Figures 9.1 and 9.2 show results for this problem after 100 time steps (time $t = 0.387$). In Figure 9.1a, we show the tracked shocks, plotted every 4 time steps. From it, we observe that the shock structure is not significantly diffracted by its interaction with a weak vortex, while it is affected by its interaction with a strong vortex.

To make use of the tracked shock information to diagnosis the shock diffraction as time evolves, one popular approach is to produce a history of the amplitude of the front perturbation, particularly producing the so-called min-max front history[11]. In the present case, at each time step, we monitor the minimum and maximum horizontal distance (x_{min} and x_{max}) from the shock to the left boundary $x = 0$. The result is shown in Figure 9.1b where we have run the problem for the interaction of a shock with a single vortex so as to distinguish the difference in the front perturbation with different vortex strengths. Note the perturbation of the shock grows weaker, as the shock moves farther away from the vortex.

Figure 9.2 shows the density contour plot at six different times, plotted every 20 time steps. Now we can see more differences in the wave structure as the shock passes through the strong and weak vortices. Although there are acoustic waves generated in front and in back of the shock in each of the vortices, the waves appearing near the strong vortex apparently are more compressive than the ones appearing near the weak vortex, and eventually form a shock wave, inducing complex wave patterns. Notice that the shape of the vortex is severely distorted during the stage of wave interaction; it tends to form an ellipse afterward.

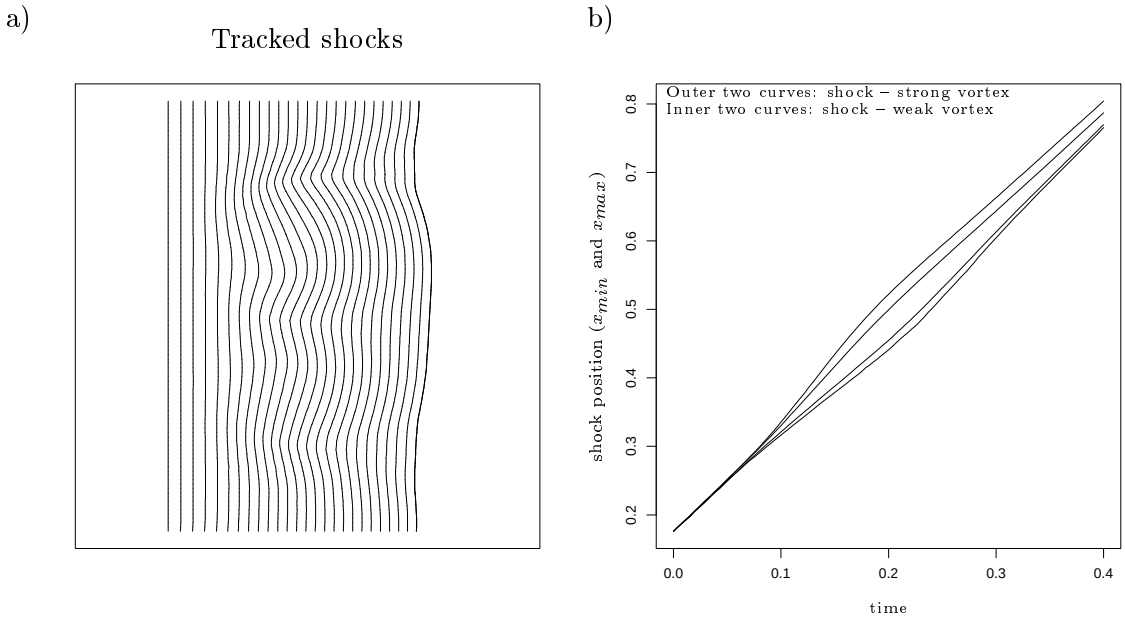


Figure 9.1: Results in the shock structure for the shock-vortex interaction problem. a) Tracked shocks, plotted every 4 time steps. b) A comparison of the history of shock positions x_{min} and x_{max} from the results of the shock-strong vortex interaction and the shock-weak vortex interaction.

Qualitatively, our results agree well with the experimental and numerical results shown in [28],[30]. It can also be demonstrated that some vorticity is generated in this case. A similar situation in the interaction between a shock and bubble is reported in several references, see, *e.g.*, [83].

In this example, a 100×100 grid was used on a unit square domain, non-reflecting outflow boundary conditions were used on the left and right boundaries, and solid wall boundary conditions were used on the top and bottom boundaries; Courant number $\nu_0 = 0.9$ was employed here.

9.1.2 Shock-ramp interaction problem

Our next example on shock diffraction concerns an oblique shock reflection in which an incident shock wave interacts with a solid wall ramp. This problem has been extensively studied over the years because it simulates various typical and also important shock diffraction patterns, *e.g.*, regular reflections, single Mach reflections, complex Mach reflections, and double Mach reflections, depending on the Mach number of the incident shock and the ramp angle, see [35],[36] for both numerical and experimental results. Here we consider one such example involving a double Mach reflection.

The structure of a double Mach reflection consists of the incident shock, the first Mach stem, the first regular shock reflection, the second Mach stem, and the second regular shock reflection. The first three waves form a triple point, and so do the last three waves. In addition, at each triple point, there is a slip line separating flow between the Mach stem and reflected shock[36].

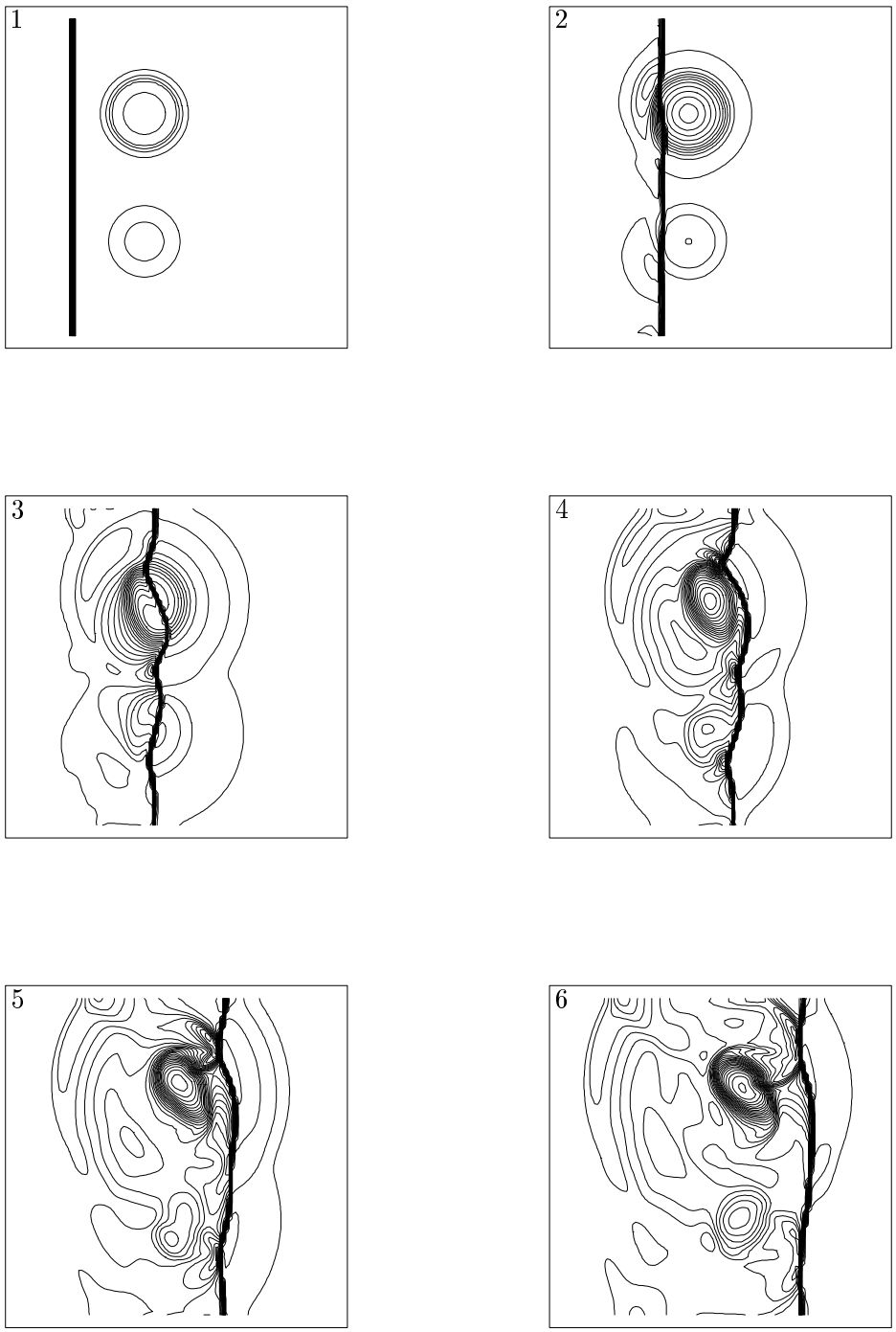


Figure 9.2: Density contours for the shock-vortex interaction problem up to time $t = 0.387$, plotted every 20 time steps.

To start the computation, an oblique Mach 4.62 shock with an angle normal to a 40^o ramp is initialized at the ramp corner ($x = 0.7$) with density $\rho = 1.4$, zero velocity, and pressure $p = 1$ on the left to the shock. (This shock is moving leftward.) For convenience, the ramp is arranged so that it is aligned with the grid in front of the shock and cuts through the underlying Cartesian grid in back of the shock as seen in Figure 9.3a.

Figure 9.3a shows the evolution of the tracked shock, plotted every 10 time steps. It is easy to observe that due to the shock-ramp interaction a kink which corresponds to the location of a triple point is formed. Since, for the moment, we are not able to handle the triple point explicitly, we only track the incident shock (above the kink) and the Mach stem (below the kink), and leave the regular shock reflection to be captured.

Figure 9.3b shows the density contour plot for the same run at time $t = 0.1$. From it, we can clearly see the wave structure around the first triple point as described above, and can also observe some structure in the downstream triple point. The density cross-section along the ramp is shown in Figure 9.3c where we compare our front tracking result (drawn in dotted points) with the shock capturing result (drawn in solid line) obtained by using the non-tracked version of the high resolution Godunov method with the same mesh size. We observe good agreement with these two results. Here we used a 160×80 grid on a rectangular region ($[0, 0.8] \times [0, 0.4]$). The solid wall boundary conditions described in Section 7.2 was applied for the ramp, and the non-reflecting outflow boundary conditions were applied for the other boundaries.

9.2 Interface Instability

Interfaces are commonly seen in the real world. In many applications, their behaviors under small perturbation are of great importance. Here we consider two standard problems associated with unstable interfaces: the Kelvin-Helmholtz and the Rayleigh-Taylor instabilities, and study the growth of interfaces using our front tracking algorithm.

9.2.1 Kelvin-Helmholtz instability

As a first example of tracking an interface for the Euler equations, we consider the Kelvin-Helmholtz instability in which there is an interface separating two fluids of different tangential velocities. This interface is unstable with respect to any sinusoidal perturbation, and often rolls up into large vortical structures which serve to entrap the fluid. Typical examples where this instability can occur are seen in many applications, *e.g.*, in jet flow and shear layer flow[103].

Here we consider one simple setup of the Kelvin-Helmholtz instability. We take constant density ρ_0 and pressure p_0 with zero vertical velocity in the computational domain. Above the interface, we have horizontal velocity $u = u_0$, and below the interface, we have horizontal velocity $u = -u_0$. For this model problem, there is one dimensionless parameter which controls the behavior of this Kelvin-Helmholtz unstable interface, namely, the Mach number u_0/c_0[84].

