Wai-How Hui

Kun Xu

Computational Fluid Dynamics Based on the Unified Coordinates

Wai-How Hui
Kun Xu

Computational Fluid Dynamics Based on the Unified Coordinates

Wai-How Hui
Kun Xu

Computational Fluid Dynamics Based on the Unified Coordinates

With 72 figures

Authors

Wai-How Hui
Professor Emeritus
Mathematics Department
Hong Kong University of Science and
Technology
E-mail: whhui@ust.hk

Kun Xu
Professor
Mathematics Department
Hong Kong University of Science and
Technology
E-mail: makxu@ust.hk

Preface

Computational fluid dynamics (CFD) uses large scale numerical computation to solve problems of fluid flow. It has been known since its onset that the numerical solution to a given flow depends on the relation between the flow and the coordinates (mesh) used to compute it. Each of the two well-known coordinate systems for describing fluid flow—Eulerian and Lagrangian—has advantages as well as drawbacks. Eulerian method is relatively simple, but its drawbacks are: ① it smears contact discontinuities badly; ② it needs generating a body-fitted mesh prior to computing flow past a body. Lagrangian method, by contrast, resolves contact discontinuities (including material interfaces and free surfaces) sharply, but it also has drawbacks: ① the gas dynamics equations could not be written in conservation partial differential equations (PDE) form, rendering numerical computation complicated; ② it breaks down due to cell deformation.

A fundamental issue in CFD is, therefore, the role of coordinates and, in particular, the search for "optimal" coordinates. It is in the long search for an optimal coordinate system that a unified coordinate (UC) system was developed by the first author and his collaborators over the last decade. While the search for an optimal coordinate system in CFD would undoubtedly continue, the unified coordinate system developed so far is found to combine the advantages of both Eulerian and Lagrangian system, while avoiding their drawbacks. Indeed, it goes beyond these. For instance, the UC system provides a foundation for automatic mesh generation by the flow being computed.

This monograph first reviews the relative advantages and drawbacks of Eulerian and Lagrangian coordinates as well as the Arbitrary-Lagrangian-Eulerian (ALE) and various moving mesh methods in CFD for one- and multi-dimensional flow. It then systematically introduces the unified coordinate approach to CFD, illustrated with numerous examples and comparisons to clarify its relation with existing approaches.

The content of this monograph is based on a graduate course taught by the first

author from 2000 to 2007 at the Hong Kong University of Science and Technology, Academia Sinica in Taiwan, Hong Kong Polytechnic University and Hong Kong Baptist University, and by the second author since 2009. We thank Prof. T. Tang for his comments on the first draft of the book. We also acknowledge the permission of Communication in Computational Physics (CiCP) for allowing us to use the material presented in a review paper[①].

Many scientists have made substantial contributions in the course of development of the UC approach to CFD. Here is a partial list: Chien-Cheng Chang, De-Lin Chu, Bo Gao, Yuan-Ping He, Jeu-Jiun Hu, Changqiu Jin, Sergei Kudriakov, Chih-Yu Kuo, Claude Lepage, Zuo-Wu Li, Ping-Yiu Li, Meng-Sing Liou, Ching Yuen Loh, Yang-Yao Niu, Keh-Ming Shyue, Ronald Ming Cho So, Yih-Chin Tai, Henry Van Roessel, Zi-Niu Wu, Jaw-Yen Yang, Gui-Ping Zhao, Yanchun Zhao. Without their valuable contributions, the UC approach to CFD could not have reached its current state of maturity. We also thank our secretary Odissa Wong for her help for many years in editing and preparing the figures. We give special thanks to our wives, Kwok Lan Hui and Jie Shen, for their strong support to us in writing this monograph.

<div align="right">

Wai-How Hui
Kun Xu
Hong Kong University of Science and Technology
December 1, 2011

</div>

---

① The unified coordinate system in computational fluid dynamics. *Communications in Computational Physics*, 2: 577-610, 2007.

Contents

Chapter 1
Introduction

1.1 CFD as Numerical Solution to Nonlinear Hyperbolic PDEs

The great majority of research work in CFD, especially those in the first several decades, treats it as numerical solution to nonlinear hyperbolic partial differential equations (PDEs). For a good summary, see Hirsch[1]. Most part of this monograph also treats CFD as numerical solution to nonlinear hyperbolic PDEs. But it is concerned mainly about the role of coordinates in CFD and, in particular, will base all CFD study on the newly discovered unified coordinates. To put it in perspective we shall first give an overview of the major developments of CFD as numerical solution to the initial value problem of nonlinear hyperbolic PDEs as follows.

The theoretical foundation for nonlinear hyperbolic PDEs was laid by Riemann in his pioneering work[2] where he introduced the concept of Riemann invariants and posed the special initial value problem—since has been known as the Riemann problem. It turns out that the Riemann problem plays a central role in most numerical methods in CFD.