The initial perturbation of the interface is given by

$$y = y_0 + \varepsilon \sin(kx) \tag{9.3}$$

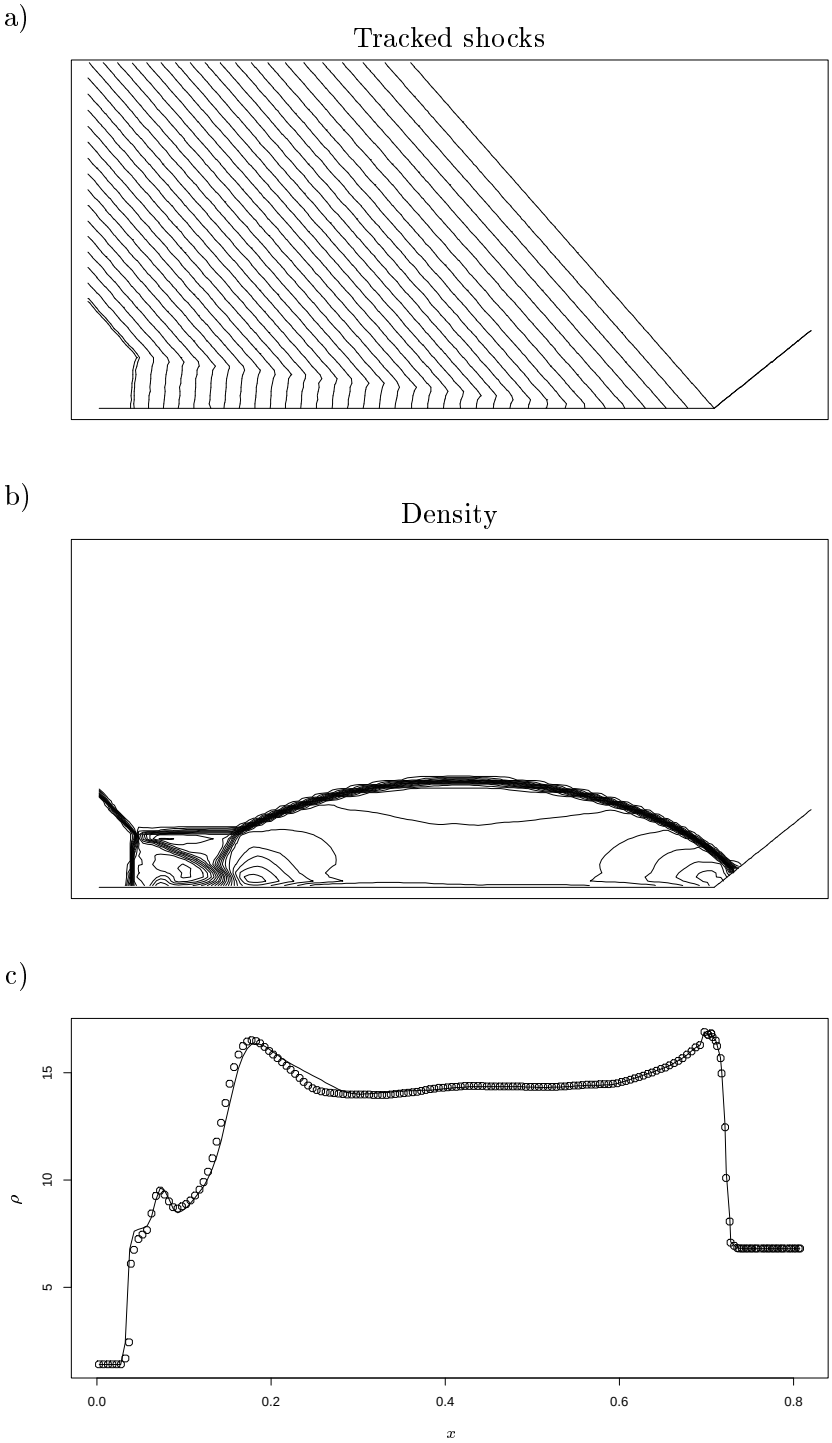


Figure 9.3: Results for a Mach 4.62 shock reflection off a 40^o ramp. a) Tracked shocks, plotted every 10 time steps. b) Density contours at time $t = 0.1$. c) Cross section of density along the ramp. The solid line is the shock capturing result, while the points are the shock tracking result.

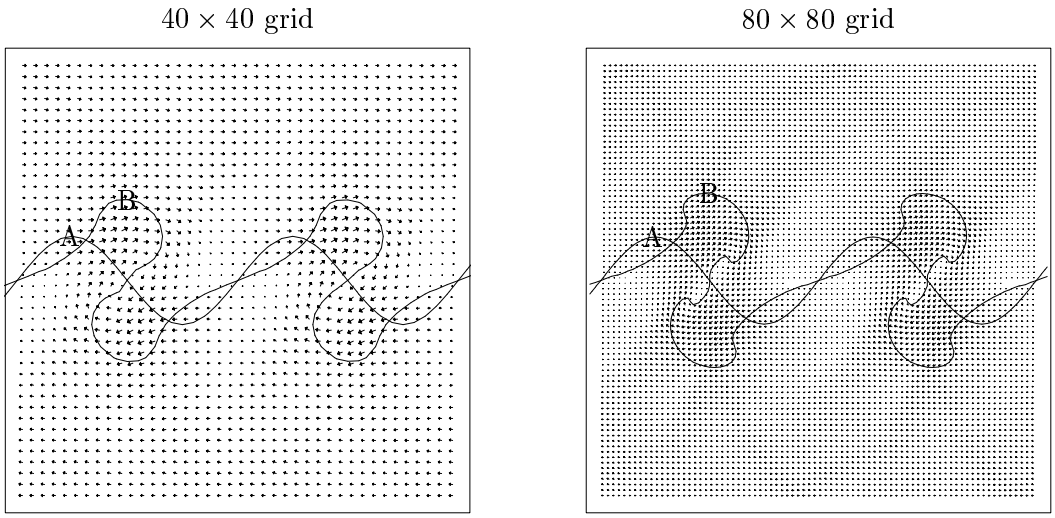


Figure 9.4: A convergence study of the interface for the Kelvin-Helmholtz instability. Results for two different mesh spacings, 40×40 and 80×80 grids, are shown at time $t = 0.48$. In each figure, curve A is the initial tracked interface at time $t = 0$, and curve B is the final tracked interface at time $t = 0.48$. Note the velocity field is superimposed on the figure.

where ε is the amplitude of the perturbation, and k is the wave number. By performing the standard linear stability analysis, we can derive the perturbed state and use it to initialize the flow, see Appendix A.1 for the analysis and solution. For comparison, in the run shown below, we choose the same parameters as used in Chern *et al.*[17], *i.e.*, we take $\rho_0 = 1.4$, $p_0 = 1$, $u_0 = 0.2$ (Mach 0.2), $y_0 = 0.5$, $\varepsilon = 0.1$, and $k = 4\pi$. The computational domain is a unit square with solid walls on the top and bottom and periodic boundaries on the left and right.

Figure 9.4 shows a convergence study of the interface using two different mesh spacings, 40×40 and 80×80 grids. From the figure, roll-up of the interface is clearly seen on the 80×80 grid. Note that the initial tracked interface is also shown in the figure and the velocity field is superimposed. Comparing our results with the one shown in [17], we see good agreement on the global structure, but not on the fine structure around the roll-up on the same grid. In fact, our result on the 80×80 grid is very similar to their result on the 40×40 grid. So this indicates that their front tracking method gives a better resolution near the tracked interface than the result obtained using our method.

9.2.2 Rayleigh-Taylor instability

Our next example of interface tracking concerns the Rayleigh-Taylor instability in which the interface separates two fluids of different densities. Assume that gravity is directed downwards. This interface is unstable under any perturbation if the light fluid lies below the heavy fluid. As mentioned in [93], typical examples where this instability can occur are in the collapse of a massive star, the formation of high luminosity jets in rotating gas clouds in an external gravitational potential, the laser implosion of deuterium-tritium fusion targets, and the electromagnetic implosion of a metal liner. An overview of this problem

can be found in Sharp[93] and Youngs[113]. For more background information on fluid interface stabilities, one may consult, for example, Chandrasekhar[14] and Shivamoggi[94].

Here we consider one simple model for the compressible flow in which the viscosity, surface tension, and heterogeneity can be ignored. Then the governing equations for the compressible Rayleigh-Taylor instability take the form

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ (\rho E+p)u\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ (\rho E+p)v\end{pmatrix}=\begin{pmatrix}0\\ 0\\ \rho g\\ \rho vg\end{pmatrix} \tag{9.4}$$

where g is the gravitational acceleration. Note that source terms on the right hand side of the equations are a result of gravity acting on a unit mass of fluid. Hence they are called the "gravitational" source terms.

For this model, it is known that there are many factors which may influence the behavior of the Rayleigh-Taylor unstable interfaces[93]. Among them the following dimensionless parameters are of great importance. The first parameter is the density ratio $D = \rho_h/\rho_l$ (or the Atwood number $A = (\rho_h - \rho_l)/(\rho_h + \rho_l)$), which governs the growth rate of small amplitude perturbation; ρ_h and ρ_l are the density of the heavy fluid and the light fluid just below and above the unperturbed interface respectively. The second parameter is the ratio of specific heats γ or other information to describe the equation of state for the fluids. The third parameter is a constant $M^2 = g\lambda/c_h^2$ defining as the ratio of a gravitational time scale to a sound speed time scale (this indicates the compressibility of the fluids), which has the effect of reducing the growth rate; λ is the wavelength of the interface perturbation, and c_h is the sound speed in the unperturbed heavy fluid.

For comparison purposes, we choose the same initial setup as used in Gardner *et al.*[33], namely, we introduce a small perturbation of an isothermal equilibrium flow with a flat interface separating exponentially stratified flow above and below the interface. As in the laboratory experiments, the heavy fluid lies *below* the light fluid and gravity is directed *upwards*. For simplicity, we take the same gas with $\gamma = 1.4$ for both the heavy and light fluids.

The unperturbed isothermal equilibrium we use, for flow in both the heavy and lighted fluids, is specified by

$$\begin{aligned}\rho &= \rho_0 e^{\beta_0(y-y_0)},\\ p &= p_0 + (\rho-\rho_0)g/\beta_0,\end{aligned}$$

where ρ_0 is equal to ρ_h (in the heavy fluid) or ρ_l (in the light fluid), $\beta_0 = \gamma g/c_0^2$ (c_0 sound speed), y_0 is the location of the unperturbed interface, and p_0 is the pressure at y_0. As in the previous example, we introduce a sinusoidal perturbation (9.3) on the interface to trigger the instability, and we incorporate the linear stability result for the perturbed states to initialize the flow, see Appendix A.2 for the analysis and solution.

Here we choose the following parameters in the run: density ratio $D = 10$ with density $\rho_h = 1$ and $\rho_l = 0.1$, $M^2 = 0.5$ with $c_h = 1$, $y_0 = 2$, $k = \pi$, and $\varepsilon = 0.03\pi/k$. The computational domain is a rectangular region ($[0,1] \times [0,4]$) with solid wall boundaries on the top and bottom and periodic boundaries on the left and right. Source terms in the equations are handled in a way similar to what is described in Section 5.2.

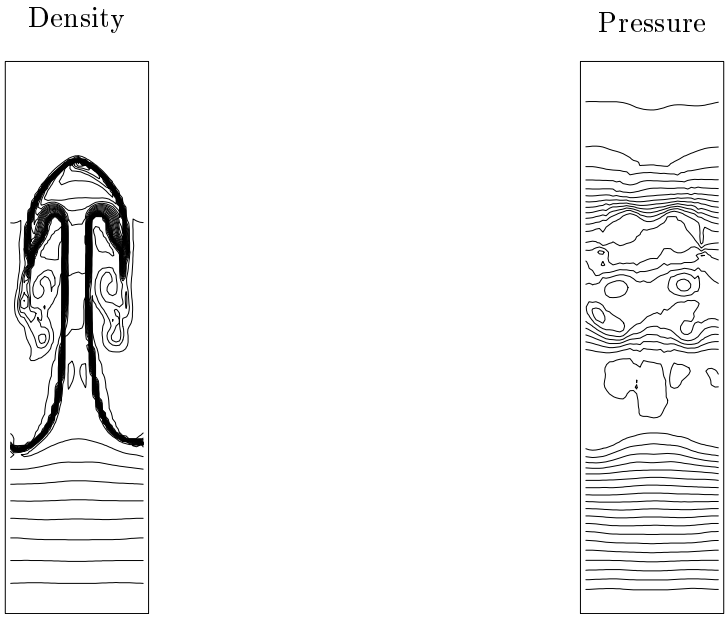
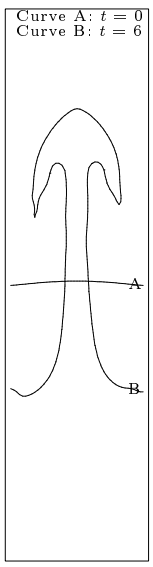


Figure 9.5: Contour plots in density and pressure for the Rayleigh-Taylor instability at time $t = 6$. Here the heavy fluid lies below the light fluid, and gravity is directed upwards. Parameters $D = 10$, $M^2 = 0.5$, and $\varepsilon = 0.03$ were used in the run.

The results are shown in Figures 9.5 and 9.6 up to time $t = 6$ using a 40×160 grid. Figure 9.5 shows the density and pressure contour plots, and Figure 9.6a shows the tracked interfaces at time $t = 0$ and $t = 6$. From these figures, the growth of the interface and the formation of a rising bubble and falling spike can be easily seen. In Figure 9.6b, we compare the history of the interface positions, $y_{\min}$ and $y_{\max}$ – the minimum and maximum vertical distance of the interface from the bottom boundary $y = 0$, showing the results of the numerical simulation and the linear theory. We observe good agreement of results in the small amplitude regime. In the large amplitude regime, qualitatively, our result agrees with the result shown in [33], at least as far as the dominant unstable mode is concerned. The results of [33] also show a secondary instability below the rising bubble. It is not clear which result is correct and work is continuing to clear up the discrepancy of the results.