Nothing very significant happened during the following six decades until Richardson proposed weather prediction by numerical process (Lewis Fry Richardson, Cambridge University Press, 1922). Even without an electronic computer, wanting to find numerical solutions to nonlinear hyperbolic PDEs immediately raises many interesting theoretical and practical questions, and progresses are made in answering them.

(1) The first of these is the discovery of the CFL condition[3]. It simply says that in a time-marching process to find a numerical solution, marching too fast causes numerical instability and destroys the solution.

(2) Practical methods for computing solutions with shock discontinuities are developed: the artificial viscosity method of von Neumann and Richtmyer which smears shock discontinuities[4]; the Godunov method which reduces the general initial value problem to a sequence of Riemann problems with cell-averaging data[5]; the Glimm random choice method which also reduces the general initial value problem to a sequence of Riemann problems but with data of randomly chosen representative states[6, 7]; and the shock-fitting (front tracking) method[8]. The last two methods are not easily extended to the three-dimensional flow.

(3) A very important discovery was made by Lax and Wendroff[9] that in order to numerically capture shock discontinuities correctly, the governing PDE should be written in conservation form to begin with. This is easily done in Eulerian coordinates (in any dimensions) and also for one-dimensional flow in Lagrangian coordinates. But for a long time, it was not known how to use Lagrangian coordinates to write the governing PDEs for multidimensional flows in conservation form. This problem was solved by Hui et al.[10].

(4) To extend Godunov's method to higher order accuracy, the important concepts of limiters and TVD were introduced which avoid non-physical oscillations in high resolution schemes[11, 12].

(5) From the onset of CFD, it was known that the numerical solution to a given flow depends on the coordinates (mesh) used to compute it; hence great efforts have been devoted to search for the optimum coordinate system: the Particle-in-Cell method[13]; the Arbitrary-Lagrangian-Eulerian method[14]; various moving mesh methods[15]; and the unified coordinate method[10].

(6) Finally, to compute a flow past a body, which is the central problem in fluid dynamics, it is necessary to construct a body-fitted mesh prior to computing the flow. Even after decades of research, mesh-generation remains tedious and time-consuming. The unified coordinate approach to CFD has opened up a way of automatic mesh-generation[16].

1.2   Role of Coordinates in CFD

1.2.1   Theoretical Issues

For more than 200 years, two famous coordinate systems have existed for describing fluid flow: Eulerian system is fixed in space, whereas Lagrangian system follows

the fluid. An immediate question arises:

   "*Are these two coordinate systems equivalent to each other theoretically?*"

This question must have been asked by numerous researchers in fluid dynamics (FD), and the answer presumably was positive. Surprisingly, the first mathematical proof of equivalency, meaning the existence of a one-to-one map between the two sets of weak solutions obtained by using the two systems, was given as late as 1987 by Wagner[17] and holds only for one-dimensional flow[①]. For 2-D and 3-D flows, Hui et al.[10, 18] showed that they are *not equivalent* to each other theoretically (see Section 8.3).

Although Lagrangian and Eulerian approaches to FD are each self-contained and general, prior to the advent of computer most text books [19−22] are written in Eulerian coordinates, with the exception of [23], which is devoted solely to Lagrangian approach. There are at least two reasons for this historical bias.

Firstly, steady flow is the most important class of flow in application of FD, and Eulerian coordinate system has a clear advantage in describing it: the time variable drops out, reducing the number of independent variables from 4 to 3. This greatly simplifies the mathematics of the governing equations. By contrast, the time variable in Lagrangian coordinates is essential and cannot disappear, so apparently we still need 4 independent variables for three-dimensional flow even when the flow is steady. Of course, one might argue that among the 4 apparent independent variables, there must be a relation expressing the steadiness of the flow. Indeed, such a relation does exist, see Eq.(20) in [24], but it is solution-dependent; hence without knowing the flow solution it is difficult to use the relation to reduce the number of independent variables from 4 to 3. On the other hand, when the flow solution is known there is no need to use that relation. This is the dilemma of Lagrangian approach for steady flow: the governing equations of FD in Lagrangian coordinates do not simplify as Eulerian coordinates do, and steady flow has to be obtained by solving the unsteady flow equations. This dilemma was resolved in [25−27] when the Lagrangian time variable was introduced which played the dual role as time and as a Lagrangian label (see also the variable λ in (9.4) when $h = 1$).