It should be mentioned that this problem (and in general any unstable interface problem) is very sensitive to small perturbations arising from either the physics or numerics, and the solution may not converge when a mesh refinement study is performed in the absence of physical viscosity[33],[74]. This scenario of nonconvergence of unstable interfaces (for both the Rayleigh-Taylor and Kelvin-Helmholtz instabilities) has been further explored by Mulder, Osher, and Sethian[74]. They use the "Hamilton-Jacobi" level set formulation to evolve interfaces and add physical viscosity to the equations to study the zero (physical) viscosity limit of the Navier-Stokes equations. They observe improvement of convergence of results with larger values of the physical viscosity. They also demonstrate that given some amount of physical viscosity, there exists a fine enough grid so that the physical viscosity dominates the numerical viscosity, and so the results are unchanged with respect to further grid refinement.

a)



b)

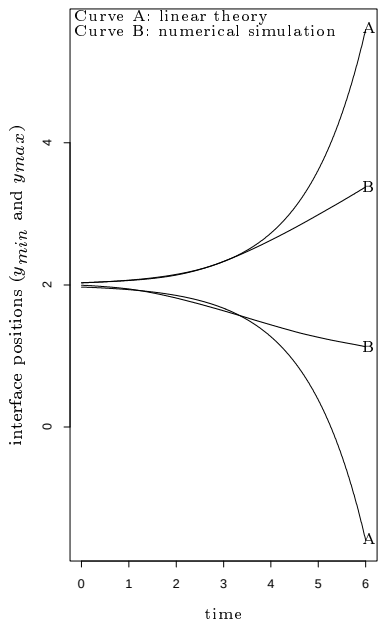


Figure 9.6: Results in the interface structure for the Rayleigh-Taylor instability. a) Tracked interfaces at time $t = 0$ and $t = 6$. b) A comparison of the history of interface positions y_{max} and y_{min} from the results of numerical simulation and linear theory.

9.3 Steady State Calculation

Finally, we apply our front tracking algorithm to another problem of great engineering interest: a steady state calculation for the Euler equations, and illustrate the potential power of using front tracking for steady state problems.

The problem we consider is a Mach 3 inflow over a 20^o ramp. It can be shown that, under certain assumptions on the boundary conditions, the steady state solution for this problem is an oblique shock with an angle $\theta = 37.8^o$ attached at the corner of the ramp[25]. This is an interesting and also difficult problem because the steady shock attaches at the ramp corner. Several other people have run similar problems using standard shock capturing methods with either body-fitted or Cartesian grids[10],[102]. They observe that difficulties at the ramp corner lead to extra entropy production near the wall giving an entropy layer near the wall and nonconvergence.

To simplify the problem, our calculation is based on a fixed "exact" grid in the sense that the exact shock location is inserted into an underlying grid. To make the grid even better, we use a body-fitted grid as the underlying grid, see Figure 9.7a. In this case, we are studying the accuracy of our numerical method on an "ideal" grid.

It is very encouraging that the numerical results obtained using our tracking algorithm on this particular grid do converge to the correct solution without spurious numerical artifacts at the tracked shock or along the boundary. An example is presented in Figure 9.7 with a 20×10 grid. In this run, a first order Godunov's method is employed for updating

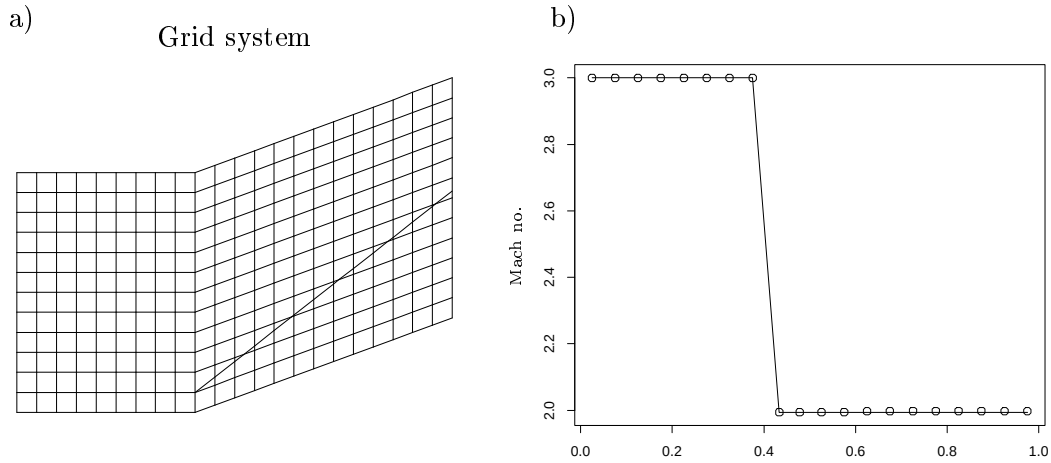


Figure 9.7: Steady state calculation for a Mach 3 inflow over a 20^o ramp. a) Grid system. b) Cross section of Mach number along the ramp. The solid line is the exact solution, and the dotted points are the numerical result.

the cell values in both regular and irregular cells, solid wall boundary conditions are used on the bottom boundary, and nonreflection-outflow boundary conditions are used on the remaining boundaries. Uniform Mach 3 flow ($\rho = 1$, $u = 1$, $v = 0$) is used as the initial condition for all grid cells, and the iteration is halted when the density variation from the previous time step to the current time step is less than the prescribed tolerance, 10^{-4} in this case, after 191 time steps (time $t = 3.57$). Here we have made no attempt to accelerate the convergence to steady state.

Extensions of this code to a Cartesian grid cut by the exact shock location and the boundary (following Berger and LeVeque[7]) is still in progress. In this case, the treatment of the solid wall boundary becomes complicated due to the fact that the cell which contains the ramp corner is subdivided by both the tracked shock and boundary segment.

Part III

Porous Media Flow

Chapter 10

OIL RESERVOIR SIMULATION

In many applications, the hyperbolic conservation laws may be combined with other types of partial differential equations, *e.g.*, an elliptic or parabolic PDE, in order to correctly describe the problem. In this instance, front tracking in the hyperbolic part of the entire system can still be a very useful tool to provide some vital information for solving the remaining part of the equations. Here we consider one such example: oil reservoir simulation in porous media, in which a hyperbolic conservation equation is coupled with an elliptic PDE. We will show some preliminary results for sample problems in both one and two space dimensions, and demonstrate the usefulness of using front tracking for this problem.

10.1 Preliminaries

We consider a simplified two phase flow in a porous medium in which diffusion, surface tension, gravity, and heterogeneity of the reservoir can be ignored. We consider a model problem for oil reservoir simulation in which the fluids are oil and water for immiscible displacement, or oil and solvent, such as CO_2, for miscible displacement. This model has been extensively studied in the past, particularly in the oil industry, because it simulates a process of secondary oil recovery where water or solvent is pumped into the oil field to force oil out of the wells, see, for example, Aziz and Settari[2], Peaceman[79], Glimm *et al.*[39],[40], and references therein for more detail.

For this two phase flow model, the governing equations consist of the following equations:

$$s_t + \nabla \cdot (\vec{q} f(s)) = \psi(s), \tag{10.1}$$

$$\vec{q} = -\kappa(s)\nabla p, \tag{10.2}$$

$$\nabla \cdot \vec{q} = \psi(s). \tag{10.3}$$

Here s denotes the saturation of the injected fluid ($s = 1$ for the injected fluid and $s = 0$ for the oil), $\vec{q}$, a vector, is the total velocity (oil velocity plus the injected fluid velocity), $f(s)$ is the so-called fractional flow function defined as the ratio in magnitude of the injected fluid velocity to the total velocity, $\psi(s)$ is the source term corresponding to the injection of fluid in wells and/or production of oil from wells, $\kappa(s)$ represents permeability divided by viscosity, and p is the pressure. In the above system, Equation (10.1) expresses the conservation of mass of the injected fluid, Equation (10.2) is Darcy's law which states that the total velocity is proportional to the pressure gradient, and Equation (10.3) expresses that the underlying fluids are incompressible. The derivation of these equations along with other physics of flow through porous media can be found in Scheidegger[92].

In practice, the functions $\kappa(s)$, $f(s)$ we use are

$$\kappa(s) = s^2 + \mu^{-1}(1 - s)^2, \tag{10.4}$$

$$f(s) = s^2/\kappa(s), \tag{10.5}$$

for immiscible displacement, and

$$\kappa(s) = (s + \mu^{-1/4}(1-s))^4, \tag{10.6}$$

$$f(s) = s, \tag{10.7}$$

for miscible displacement where μ is the viscosity ratio of the underlying fluid; the oil viscosity over the injected fluid viscosity.

It is interesting to note that in this model there is a dimensionless parameter called the frontal mobility ratio M defining as $M = \kappa(s_l)/\kappa(s_r)$ which characterizes the stability of the interface (for $M \leq 1$ the interface is stable, while for $M > 1$ the interface is unstable, see, for example, [19],[39],[41],[52],[111]); s_l and s_r are the saturations behind and ahead of the interface respectively ($s_l > s_r$). It is easy to verify that the relationship between the mobility ratio M and the viscosity ratio μ is simply $M = 2(1-(1+\mu)^{-1/2})$ for immiscible displacement and $M = \mu$ for miscible displacement[58].

Notice that using the incompressibility condition (10.3), the conservation equation (10.1) can be rewritten as a nonconservative hyperbolic PDE

$$s_t + \vec{q} \cdot \nabla f(s) = \psi(s) \tag{10.8}$$

with velocity $\vec{q}$ considered as known. In fact, if $\psi(s) = 0$, with the above mentioned fractional flow function, Equation (10.8) is simply the Buckley-Leverett equation for the immiscible displacement and the linear advection equation for the miscible displacement. Because of the ease in solving the Riemann problems for these equations (see [24] for the construction of Riemann solution for the Buckley-Leverett equation), this nonconservative form of the saturation equation is always used in practice. Nevertheless to obtain the velocity $\vec{q}$, from Equation (10.2), we need to know the pressure p. It is easy, however, to derive the governing equation for p by simply substituting (10.2) into (10.3), which yields an elliptic PDE

$$\nabla \cdot (-\kappa(s)\nabla p) = \psi(s). \tag{10.9}$$

Note that in general this elliptic equation (10.9) would have discontinuous coefficients $\kappa(s)$ across the discontinuities because of the jumps in saturation and also the viscosity. In this case, for a material interface, from the dynamic boundary condition the pressure should be continuous at the discontinuity, and from the kinematic boundary condition the normal velocity at the discontinuity should be continuous also. Based on this fact and others, we can easily show that the tangential component of the pressure gradient is continuous as is the normal derivative of $\kappa(s)p$, but the normal component of the pressure gradient is not since $\kappa(s)$ is not[41].

10.2 Algorithm

To solve this two phase flow model, a very popular approach is the so-called IMPES (implicit pressure and explicit saturation) procedure in which in each time step the hyperbolic saturation equation (10.8) and the elliptic pressure equation (10.9) are dealt with separately and sequentially. It is described in the following algorithm in the context of using front tracking algorithm to handle the hyperbolic part of the equation:

Algorithm 10.1

1) Given the saturation s, solve the elliptic PDE (10.9), obtaining the pressure p.

2) Compute the total velocity $\vec{q}$ by differencing the pressure p obtained from Step 1 and substituting into the velocity equation (10.2).

3) Solve the hyperbolic equation (10.8) using the front tracking algorithm to update the saturation s.

Since the front tracking algorithm introduced in the previous chapters is very easy to apply for the hyperbolic saturation equation (10.8), here we focus our discussions on Steps 1 and 2 of the algorithm.

Step 1: In solving the elliptic pressure equation (10.9) with possibly discontinuous coefficients $\kappa(s)$, the conventional approach is to employ a finite element method having elements aligned with the interface. In one space dimension, this can be done quite easily and we describe one approach below. In two space dimensions, however, this is difficult to do as seen in the work done by McBryan and others[43],[72]. Here as a first attempt to tackle this problem in two space dimensions, we adopt a simpler approach by employing a standard five-point stencil finite difference method on a uniform grid, *i.e.*, we ignore the appearance of the discontinuities, even though we know their location explicitly. Doing so causes some smearing of the pressure profile, and so less accurate result as we might hope to obtain. As seen from the results shown below, we still obtain reasonable solutions, however. In future work, we hope to improve upon this method by using, for example, the immersed interface method developed by LeVeque and Li[66].

To be more specific, let us discuss some examples. We first consider a one-dimensional example. Consider the Dirichlet boundary condition at each side of the boundary and source terms $\psi(s) = 0$ in the entire computational domain; $x \in [0, 1]$. The elliptic pressure equation (10.9) then corresponds to a second order ordinary differential equation

$$(-\kappa(s)p_x)_x = 0 \tag{10.10}$$

in one space dimension with boundary conditions $p(x{=}0) = a$ and $p(x{=}1) = b$, where $\kappa(s)$ is known spatially. For simplicity, we take a and b to be constants for all time, although in general they can vary with time.

To discretize (10.10), we use a three-point finite difference method on a nonuniform grid by first taking a backward difference for the outer derivative and then a forward difference for the inner derivative; collecting terms, we get the following difference formula

$$-\frac{\kappa_{i-\frac{1}{2}}}{h_{i-1}}\, p_{i-1} + \left(\frac{\kappa_{i-\frac{1}{2}}}{h_{i-1}} + \frac{\kappa_{i+\frac{1}{2}}}{h_i}\right) p_i - \frac{\kappa_{i+\frac{1}{2}}}{h_i}\, p_{i+1} = 0 \tag{10.11}$$

for node i, where $\kappa_{i-\frac{1}{2}} = \kappa(s(x_{i-\frac{1}{2}}))$, $p_{i-1} = p(x_{i-1})$, and h_i is the mesh size, see Figure 10.1a for the notation used here. Going through all the nodal points for $i = 1, 2, \cdots, N$, and using the boundary conditions, we obtain a symmetric positive definite tridiagonal linear system for the unknown pressure p.