Secondly, in the problem of flow past a body, which is the central problem

---

   ① In the presence of a vacuum, the definition of weak solution for the Lagrangian equations must be strengthened to admit test functions which are discontinuous at the vacuum.

in FD, very often one is interested only in the flow quantities on the body sur-
face, e.g., pressure, temperature, velocity and shear stresses on the airfoil surface.
Eulerian approach naturally and easily produces these quantities. By contrast, in
Lagrangian approach, we need to calculate the motion histories of all fluid particles
and then trace them back to find the flow quantities on the body surface. This is
quite cumbersome.

    With the advent of computer and the birth of computational fluid dynamics, the
advantages and drawbacks of Eulerian and Lagrangian approach need be critically
re-examined from the computational point of view.

1.2.2   Computational Issues

Computationally, Eulerian and Lagrangian systems are not equivalent even for 1-D
flow. Indeed, it has been known since the onset of CFD that the numerical solution
to a given flow depends on the relation between the flow and the coordinates used
to compute it. For 1-D flow, we shall show in Chapter 4 that Lagrangian system
is superior to the Eulerian and, in turn, the UC (i.e., the generalized Lagrangian
plus shock-adaptive Godunov scheme) is superior to both the Lagrangian and the
Eulerian, and is completely satisfactory.

    The situation for 2-D and 3-D flow is more complicated. Each of the two well-
known coordinate systems for describing fluid flow has advantages as well as draw-
backs. Eulerian method is relatively simple, but its drawbacks are: ① it smears
contact discontinuities badly; ② it needs generating a body-fitted mesh prior to
computing flow past a body. Lagrangian method, by contrast, resolves contact
discontinuities (including material interfaces and free surfaces) sharply, but it also
has drawbacks: ① the gas dynamics equations could not be written in conserva-
tion partial differential equations (PDE) form, rendering numerical computation
complicated; ② it breaks down due to cell deformation.

    A fundamental issue in CFD is, therefore, the role of coordinates and, in par-
ticular, the search for "optimal" coordinates. The search for optimal coordinates
has led to the development of the unified coordinate (UC) system[28], in a series
papers beginning with [25−27]. See also [29−33]. This monograph first reviews the
relative advantages and drawbacks of Eulerian and Lagrangian coordinates in CFD
for 1-D and multi-dimensional flow, and then systematically discusses the unified

coordinate system and its applications.

For 1-D flow, UC uses a material coordinate and also applies the shock-adaptive Godunov scheme [34−36] instead of the classical Godunov scheme[5]. For 2-D flow, it uses one material coordinate, with the other coordinate determined so as to preserve mesh othorgonality (or preserve the Jacobian), whereas for 3-D flow, it uses two material coordinates, with the third one determined so as to preserve mesh skewness (or preserve the Jacobian). The unified coordinate system may be regarded as a generalization of Lagrangian system. It combines the advantages of both Eulerian and Lagrangian system while avoiding their drawbacks. The UC formulation also provides a foundation for automatic mesh-generation by the flow being computed. It may also be regarded as a moving mesh method in that the mesh can move in any manner, while the effects of its movement on the flow are fully accounted for.

1.3   Outline of the Book

This book is arranged as follows: Derivation of the equations of physical conservation laws are given in Chapter 2. Chapter 3 reviews shock-capturing methods for 1-D flow based on Eulerian coordinates, pointing out their defects. Although most of the materials in Chapters 2 and 3 can be found in existing texts, e.g., Ref. [1, 37−39], they are included here to give a smooth introduction to the main theme of this monograph and also to make it self-contained. Chapter 4 introduces UC method for 1-D flow and shows how all defects of Eulerian and Lagrangian computation are cured or avoided by UC. Chapter 5 comments on the difficulties encountered in current computational methods for the general case of multidimensional unsteady flow. Chapter 6 gives the unified coordinates formulation of CFD for multi-dimensional unsteady flow, whose mathematical properties are studied in Chapter 7. Chapter 8 is devoted to the very important special case of Lagrangian gas dynamics. Chapter 9 uses UC to study the simpler problem of steady 2-D supersonic flow, showing that it can be solved essentially as 1-D unsteady flow. 3-D steady supersonic flow is also discussed. Chapter 10 discusses the general case of unsteady flow computation using UC, illustrated with typical examples and comparisons with existing methods. Chapter 11 discusses viscous flow computation. Chapter 12 is devoted to the applications of the unified coordinates

to kinetic theory. Finally, a summary of the book is given in Chapter 13.

References

[1] C. Hirsch. *Numerical Computation of Internal and External Flows, Vol. II: Computational Methods for Inviscid and Viscous Flows.* Wiley, 1990.

[2] G.F.B.Riemann. *Ueber die Fortpflanzung ebener Luftwellen von endlicher Schwingungsweite, Abh. König. Gesell. Wiss. Göttingen,* 8, 43, 1860.