Note that in the above discretization we have used a staggered grid representation for the pressure p and saturation s, *i.e.*, p is defined at the cell interface, and s is defined

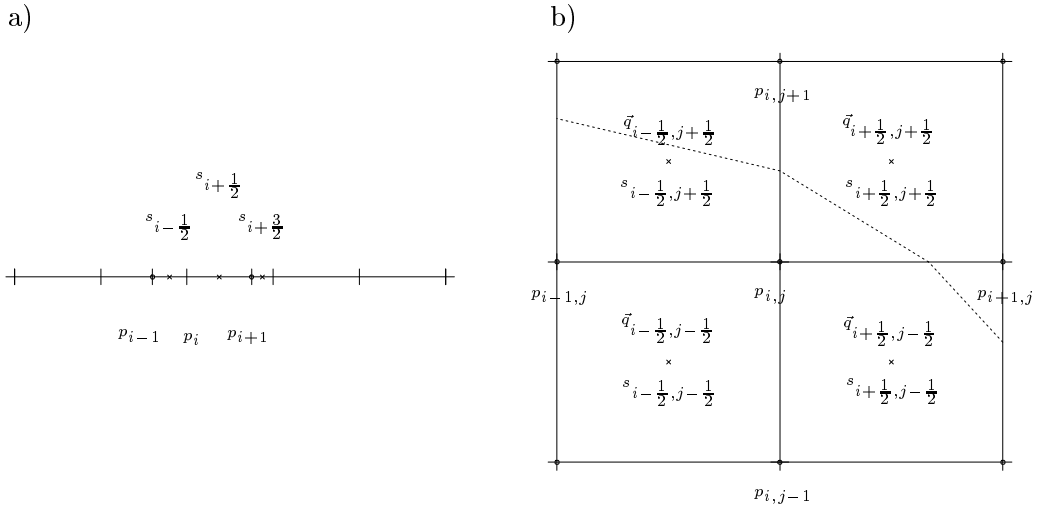


Figure 10.1: The computational grid used for solving the elliptic pressure equation (10.9) at Step 1 of the algorithm. a) A nonuniform grid in one space dimension. b) A uniform grid in two space dimensions where we ignore the appearance of the tracked discontinuity (shown as dashed line) on the grid. Note that the pressure in each figure is defined at the grid point, while the velocity and saturation are defined at the cell center; a staggered grid approach.

at the cell center. In addition, it was discretized on a nonuniform grid rather than on a uniform grid. Doing so prevents us from differencing across discontinuities, and hence yields accurate pressures and hence velocities. This finite difference discretization (10.11) reduces to the standard three-point stencil finite difference method if tracked points disappear.

Now we consider a two-dimensional example. We consider a model problem in which the geometry is a standard five-spot pattern with fluid (water or solvent) being injected into the center of a unit square domain ($\Omega = [0,1] \times [0,1]$) and the oil being recovered from the four corners. For this model problem, we use the Neumann boundary conditions

$$\partial p / \partial n = 0 \tag{10.12}$$

on the boundaries $\partial\Omega$, where n is the direction normal to the boundary pointing toward the outside of the computational domain. (This gives zero normal velocities on the boundaries.) We must then solve the elliptic equation

$$(-\kappa(s)p_x)_x + (-\kappa(s)p_y)_y = \psi(s) \tag{10.13}$$

with an appropriate choice of source terms $\psi(s)$ for the injection and production of fluid at the wells.

Denote by ψ_{i} and ψ_{o} the source terms for the injection fluid and production of oil respectively. The source ψ_{i} centered at (x_0, y_0) is taken of the form

$$\psi_{\mathrm{i}}(x - x_0, y - y_0) = \begin{cases} (1 + \cos(\pi r / r_\delta))/2r_\delta & r \le r_\delta \\ 0 & r > r_\delta \end{cases} \tag{10.14}$$

where r_δ is a constant (we choose $r_\delta = 0.05$ in the numerical results shown below.) and $r^2 = (x-x_0)^2+(y-y_0)^2$ is the distance from the center (x_0, y_0). With $\psi_{\rm I}$ in this form, the source is spread to a circular region of radius r_δ with center (x_0, y_0); it has the biggest strength with magnitude $1/r_\delta$ at the center. Analogously, source (or more appropriately sink) $\psi_{\rm O}$ centered at (x_0, y_0) is defined by putting a minus sign in front of the function (10.14). Source terms of the above form were introduced by Peskin[81] and used widely in the immersed boundary method for representing singular sources.

It is well known that the solution for this problem (the elliptic equation with zero Neumann boundary conditions) may fail to exist if the consistency condition

$$\int_\Omega \psi(s)\, dA = 0 \tag{10.15}$$

is not satisfied. In our model problem with the above chosen source terms $\psi_{\rm I}$ and $\psi_{\rm O}$, it is easy to check that condition (10.15) is satisfied, and hence there is a solution. Although the solution is determined only up to an additive constant, this causes no additional problem because only its first derivative will be used in the algorithm to compute the velocity.

We use a staggered grid in which the pressure p is defined at grid points, and the saturation s is defined at cell centers (see Figure 10.1). To discretize (10.13), we use a standard five-point stencil finite difference method on a uniform grid. This is done using a similar procedure to what we used for discretizing the one-dimensional pressure equation (10.10). Applying this in both the x- and y-directions, and collecting terms, yields the following difference formula

$$a_{ij}\, p_{i,j-1} + b_{ij}\, p_{i-1,j} + c_{ij}\, p_{ij} + d_{ij}\, p_{i+1,j} + e_{ij}\, p_{i,j+1} = r_{ij} \tag{10.16}$$

with

$$\begin{aligned}
a_{ij} &= -\kappa(s_{i,j-\frac{1}{2}})/h^2 \\
b_{ij} &= -\kappa(s_{i-\frac{1}{2},j})/h^2 \\
d_{ij} &= -\kappa(s_{i+\frac{1}{2},j})/h^2 \\
e_{ij} &= -\kappa(s_{i,j+\frac{1}{2}})/h^2 \\
c_{ij} &= -(a_{ij}+b_{ij}+d_{ij}+e_{ij}) \\
r_{ij} &= \psi_{\rm I}(x_i - x_\otimes, y_j - y_\otimes) - \sum_{\forall (x_\odot, y_\odot)} \psi_{\rm O}(x_i - x_\odot, y_j - y_\odot)
\end{aligned}$$

for node (i, j), see Figure 10.1b for the notation used here. Here $\kappa(s_{*,\diamond}) = \kappa(s(x_*, y_\diamond))$ is defined using the harmonic average of the neighboring cells, *e.g.*,

$$\kappa(s_{i,j-\frac{1}{2}}) = 2\left(\frac{1}{\kappa(s_{i-\frac{1}{2},j-\frac{1}{2}})} + \frac{1}{\kappa(s_{i+\frac{1}{2},j-\frac{1}{2}})}\right)^{-1},$$

$(x_\otimes, y_\otimes)$ denotes the center of the injection well which is at the center of the domain, and $(x_\odot, y_\odot)$ denotes the centers of the production wells which are at the corners of the domain, in our case.

Because of the boundary conditions (10.12), the above difference formula (10.16) needs to be modified for nodes at the boundaries. It is easy to show that by introducing fictitious nodes outside the computational domain and approximating the boundary conditions with a central difference formula at the boundary, the pressure at a fictitious node is simply equal to the pressure at the interior node adjacent to the boundary, *e.g.*, $p_{i,2} = p_{i,0}$ for all the i on the $y = 0$ boundary. Since we have $\partial s/\partial n = 0$ at the boundaries, the coefficient of the pressure in (10.16) at the fictitious node is also the same as the pressure at the interior node adjacent to the boundary. Having known this, we can then modify the difference equation (10.16) by ignoring nodes outside the domain and multiplying the coefficient of the related interior nodes by two. Alternatively, the above result can be derived more directly by approximating the boundary condition with a one-sided difference formula, expanding this difference formula at the boundary using the Taylor series expansion, and employing the elliptic equation (10.13) that takes into account the differential equation at the boundary node[112].

Going through all the nodal points for $i = 1, 2, \cdots, N$ and $j = 1, 2, \cdots, N$, using row-wise ordering, we obtain a block-tridiagonal linear system for the unknown pressure p. For illustration purposes, we display the coefficient matrix A in the case where $\kappa(s) = 1$ in the elliptic equation (10.13) and $N = 3$, *i.e.*,

$$A = \frac{1}{h^2}\begin{pmatrix}
4 & -2 & & -2 & & & & & \\
-1 & 4 & -1 & & -2 & & & & \\
 & -2 & 4 & & & -2 & & & \\
-1 & & & 4 & -2 & & -1 & & \\
 & -1 & & -1 & 4 & -1 & & -1 & \\
 & & -1 & & -2 & 4 & & & -1 \\
 & & & -2 & & & 4 & -2 & \\
 & & & & -2 & & -1 & 4 & -1 \\
 & & & & & -2 & & -2 & 4
\end{pmatrix}$$

where h is the underlying uniform grid size, assuming that a square grid is used. Analogous to the original differential equation, the solution to this linear system may not exist, unless the sum on the right hand side of the linear system equals zero to satisfy a discrete version of the consistency condition (10.15) for this problem. It is easy to check that the sum on the right hand side of our linear system does satisfy this condition. Here this linear system is solved using the incomplete LU generalized minimum residual method in the SLAP (Sparse Linear Algebra Package) Library.

Step 2: Having calculated the pressure p, we can compute the total velocity $\vec{q}$ by differencing p and putting the result back in the velocity equation (10.2). In one space dimension with $\vec{q} = u$, this results in

$$u_{i+\frac{1}{2}} = -\kappa(s_{i+\frac{1}{2}})(p_{i+1} - p_i)/h_i$$

when a forward difference on p is employed, where $u_* = u(x_*)$ is defined at the cell center. Since the method we use for updating the saturation is based on solving the Riemann problem at each cell interface, it is necessary to also compute the velocity u at the cell interface. One simple approach is to take an average of two neighboring velocities and

assign it to the cell interface, namely, set

$$u_i = \frac{1}{2}(u_{i-\frac{1}{2}} + u_{i+\frac{1}{2}}).$$

In two space dimensions with $\vec{q} = (u, v)$, we do the same thing as in one dimension by computing the cell averaged velocity defined at the cell center as follows:

$$\begin{pmatrix} u_{i+\frac{1}{2},j+\frac{1}{2}} \\ v_{i+\frac{1}{2},j+\frac{1}{2}} \end{pmatrix} = -\frac{1}{2h} \begin{pmatrix} \kappa(s_{i+\frac{1}{2},j})(p_{i+1,j} - p_{i,j}) + \kappa(s_{i+\frac{1}{2},j+1})(p_{i+1,j+1} - p_{i,j+1}) \\ \kappa(s_{i,j+\frac{1}{2}})(p_{i,j+1} - p_{i,j}) + \kappa(s_{i+1,j+\frac{1}{2}})(p_{i+1,j+1} - p_{i+1,j}) \end{pmatrix}$$

where $\kappa(s_{*,\diamond})$ is computed using the harmonic average of the neighboring cells. This gives the velocities for the regular cells. Since the grid we used for updating the saturation consists of not only the regular cells, but also the irregular cells, we need to compute the velocities for the irregular cells also. This can be done by employing an interpolation scheme that makes use of the cell averaged velocities on the neighboring regular cells, *e.g.*, by interpolating data based on a rectangular box over the neighboring cells. The velocity at the cell interface for both regular and irregular cells can be computed in a manner similar to the one-dimensional case.

Before presenting numerical results, we make two remarks on Step 3 of the algorithm. First, in solving the Riemann problem at the each cell interface, the cell interface velocity described above is used for the velocity $\vec{q}$ appearing in the saturation equation (10.8). Second, the source term appearing in the saturation equation (10.8), only the $\psi_{\rm i}$ term exists, is treated as a boundary condition in each time step, *i.e.*, we reset saturation $s = 1$ in the region where the injection source $\psi_{\rm i}$ is in effect.

10.3 Numerical Results

We now show some preliminary results for this two phase model.

One space dimension. We first show some one-dimensional results for our one-dimensional model problem discussed in the previous section. As initial conditions, we take saturation $s = 0$ (pure oil) in the entire computational domain, except for the first grid cell ($x \in [0, h]$, where $h = 0.01$ is the mesh size) which has $s = 1$ (pure water or solvent). As boundary conditions, we use Dirichlet data, $p(x{=}0) = 1$ and $p(x{=}1) = 0$, for the elliptic pressure equation, and fixed boundary conditions with $s = 1$ on the left hand side boundary and outflow boundary condition on the right hand side boundary for the hyperbolic saturation equation.

Results are shown in Figure 10.2. In Figure 10.2a and c, we show results for the immiscible displacement calculations where the saturation and pressure are shown, respectively, with three different viscosity ratios, $\mu = 1, 2, 10$, at time $t = 0.8$. It is clearly seen that as the viscosity ratio increases the saturation profile behind the discontinuity becomes more and more depressed, and it becomes harder and harder to displace the resident oil. It can also be observed in Figure 10.2c that the pressure profile has a kink at the discontinuity for some viscosity ratios. This is a consequence of the jumps in saturation and the viscosity there. It is easy to check that as the frontal mobility ratio M approaches one, the jump in the pressure gradient approaches zero. This can be seen in Figure 10.2c where the frontal mobility ratios we use are $M = 0.586$ for $\mu = 1$, $M = 0.845$ for $\mu = 2$, and $M = 1.397$ for $\mu = 10$.

In Figure 10.2b and d, we show results for the miscible displacement calculations with the same viscosity ratios and stopping time as for the immiscible displacement calculations. Notice that the saturation profile remains the same shape for all the viscosity ratios. This is due to the fact that only a single phase of fluid (solvent or oil) can exist in a volume of the miscible porous medium. This is not so for fluids in an immiscible environment, however. In Figure 10.2d, kinks are clearly seen at the discontinuities in the case where the frontal mobility ratio is not equal to one there. (Recall that for miscible displacement $M = \mu$.)

The above results were obtained using the high resolution front tracking method with Courant number $\nu = 0.9$.

Two space dimensions. We now show two-dimensional results for our two-dimensional model problem with the five-spot pattern. Here the problem of interest is to study stability of the interfaces under various mobility ratios. As initial saturation, we use $s = 1$ (pure water or solvent) inside a *perturbed* circular interface and $s = 0$ (pure oil) outside the perturbed interface. As boundary conditions, we have Neumann boundary conditions (10.12) for the elliptic equation, and reflecting boundary conditions $\partial s/\partial n = 0$ for the saturation equation. The source $\psi_{\rm I}$ that corresponds to the injection of fluid at the wells is located at the center of the domain, and sinks $\psi_{\rm O}$ that correspond to the production of oil from the wells are located at the corners of the domain.

Results are shown in Figures 10.3 and 10.4. In Figure 10.3a and b, we show results for the immiscible displacement calculations where the evolution of the tracked interfaces are shown with viscosity ratio $\mu = 2$ and $\mu = 10$, respectively. Note that with $\mu = 2$ we have frontal mobility ratio $M = 0.845$. So based on the stability criterion mentioned in the beginning of this chapter this interface is stable under small perturbations. Note that our numerical result shown in Figure 10.3a reflects this fact. Moreover, our result shown in Figure 10.3b also predicts the right behavior of an unstable front where $M = 1.397$ there.

In Figure 10.4a and b, we show results for the miscible displacement calculations where the evolution of the tracked fronts are shown with viscosity ratio $\mu = 1$ and $\mu = 10$, respectively. Again, based on the mobility ratio we can readily predict that the interface is stable for $\mu = 1$ ($M = 1$), while the interface is unstable for $\mu = 10$ ($M = 10$). Our results give a correct indication of the stability of the interface.

The above two-dimensional calculations were run on a 50×50 grid. For updating the saturation, we use the high resolution front tracking method with Courant number $\nu_0 = 0.9$. For similar calculations, see [39],[58].

Finally, in Table 10.1, we report results on the timing (the CPU time) of the two-dimensional oil reservoir simulations. It is easy to observe that Step 1 of solving the elliptic equation leads the usage of the CPU time (the case for the miscible calculation with $\mu = 1$ is a special situation where the pressure is constant for all time), Step 3 of solving the hyperbolic equation is the second most expensive, and Step 2 of evaluating the velocity is the last. Hence it is desirable to use a fast Poisson solver for this problem so as to improve the performance of this algorithm. Note that the CPU time on the second column of the table consists of time for the integration step (three basic steps in Algorithm 10.1) and the IO (input and output). The above calculations were run on a DEC station 5000/200 using a Fortran 77 complier under the Ultrix operating system.

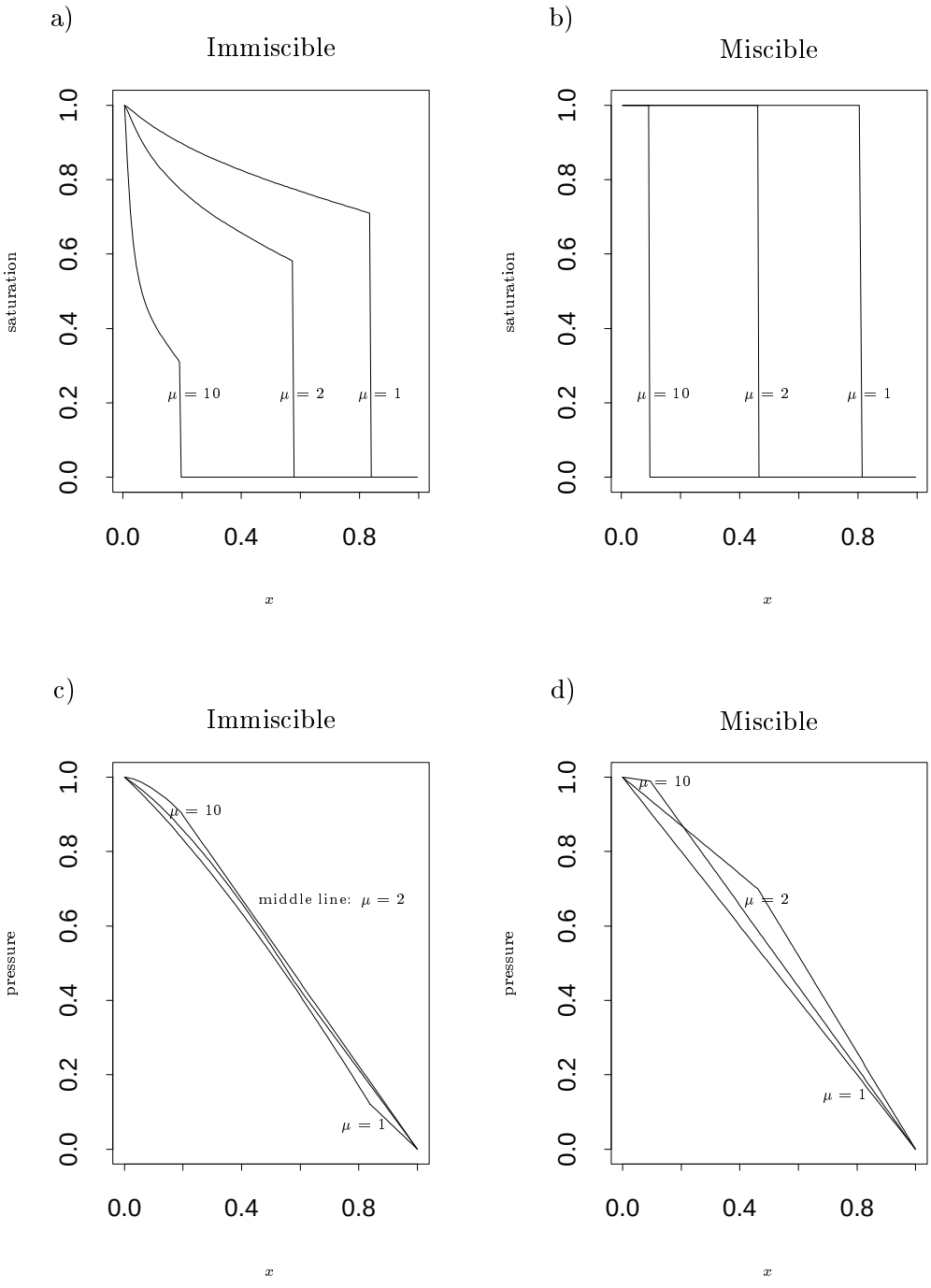


Figure 10.2: Results for the one-dimensional oil reservoir simulation. Figures a) and c) show results for the immiscible displacement computations. Figures b) and d) show results for the miscible displacement computation. In each figure, three viscosity ratios, $\mu = 1, 2, 10$, were used in the test up to time $t = 0.8$.

a) $\mu = 2$ b) $\mu = 10$

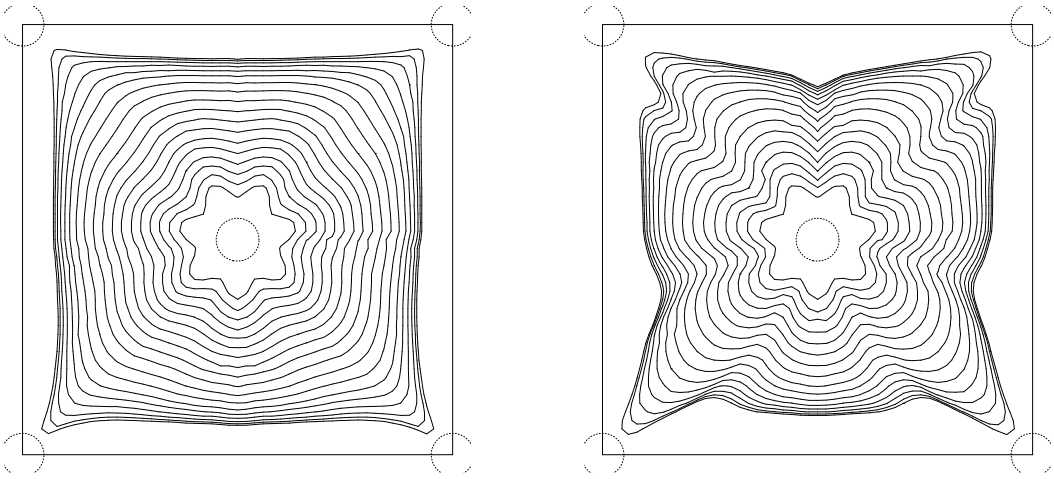


Figure 10.3: Evolution of the tracked interfaces for the immiscible displacement oil reservoir simulation, plotted every other time step. a) $\mu = 2$ ($M = 0.845$) up to time $t = 10.5$. b) $\mu = 10$ ($M = 1.397$) up to time $t = 5.75$.

a) $\mu = 1$ b) $\mu = 10$

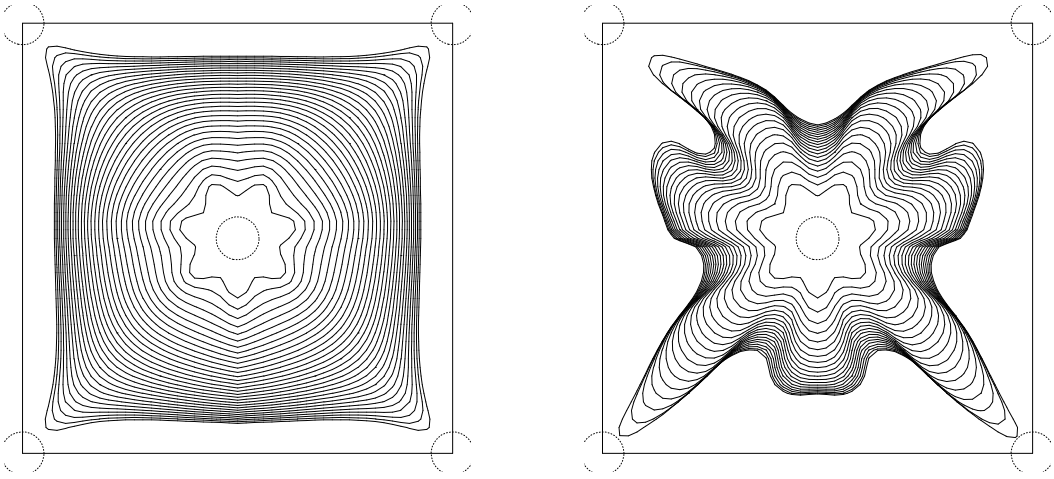


Figure 10.4: Evolution of the tracked interfaces for the miscible displacement oil reservoir simulation, plotted every 5 time steps. a) $\mu = 1$ ($M = 1$) up to time $t = 15$. b) $\mu = 10$ ($M = 10$) up to time $t = 9.4$.

Table 10.1: Timing of the two-dimensional oil reservoir simulations on a DEC station 5000/200.

	CPU time (seconds)	CPU time/step			Time steps
		Step 1	Step 2	Step 3	
immiscible $\mu = 2$	810	15	1	7.8	34
immiscible $\mu = 10$	738	15.1	1	8.4	30
miscible $\mu = 1$	1350	2.6	1.3	5	150
miscible $\mu = 10$	2360	16.5	2.6	5.7	95

Chapter 11

CONCLUSIONS

11.1 Thesis Summary

In this thesis, we have developed and studied a simple front tracking approach that models the propagation of discontinuous solutions for nonlinear hyperbolic systems of conservation laws with source terms in both one and two space dimensions. In this approach, we use a uniform underlying grid with some grid cells subdivided by tracked interfaces, made up of moving points in one space dimension and curves in two space dimensions, approximately aligned with the discontinuities in the flow field. In each time step, we solve Riemann problems at the tracked interfaces using the values from adjacent cells as data, and follow strong waves (shocks or interfaces) to determine a new set of tracked interfaces at the end of the time step. A conservative high resolution finite volume method based on the wave propagation approach is then applied on the resulting nonuniform grid to update the cell values. Potential problems with stability are dealt with by the use of a "large time step" method (see Chapters 2 and 6). Since the new interface locations have been chosen carefully, the resulting solution remains sharp and is smooth away from these new interfaces. The old interfaces can then be eliminated by recombining the adjacent cells. This front tracking algorithm is described in more detail in Chapters 3 and 7 for the one- and two-dimensional cases, respectively.

Error analysis. To examine stability and accuracy of the algorithm, in Chapters 4 and 8 we performed error estimation for the one- and two-dimensional algorithms. The results presented there show that our algorithm is stable even if some of the small cells are orders of magnitude smaller than the regular cell that is used to determine the time step. In addition, our algorithm is first order and *almost* second order accurate in the 1-norm for problems involving contact discontinuities and shocks, respectively, in which a high resolution method is used in the experiments (see Sections 4.1 and 8.1). Note the above results hold for both the one- and two-dimensional algorithms. As noted there, our tracking results are much better than what is obtained with shock capturing.

Concerning the error behavior in cells near the tracked interfaces, we have also investigated several issues that can lead to loss of accuracy, such as the choice of slopes in neighboring cells, the use of nonuniform and time-dependent grid, and the linearization of wave interactions due to the use of the large time step method. From this study, we found that significant improvement of the errors in cells near tracked contact discontinuities can be obtained using "one-sided" slopes, though this gives much less improvement in cells near tracked shocks (see Section 4.2). In addition, there is some loss of accuracy due to the use of time-dependent nonuniform grids, particularly in the max-norm for the first order method (see Sections 4.3 and 8.3). It has been seen from a one-dimensional model problem that the errors introduced by the linearization of wave interactions in the large time step method are of magnitude $O(h)$, and they only occur locally near the tracked interfaces. Because of this, the 1-norm error of the method has not been severely affected by this loss

of accuracy (see Section 4.1). For stability, there is no problems with the linear wave interaction approach for most calculations, except in an extreme case where a strong rarefaction wave overtakes a shock. In that case, we modify the method so that the interaction of the tracked discontinuity and the weak waves is handled "exactly" (see Section 4.4).

Lastly, we have observed very nice results in the accuracy of the tracked front location using our front-tracking algorithm in both the one- and two-dimensional cases (see, *e.g.*, Figures 4.8 and 4.11, and Table 8.1). An approach that replaces the piecewise linear representation of the tracked front by a piecewise quadratic representation did not significantly improve the tracked front accuracy (see Section 8.2).

Applications. To demonstrate the potential power of our front tracking algorithm on more complex problems, a wide variety of problems have been solved to validate the algorithm for problems involving shock waves and interfaces arising in gas dynamics. In one space dimension, the examples considered are a double piston problem, the Woodward-Colella blast wave problem, the steady quasi one-dimensional nozzle flow, and unstable detonation waves. In two space dimensions, they are radially symmetric shock waves, a shock-vortex interaction, a shock-ramp interaction, the Kelvin-Helmholtz and Rayleigh-Taylor instabilities, and a steady state calculation for a supersonic flow over a ramp. These results show the effectiveness of our front tracking algorithms in both one and two space dimensions. They also show the importance of using front tracking for these problems (see Section 3.3, Chapter 5, Section 7.3, and Chapter 9).

To further test the capability of our front tracking algorithm, we also considered a model problem arising from oil reservoir simulation. In this case, we need to solve a coupled system of elliptic and hyperbolic partial differential equations. We use an IMPES procedure to do this, in which the hyperbolic equation and the elliptic equation are dealt with separately and sequentially in each time step. Here the hyperbolic equation is solved using the front tracking algorithm and the elliptic equation is solved using a standard five-point finite difference method on a uniform grid. Our preliminary results for some sample problems indicate that front tracking is a very useful tool for this problem also (see Chapter 10).

11.2 Future Research

Even though our front tracking algorithm is quite successful in solving many practical problems, there are many aspects that have not been handled as well as we might hope, particularly for two-dimensional problems. Here we sketch some of them, and describe future work.

Improve resolution near the tracked interfaces. As we have seen from the two-dimensional error analysis performed in Chapter 8, there is some loss of accuracy of our front tracking algorithm in cells near the tracked interfaces. From the Kelvin-Helmholtz unstable interface problem in Section 9.2.1, we also see that our result is not as sharp as that obtained from another front tracking method. Because of this resolution discrepancy near the tracked interfaces, we observe different solution behavior for the Rayleigh-Taylor problem (see Section 9.2.2) obtained from our method vs. another tracking method. For the purpose of clearing up the difference as well as improving the algorithm, we plan to do the following work:

1) Explore various approaches that take account of slopes information for the irregular cells. Some possible approaches has been mentioned in Section 6.5. Although the work will increase a great deal by considering these approaches to the method, it is still worth while studying them in depth. Ultimately, we would like to find an *efficient* way to do this.

2) Explore various approaches that achieve high resolution of the tracked front. In this case, we may want to use curve-fitting or other interpolation technique to construct a smoother parametric curve to higher order, and also use a higher order representation for the grid interfaces.

It is important to note that we should consider this work as a whole, because the resolution near the tracked interfaces depends not only on the accuracy of the grid we construct, but also on the accuracy of the finite volume method we use on the grid. We should also keep in mind that we want to modify the method so that the numerical diffusion is as small as possible. For some interface problems, we might need to put more restrictions on the method so that the mass of each of two distinct fluids is conserved independently, with no leaking across the interface.

Code development. So far, to simplify programming, our current version of the two-dimensional front tracking code is only capable of dealing with discontinuities that have sufficiently smooth structure; the splitting of fronts as well as the collision of fronts are not allowed in the program. Because of this, the applicability of this code is limited to simple front geometries. As a first step toward tackling more complex problems, we need to consider using more general data structures in the code that can take account of complicated topological changes in the front structure. Some ideas suggested in [38],[42] are very valuable here. Another modification of the code that would enhance our ability to solve complex problems is to couple front tracking with local adaptive mesh refinement as has already been done for some one-dimensional problems. Doing so would be particularly useful for problems involving some internal structures near the tracked discontinuities, such as in the detonation wave computation.

Finally, work is continuing in the application of our front tracking algorithm to real applications. In particular, we are interested in studying physical effects, such as surface tension, viscosity, chemically reactions, moving sources, and more general equations of state, on the solutions of hyperbolic conservation laws arising in various situations (*e.g.*, in gas, water, porous media, or elastic and plastic materials).

BIBLIOGRAPHY

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms.* Addison-Wesley, 1987.

[2] K. Aziz and A. Settari. *Petroleum Reservoir Simulation.* Elsevier, London and New York, 1979.

[3] J. B. Bell, P. Colella, and M. L. Welcome. Conservative front-tracking for inviscid compressible flow. *UCRL-JC-105251, preprint*, 1991. Present at AIAA 10th Computational Fluid Dynamics Conference, Honolulu, Hawaii, June 24-26.

[4] M. Ben-Artzi and J. Falcovitz. An upwind second order scheme for compressible duct flow. *SIAM J. Sci. Stat. Comput.*, 7(3):744–768, 1986.

[5] M. J. Berger. On conservation at grid interfaces. *SIAM J. Numer. Anal.*, 24(5):967–984, 1987.

[6] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84, 1989.

[7] M. J. Berger and R. J. LeVeque. Stable boundary conditions for Cartesian grid calculations. *Computing Systems in Engineering*, 1:305–311, 1990.

[8] M. J. Berger and R. J. LeVeque. A rotated difference scheme for Cartesian grids in complex geometries. *AIAA paper CP-91-1602*, 1991.

[9] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.

[10] N. Botta. *Private communication*, 1991.

[11] A. Bourlioux. *Numerical Study of Unstable Detonations.* Ph.D. thesis, Princeton University, June, 1991.

[12] A. Bourlioux, A. J. Majda, and V. Roytburd. Theoretical and numerical structure for unstable one-dimensional detonations. *SIAM J. Appl. Math.*, 51:303–343, 1991.

[13] J. U. Brackbill, D. B. Kothe, and C. Zemach. A continuum method for modeling surface tension. *J. Comput. Phys.*, 100:335–354, 1992.

[14] S. Chandrasekhar. *Hydrodynamic and Hydromagnetic Stability.* Clarendon Press, 1961.

[15] T. Chang and L. Hsiao. *The Riemann Problem and Interaction of Waves in Gas Dynamics.* Longman Scientific & Technical, England, 1989.

[16] I-L. Chern and P. Colella. A conservative front tracking method for hyperbolic systems of conservation laws. *UCRL-97200, LLNL*, 1987.

[17] I-L. Chern, J. Glimm, O. A. McBryan, B. Plohr, and S. Yaniv. Front tracking for gas dynamics. *J. Comput. Phys.*, 63:83–110, 1986.

[18] A. J. Chorin. Flame advection and propagation algorithm. *J. Comput. Phys.*, 35:1–11, 1980.

[19] A. J. Chorin. The instability of fronts in a porous medium. *Commun. Math. Phys.*, 91:103–116, 1983.

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

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

[22] P. Colella, A. Majda, and V. Roytburd. Fractional step methods for reacting shock waves. In *Lectures in Applied Mathematics, vol 24*, pages 459–477. American Mathematical Society, 1986.

[23] P. Colella, A. Majda, and V. Roytburd. Theoretical and numerical structure for reacting shock waves. *SIAM J. Sci. Stat. Comput.*, 7(4):1059–1080, 1986.

[24] R. E. Concus and W. Proskurowski. Numerical solution of a nonlinear hyperbolic equation by the random choice method. *J. Comput. Phys.*, 30:153–166, 1979.

[25] R. Courant and K. O. Friedrichs. *Supersonic Flow and Shock waves.* McGraw-Hill, New York, 1954.

[26] C. M. Dafermos. Polygonal approximation of solution to the initial value problem for a conservation law. *J. Math. Anal. Appl.*, 38:33–41, 1972.

[27] S. F. Davis. A rotationally biased upwind difference scheme for the Euler equations. *J. Comput. Phys.*, 56:65–92, 1984.

[28] D. S. Dosanjh and T. M. Weeks. Interaction of a starting vortex as well as a vortex street with a traveling shock wave. *AIAA J.*, 3:216–223, 1965.

[29] B. Einfeldt, C. D. Munz, P. L. Roe, and B. Sjogreen. On Godunov type methods near low densities. *J. Comput. Phys.*, 92:273–295, 1991.

[30] J. L. Ellzey, J. M. Picone, and E. S. Oran. The interaction of a shock with a compressible vortex. *preprint*, 1992.

[31] W. Fickett and W. W. Wood. Flow calculations for pulsating one-dimensional detonations. *Phys. Fluids*, 9(5):903–916, 1966.

[32] B. A. Finlayson. *Numerical Methods for Problems with Moving Fronts*. Ravenna Park Publishing Inc., 1992.

[33] C. L. Gardner, J. Glimm, O. McBryan, R. Menikoff, D. H. Sharp, and Q. Zhang. The dynamics of bubble growth for Rayleigh-Taylor unstable interfaces. *Phys. Fluids*, 31:447, 1988.

[34] M. Di Giacinto and M. Valorani. Shock detection and discontinuity tracking for unsteady flows. *Computers and Fluids*, 17(1):61–84, 1989.

[35] H. M. Glaz, P. Colella, I. I. Glass, and R. L. Deschambault. A numerical study of oblique shock-wave reflections with experimental comparisons. *Proc. R. Soc. Lon. A*, 398:117–140, 1985.

[36] H. M. Glaz, P. Colella, I. I. Glass, and R. L. Deschambault. A detailed numerical, graphical, and experimental study of oblique shock wave reflections. *UTIAS Report No. 285*, 1986. University of Toronto.

[37] H. M. Glaz and T.-P. Liu. The asymptotic analysis of wave interactions and numerical calculations of transonic nozzle flow. *Adv. in Appl. Math.*, 4:353–379, 1983.

[38] J. Glimm, J. Grove, B. Lindquist, O. A. McBryan, and G. Tryggvason. The bifurcation of tracked scalar waves. *SIAM J. Sci. Stat. Comput.*, 9(1):61–79, 1988.

[39] J. Glimm, E. Isaacson, D. Marchesin, and O. A. McBryan. Front tracking for hyperbolic systems. *Adv. in Appl. Math.*, 2:91–119, 1981.

[40] J. Glimm, B. Lindquist, O. McBryan, and L. Padmanabhan. A front tracking reservoir simulator, five-spot validation studies and the water coning problem. In *Mathematics of Reservoir simulation*, pages 107–135. SIAM, Philadelphia, 1983. Frontier in Applied Mathematics, Vol. 1.

[41] J. Glimm, D. Marchesin, and O. A. McBryan. Unstable fingers in two phase flow. *Comm. Pure Appl. Math.*, 34:53–75, 1981.

[42] J. Glimm and O. A. McBryan. A computational model for interfaces. *Adv. in Appl. Math.*, 6:422–435, 1985.

[43] J. Glimm, O. A. McBryan, R. Menikoff, and D. H. Sharp. Front tracking applied to Rayleigh-Taylor instability. *SIAM J. Sci. Stat. Comput.*, 7(1):230–251, 1986.

[44] J. Grove. The interaction of shock waves with fluid interfaces. *Adv. in Appl. Math.*, 10:201–227, 1989.

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

[46] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluids with free surfaces. *Phys. Fluids*, 8:2182–2189, 1965.

[47] A. Harten. ENO schemes with subcell resolution. *J. Comput. Phys.*, 83:148–184, 1989.

[48] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high order accurate essentially nonoscillatory scheme III. *J. Comput. Phys.*, 71:148–184, 1987.

[49] A. Harten and J. M. Hyman. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *J. Comput. Phys.*, 50:235–269, 1983.

[50] A. Harten, P. D. Lax, and B. van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25:35–61, 1983.

[51] G. W. Hedstrom. Some numerical experiments with Dafermos's method for nonlinear hyperbolic equations. In *Lecture notes in Mathematics, Vol. 267*. Springer, Berlin, 1972.

[52] F. J. Hickernell and A. B. Yortsos. Linear stability of miscible processes in porous media in the absence of dispersion. *Stud. Appl. Math.*, 74:93–115, 1986.

[53] A. L. Hoffman. A single fluid model for shock formation in MHD shock tubes. *J. Plasma Phys.*, 1:193–207, 1967.

[54] J. M. Hyman. Numerical methods for tracking interfaces. *Physica D*, 12:396–407, 1984.

[55] D. E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley, 2 edition, 1973.

[56] L. D. Landau and E. M. Lifshitz. *Fluid Mechanics*. Pergamon, New York, 1959.

[57] P. D. Lax. Hyperbolic systems of conservation laws and the mathematical theory of shock waves. In *SIAM Regional Conference Series in Applied Mathematics, no. 11*, 1973.

[58] P. Lötstedt. A front tracking method applied to Burger's equation and two-phase porous flow. *J. Comput. Phys.*, 47:211–228, 1982.

[59] R. J. LeVeque. Convergence of a large time step generalization of Godunov's method for conservation laws. *Comm. Pure Appl. Math.*, 37:463–477, 1984.

[60] R. J. LeVeque. A large time step generalization of Godunov's method for systems of conservation laws. *SIAM J. Numer. Anal.*, 22(6):1051–1073, 1985.

[61] R. J. LeVeque. High resolution finite volume methods on arbitrary grids via wave propagation. *J. Comput. Phys.*, 78:36–63, 1988.

[62] R. J. LeVeque. Hyperbolic conservation laws and numerical methods. In *Von Karman Institute for Fluid Dynamics*, 1990. Lecture series on computational fluid dynamics 1990-03.

[63] R. J. LeVeque. *Numerical Methods for Conservation Laws.* Birkhäuser-Verlag, 1990.

[64] R. J. LeVeque. The accuracy of conservative methods for the advection equation on nonuniform grids. *in preparation*, 1991.

[65] R. J. LeVeque. Simplified multi-dimensional flux limiter methods. *Proc. ICFD Conference on Numerical Methods in Fluid Dynamics, Reading*, 1992.

[66] R. J. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *Technical report, Department of Applied Mathematics, University of Washington*, 1993.

[67] R. J. LeVeque and H. C. Yee. A study of numerical methods for hyperbolic conservation laws with stiff source terms. *J. Comput. Phys.*, 86:187–210, 1990.

[68] D. W. Levy, K. G. Powell, and B. van Leer. Use of a rotated Riemann solver for the two-dimensional Euler equations. *Submitted to J. Comput. Phys.*, 1991.

[69] H. W. Liepmann and A. Roshko. *Elements of Gas Dynamics.* John Wiley, New York, 1956.

[70] D.-K. Mao. A treatment of discontinuities in shock-capturing finite difference methods. *J. Comput. Phys.*, 92:422–455, 1991.

[71] D.-K. Mao. A treatment of discontinuities for finite difference methods in the two dimensional case. *J. Comput. Phys.*, 104:377–397, 1993.

[72] O. A. McBryan. Shock tracking methods in 2d flows. In J. J. H. Miller, editor, *computational and asymptotic methods for boundary and interior layers*, pages 68–74. Boole press, 1982.

[73] G. Moretti. A technique for integrating two-dimensional Euler equations. *Computers and Fluids*, 15:59–75, 1987.

[74] W. Mulder, S. Osher, and J. A. Sethian. Computing interface motion in compressible gas dynamics. *J. Comput. Phys.*, 100:209–228, 1992.

[75] W. F. Noh. A time-dependent, two-space-dimensional coupled Eulerian-Lagrangian code. *Methods in Comput. Phys.*, 1964.

[76] W. F. Noh and P. Woodward. SLIC (simple line interface calculation). In A. I. van de Vooren and P. J. Zandbergen, editors, *Proc. 5th Intl. Conf. on Numer. Meth. in Fluid Dynamics*. Springer-Verlag, 1976.

[77] E. S. Oran and J. P. Boris. *Numerical Simulation of Reactive Flow*. Elsevier, New York, 1987.

[78] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithm based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79:12–49, 1988.

[79] D. W. Peaceman. *Fundamentals of Numerical Reservoir Simulation*. Elsevier, 1977.

[80] P. Pelcé. *Dynamics of Curved Fronts*. Academic Press, 1988.

[81] C. S. Peskin. Numerical analysis of blood flow in the heart. *J. Comput. Phys.*, 25:220–252, 1977.

[82] C. S. Peskin and B. F. Printz. Improved volume conservation in the computation of flows with immersed elastic boundaries. *J. Comput. Phys.*, 105:33–46, 1993.

[83] J. M. Picone and J. Boris. Vorticity generation by shock propagation through bubbles in gas. *J. Fluid Mech.*, 189:23–51, 1988.

[84] B. Plohr, J. Glimm, and O. A. McBryan. Application of front tracking to two-dimensional gas dynamics calculations. In J. Chandra and J. Flaherty, editors, *Lecture Notes in Engineering, Vol. 3*, pages 180–191. Springer-Verlag, New York, 1983.

[85] M. J. D. Powell. *Approximation theory and methods*. Cambridge University Press, 1981.

[86] N. H. Risebro and A. Tveito. Front tracking applied to a nonstrictly hyperbolic system of conservation laws. *SIAM J. Sci. Stat. Comput.*, 12:1401–1419, 1991.

[87] N. H. Risebro and A. Tveito. A front tracking method for conservation laws in one dimension. *J. Comput. Phys.*, 101:130–139, 1992.

[88] P. L. Roe. Approximate Riemann solvers, parameter vector, and difference scheme. *J. Comput. Phys.*, 43:357–372, 1981.

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

[90] P. L. Roe. Upwind differencing schemes for hyperbolic conservation laws with source terms. In D. Serre C. Carasso, P. A. Raviart, editor, *Proc. Nonlinear Hyperbolic Problems*, 1986. Springer Lecture Notes in Mathematics 1270.

[91] M. D. Salas. Shock fitting method for complicated two-dimensional supersonic flows. *AIAA Journal*, 14(5):583–588, 1976.

[92] A. Scheidegger. *The Physics of Flow through Porous Media.* University of Toronto press, Toronto, Canada, 1974.

[93] D. H. Sharp. An overview of Rayleigh-Taylor instability. *Physica D*, 12:3–18, 1984.

[94] B. K. Shivamoggi. *Theoretical Fluid Dynamics.* Martinus Nijhoff, 1985.

[95] K.-M. Shyue. *Quasi one-dimensional gas flow.* Master thesis, National Tsing Hua University, Taiwan, R.O.C., June, 1986. unpublished.

[96] G. A. Sod. A numerical study of a converging cylindrical shock. *J. Fluid Mech.*, 83:785–794, 1977.

[97] G. Strang. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.*, 5:506–517, 1968.

[98] B. K. Swartz and B. Wendroff. AZTEC: a front tracking code based on Godunov's method. *Appl. Numer. Math.*, 2:385–397, 1986.

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

[100] Z.-H. Teng, A. J. Chorin, and T.-P. Liu. The Riemann problem for reacting gas, with applications to transition. *SIAM J. Appl. Math.*, 42(42):964–981, 1982.

[101] C. Tu and C. S. Peskin. Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods. *SIAM J. Sci. Stat. Comput.*, 13(6):1361–1376, 1992.

[102] E. van der Maarel and B. Koren. Spurious, zeroth-order entropy generation along a kinked wall. *Intl. J. Numer. Meth. in Fluids*, 13:1113–1129, 1991.

[103] M. van Dyke. *An Album of Fluid Motion.* The Parabolic Press, Stanford, CA, 1982.

[104] B. van Leer. Towards the ultimate conservative difference scheme V. A second order sequel to Godunov's method. *J. Comput. Phys.*, 32:101–136, 1979.

[105] B. Wendroff. An analysis of front tracking for chromatography. *preprint*, 1991.

[106] B. Wendroff and Jr. A. B. White. Some supraconvergent schemes for hyperbolic equations on irregular grids. In J. Ballmann and R. Jeltsch, editors, *Notes on Numerical Fluid Mechanics, no. 24*, 1988.

[107] H. Westenberger and J. Ballmann. The homogeneous homentropic compression or expansion-A test case for analyzing Sod's operator splitting. In J. Ballmann and R. Jeltsch, editors, *Notes on Numerical Fluid Mechanics, no. 24*, 1988.

[108] G. B. Whitham. *Linear and Nonlinear Waves.* John Wiley & Sons, Inc., New York, 1974.

[109] P. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shock. *J. Comput. Phys.*, 54(1):115–173, 1984.

[110] P. Woodward and P. Colella. The piece-wise parabolic method (PPM) for gas dynamical simulations. *J. Comput. Phys.*, 54:174–201, 1984.

[111] A. B. Yortsos and Y. C. Huang. Linear stability analysis of immiscible displacement. *SPERE*, pages 378–390, 1986.

[112] D. M. Young and R. T. Gregory. *A Survey of Numerical Mathematics.* Dover, Vol. II, 1988.

[113] D. L. Youngs. Numerical simulation of turbulent mixing by Rayleigh-Taylor instability. *Physica D*, 12:32–44, 1984.

[114] S. T. Zalesak. A preliminary comparison of modern shock-capturing schemes: linear advection. In R. Vichnevetsky and R. S. Stepleman, editors, *Advances in Computer Methods for Partial Differential Equations VI*, pages 15–22, 1987.

[115] T. A. Zang, M. Y. Hussaini, and D. M. Bushnell. Numerical computations of turbulence amplification in shock-wave interactions. *AIAA Journal*, 22(1):13–21, 1984.

Appendix A

LINEAR THEORY REVIEW

Here we review the linear stability analysis for the Kelvin-Helmholtz and Rayleigh-Taylor problems, see also [33] and [84]. Results obtained from this analysis were used to initialize the flow in the numerical simulation performed in Sections 9.2.1 and 9.2.2. These results also provide comparison solutions in the regime where the amplitude of the interface is small.

A.1 Kelvin-Helmholtz instability

In Section 9.2.1, we consider the Kelvin-Helmholtz unstable interface problem in which the interface separates two fluids of different tangential velocities. We take constant density ρ_0 and pressure p_0 with zero vertical velocity in the computational domain. Above the interface, we have horizontal velocity $u = u_0$, and below the interface, we have horizontal velocity $u = -u_0$. We introduce a sinusoidal perturbation (9.3) on the interface to trigger the instability.

In the standard linear stability analysis, we study the solution behavior on the short time scale, while the amplitude of the interface is small. In the present case, it is then reasonable to assume that the flow is irrotational away from the interface, and the entropy is constant in the domain. Let Φ denote the velocity potential with $u = \partial\Phi/\partial x$ and $v = \partial\Phi/\partial y$ so that the irrotationality condition $\nabla \times \vec{q} = \nabla \times (\nabla\Phi) = 0$ holds, where $\vec{q} = (u, v)$ is the velocity vector. It is easy to show that the governing equations, in the regions above and below the interface, are the following:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}\left(\rho \frac{\partial \Phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(\rho \frac{\partial \Phi}{\partial y}\right) = 0, \tag{A.1}$$

$$\frac{\partial \Phi}{\partial t} + \frac{1}{2}\left((\frac{\partial \Phi}{\partial x})^2 + (\frac{\partial \Phi}{\partial y})^2\right) + H(\rho) = H(\rho_0) + \frac{1}{2}u_0^2, \tag{A.2}$$

where $H = e + p/\rho$ is the enthalpy (for polytropic gas $H = \frac{\gamma}{\gamma-1}p/\rho$). Note that H is a function of density ρ only. This is due to the fact that the entropy, $s_0 = p_0\rho_0^{-\gamma}$, is constant throughout the domain, and so pressure $p = s_0\rho^\gamma$. In the above system, the first equation is the conservation of mass, and the second equation is the Bernoulli's equation for compressible flows, see [25] and [84].

For the boundary conditions, we have solid walls on the top and bottom, and periodic boundaries on the left and right. On the interface, we have the kinematic boundary condition which states that particles at the interface should remain at the interface, namely, on the interface

$$\nabla\Phi \cdot \vec{n} = \vec{q} \cdot \vec{n} \tag{A.3}$$

where $\vec{n}$ is the direction normal to the interface. Let η be the interface position. After some algebra, (A.3) can be written in the following form

$$\left(\frac{\partial \Phi}{\partial y}\right)_{y=\eta} = \frac{\partial \eta}{\partial t} + \left(\frac{\partial \Phi}{\partial x}\right)_{y=\eta} \frac{\partial \eta}{\partial x}. \tag{A.4}$$

To derive the linearized equations for (A.1) and (A.2), we write the solution as its zeroth order (equilibrium) solution and a first order correction,

$$\begin{aligned} \Phi(x,y,t) &\approx \bar{u}x + \Phi_1(x,y,t) \qquad \Phi_1 \ll \bar{u}x, \\ \rho(x,y,t) &\approx \rho_0 + \rho_1(x,y,t) \qquad \rho_1 \ll \rho_0, \end{aligned}$$

and substitute it to (A.1) and (A.2). Retaining only the first order terms, we obtain

$$(\frac{\partial}{\partial t} + \bar{u}\frac{\partial}{\partial x})\rho_1 + \rho_0(\frac{\partial^2 \Phi_1}{\partial x^2} + \frac{\partial^2 \Phi_1}{\partial y^2}) = 0, \tag{A.5}$$

$$(\frac{\partial}{\partial t} + \bar{u}\frac{\partial}{\partial x})\Phi_1 + H^{'}(\rho_0)\rho_1 = 0, \tag{A.6}$$

where $H^{'}(\rho_0) = c_0^2/\rho_0$, and c_0 is the sound speed. Eliminating ρ_1 in (A.5) and (A.6), we get the wave equation for the velocity potential Φ_1,

$$-(\frac{\partial}{\partial t} + \bar{u}\frac{\partial}{\partial x})^2\Phi_1 + c_0(\frac{\partial^2 \Phi_1}{\partial x^2} + \frac{\partial^2 \Phi_1}{\partial y^2}) = 0. \tag{A.7}$$

Note that $\bar{u} = u_0$ in the region *above* the interface, while $\bar{u} = -u_0$ in the region *below* the interface. To simplify the expressions, we ignore the subscript for the velocity potential.

Since our initial interface (9.3) is perturbed sinusoidally with varying interface position in the y-direction, we can write the solution as

$$\Phi = f(y)\exp(\sigma t + ikx) \tag{A.8}$$

where σ is the growth rate (a real number in this case), and k is the wave number. Substituting (A.8) to (A.7), combining terms, we obtain a second order ordinary differential equation for the magnitude function $f(y)$,

$$f^{''}(y) - (\alpha + i\beta)f(y) = 0 \tag{A.9}$$

where

$$\alpha = (M\delta\sigma)^2 - (M^2 - 1)k^2 \qquad \text{and} \qquad \beta = 2M^2\delta k\sigma$$

with $M = |\bar{u}|/c_0$ (Mach number) and $\delta = 1/\bar{u}$.

Note that with the appropriate $\bar{u}$, (A.9) is valid in both the region above and below the interface. In each region, its solution can be determined explicitly when the solid wall and kinematic (a linearized version) boundary conditions are used, see [84]. Since these solutions are rather complicated in expression, we do not present them in detail here. It should be mentioned that using the dynamic boundary condition on the interface, which states that the pressure is continuous across the interface, we obtain a nonlinear equation for the growth rate σ. This is solved numerically using a root-finding routine. In Figure A.1, we plot the solutions obtained from this linear stability analysis with two different Mach numbers, $M = 0.2$ and $M = 0.5$.

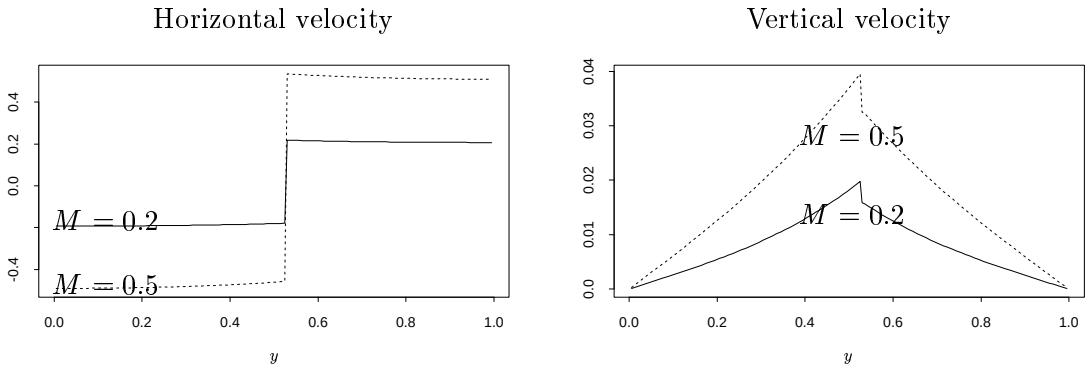


Figure A.1: Typical solutions of the linear theory for the Kelvin-Helmhotlz unstable interface problem. Results with two different Mach numbers, $M = 0.2$ and $M = 0.5$, are shown.

A.2 Rayleigh-Taylor instability

In Section 9.2.2, we consider the Rayleigh-Taylor unstable interface problem in which the interface separates two fluids of different densities. We introduce a small perturbation of an isothermal equilibrium flow with a flat interface separating exponentially stratified flow above and below the interface. Let ρ_0, p_0, and e_0 be the density, pressure, and internal energy of this equilibrium state, respectively. The equations governing these states are

$$\frac{\partial p_0}{\partial y} = \rho_0 g, \tag{A.10}$$

$$\frac{\partial}{\partial y}(e_0 + p_0/\rho_0) = 0, \tag{A.11}$$

see [33] and [108].

As in the Kelvin-Helmholtz problem, we look for solutions in the small amplitude regime, and we write the solutions as

$$\begin{aligned}
\rho(x,y,t) &\approx \rho_0(y) + \rho_1(x,y,t),\\
u(x,y,t) &\approx u_1(x,y,t),\\
v(x,y,t) &\approx v_1(x,y,t),\\
p(x,y,t) &\approx p_0(y) + p_1(x,y,t).
\end{aligned}$$

Substituting them into the Euler equations with gravitational sources (9.4), we can derive the linearized equations for the first order correction terms. For polytropic gas, they are

$$\frac{\partial \rho_1}{\partial t} + \rho_0 \frac{\partial u_1}{\partial x} + \rho_0 \frac{\partial v_1}{\partial y} + \frac{\partial \rho_0}{\partial y} v_1 = 0, \tag{A.12}$$

$$\rho_0 \frac{\partial u_1}{\partial t} + \frac{\partial p_1}{\partial x} = 0, \tag{A.13}$$

$$\rho_0 \frac{\partial v_1}{\partial t} + \frac{\partial p_1}{\partial y} = g\rho_1, \tag{A.14}$$

$$\left(\frac{-c_0^2}{\gamma - 1}\right) \frac{\partial \rho_1}{\partial t} + \left(\frac{1}{\gamma - 1}\right) \frac{\partial p_1}{\partial t} = \rho_0 g v_1. \tag{A.15}$$

Assume that the solutions of these equations take the form,

$$\begin{pmatrix} \rho_1 \\ u_1 \\ v_1 \\ p_1 \end{pmatrix} = \exp(\sigma t + ikx) \begin{pmatrix} \hat{\rho}_1 \\ \hat{u}_1 \\ \hat{v}_1 \\ \hat{p}_1 \end{pmatrix} (y). \tag{A.16}$$

Substituting (A.16) to (A.12)–(A.15), we obtain equations governing the magnitude functions $\hat{\rho}_1$, $\hat{u}_1$, $\hat{v}_1$, and $\hat{p}_1$:

$$\sigma\hat{\rho}_1 + ik\rho_0\hat{u}_1 + \rho_0\frac{\partial\hat{v}_1}{\partial y} + \frac{\partial\rho_0}{\partial y}\hat{v}_1 = 0, \tag{A.17}$$

$$\sigma\rho_0\hat{u}_1 + ik\hat{p}_1 = 0, \tag{A.18}$$

$$\sigma\rho_0\hat{v}_1 + \frac{\partial\hat{p}_1}{\partial y} = g\hat{\rho}_1, \tag{A.19}$$

$$\left(\frac{-c_0^2\sigma}{\gamma-1}\right)\hat{\rho}_1 + \left(\frac{\sigma}{\gamma-1}\right)\hat{p}_1 = \rho_0 g\hat{v}_1. \tag{A.20}$$

We now eliminate $\hat{u}_1$ and $\hat{v}_1$ from the above equations, which yields the following two equations,

$$\sigma^2\hat{\rho}_1 + g\frac{\partial\hat{\rho}_1}{\partial y} + k^2\hat{p}_1 - \frac{\partial^2\hat{p}_1}{\partial y^2} = 0,$$

$$-(c_0^2\sigma^2 + (\gamma-1)g^2)\hat{\rho}_1 + \sigma^2\hat{p}_1 + (\gamma-1)g\frac{\partial\hat{p}_1}{\partial y} = 0.$$

Finally, we eliminate $\hat{\rho}_1$ from the above two equations, and obtain a second order ordinary differential equation for $\hat{p}_1$,

$$\frac{\partial^2\hat{p}_1}{\partial y^2} - \frac{\gamma g}{c_0^2}\frac{\partial\hat{p}_1}{\partial y} - \left(\frac{\sigma^2}{c_0^2} + k^2 + \frac{(\gamma-1)g^2k^2}{c_0^2\sigma^2}\right)\hat{p}_1 = 0. \tag{A.21}$$

This can be solved explicitly with the appropriate boundary conditions, such as solid walls on the top and bottom, and kinematic boundary on the interface, see [33].

We assume that the position of the perturbed interface has the form

$$\eta = y_0 + \varepsilon\exp(\sigma t + ikx),$$

where y_0 is the unperturbed position, and ε is the amplitude of the perturbation. It is easy to check that the solution is

$$\hat{p}_1 = \frac{\rho_0\varepsilon(g^2(\gamma-1) + c_0^2\sigma^2)}{\exp(\alpha_-(y_{bdry}-y_0)) - \exp(\alpha_+(y_{bdry}-y_0))} \times \tag{A.22}$$

$$\left(\frac{\exp(-\alpha_+(y-y_{bdry}))}{(\gamma-1)g + \alpha_- c_0^2} - \frac{\exp(-\alpha_-(y-y_{bdry}))}{(\gamma-1)g + \alpha_+ c_0^2}\right), \tag{A.23}$$

where y_{bdry} is the location of the solid wall, and

$$\alpha_\pm = -\frac{\gamma g}{2c_0^2} \pm \left(\frac{\gamma^2 g^2}{4c_0^4} + \frac{\sigma^2}{c_0^2} + k^2 + \frac{(\gamma-1)g^2k^2}{c_0^2\sigma^2}\right)^{1/2}.$$

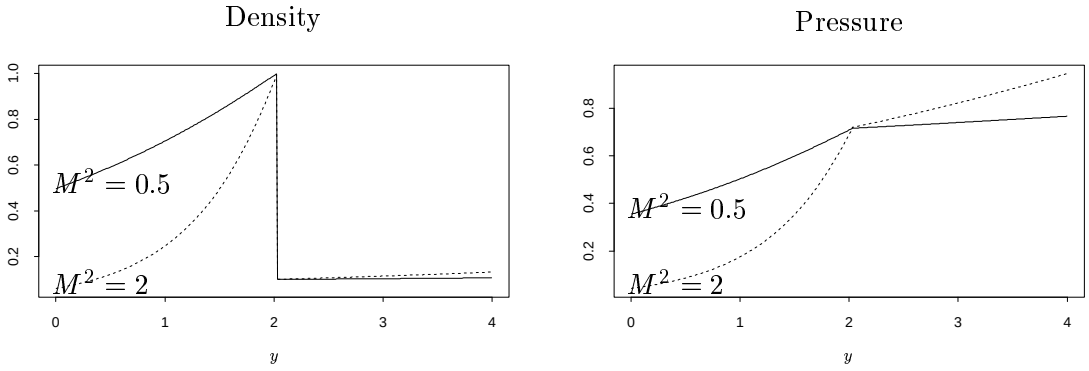


Figure A.2: Typical solutions of the linear theory for the Rayleigh-Taylor unstable interface problem. Results with two different dimensionless parameters, $M^2 = 0.5$ and $M^2 = 2$, are shown. This parameter measures the compressibility of the fluids.

Having obtained the $\hat{p}_1$, other variables can also be determined,

$$\hat{\rho}_1 = \frac{1}{c_0^2\sigma^2 + (\gamma-1)g^2}\left(\sigma^2\hat{p}_1 + (\gamma-1)g\frac{\partial \hat{p}_1}{\partial y}\right), \tag{A.24}$$

$$\hat{u}_1 = -\frac{ik}{\sigma\rho_0}\hat{p}_1, \tag{A.25}$$

$$\hat{v}_1 = \frac{1}{\sigma\rho_0}(g\hat{\rho}_1 - \frac{\partial \hat{p}_1}{\partial y}). \tag{A.26}$$

The requirement on the continuity of the pressure at the interface leads to a nonlinear equation,

$$(\hat{p}_1 + \rho_0 g\varepsilon)_{y\uparrow y_0} = (\hat{p}_1 + \rho_0 g\varepsilon)_{y\downarrow y_0}$$

that determines the growth rate σ.

In Figure A.2, we plot the solutions obtained from this linear stability analysis with two different dimensionless parameters, $M^2 = 0.5$ and $M^2 = 2$, where $M^2 = (2\pi g)/(kc_0^2)$. This parameter measures the compressibility of the fluids.