[3] R. Courant, K. O. Friedrichs and H. Lewy. Über die Partiellen differenzengleichugen der mathematischen *Physik. Math. Ann.,* 100, 32, 1928.

[4] J. von Neumann and R.D. Richtmyer. A method for the numerical calculation of hydrodynamic shocks. *J. Appl. Phys.,* 21: 232, 1950.

[5] S.K. Godunov. A difference scheme for numerical computation of discontinuous solutions of hydrodynamic equations. *Math. Sbornik,* 47:271, 1959.

[6] J. Glimm. Solution in the large for nonlinear hyperbolic systems of equations. *Comm. Pure. Appl. Math.,* 18:697-715, 1965.

[7] A.J. Chorin. Random Choice Solutions of Hyperbolic Systems. *J. Comput. Phys.,* 22:517-533, 1976.

[8] G. Moretti. Thoughts and Afterthoughts About Shock Computations. *Polytechnic Institute of Brooklyn PIBAL Report,* 72, 1972.

[9] P.D. Lax and B. Wendroff. Syetems of conservation laws. *Comm. Pure and Applied Mathematics,* 13:217-237, 1960.

[10] W.H. Hui, P.Y. Li and Z.W. Li. A unified coordinate system for solving the two-dimensional Euler equations. *J. Comput. Phys.,* 153:596-637, 1999.

[11] J.P. Boris and D.L. Book. Flux-corrected transport. I.SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.,* 11:38-69, 1973.

[12] B. van Leer. Towards the ultimate conservative difference scheme IV, a new approach to numerical convection. *J. Comput. Phys.,* 23: 276-299, 1977.

[13] F.H. Harlow. LAMS-1956, Los Alamos Scientic Laboratory Report, 1955.

[14] C. W. Hirt, A. A. Amsden and J. L. Cook. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *J. Comput. Phys.,* 14:227-253, 1974.

[15] J. U. Brackbill and J. S. Saltzman. Adaptive zoning for singular problems in two dimensions. *J. Comput. Phys.,* 46, 342, 1982.

[16] W.H. Hui, G.P. Zhao, J.J. Hu and Y. Zheng. Flow-generated-grids – gridless computation using the unified coordinates. in *Proceedings of the 9th International Conference on Numerical Grid Generation,* 111-121, 2005.

[17] D.H. Wagner. Equivalence of Euler and Lagrangian equations of gas dynamics for weak solutions. *J. Differ. Equations,* 68:118-136, 1987.

[18] W.H. HUI AND S. KUDRIAKOV. A unified coordinate system for solving the three-dimensional Euler equations. *J. Comput. Phys.*, 172:235-260, 2001.

[19] H. LAMB. *Hydrodynamics.* 6th Ed., New York: Dover publication, 1991.

[20] G. K. BATCHELOR. *An Introduction to Fluid Dynamics.* Cambridge University Press, 1973.

[21] C.S. YIH. *Fluid Mechanics: A Concise Introduction to the Theory.* Ann Arbor, Michigan, West River Press, 1988.

[22] L. D. LANDAU AND E.M. LIFSHITZ. *Fluid Mechanics.* Addison-Wesley Pub. Co., 1959.

[23] M. SHASHKOV. *Conservative Finite-difference Methods on General Grids.* CRC Press, 1996.

[24] W.H. HUI, Z.N. WU AND B. GAO. Preliminary extension of the unified coordinate approach to computation of viscous flows. *J. Sci. Comput.*, 30:301-344, 2007.

[25] W. H. HUI AND C.Y. LOH. A new Lagrangian method for steady supersonic flow computation, Part II: Slipline resolution. *J. Comput. Phys.*, 103:450-464, 1992.

[26] W.H. HUI AND C.Y. LOH. A new Lagrangian method for steady supersonic flow computation, Part III: Strong shocks. *J. Comput. Phys.*, 103:465-471, 1992.

[27] C.Y. LOH AND W.H. HUI. A new Lagrangian method for steady supersonic flow computation Part I: Godunov scheme. *J. Comput. Phys.*, 89:207-240, 1990.

[28] W.H. HUI. The unified coordinate system in computational fluid dynamics. *Communications in Computational Physics*, 2:577-610, 2007.

[29] Z. N. WU. A note on the unified coordinate system for computing shock waves. *J. Comput. Phys.*, 180:110-119, 2002.

[30] R. M. C. SO, Y. LIU AND Y. G. LAI. Mesh shape preservation for flow-induced vibration problems. *Journal of Fluids and Structures*, 18:287-304, 2003.

[31] W. H. HUI AND G. P. ZHAO. Capturing contact discontinuities using the unified coordinates, in Proceedings of the 2nd MIT Conference on Computational Fluid and Solid Mechanics (Ed: K J Bathe), Volume 2:2000-2005, 2003.

[32] Y. C. TAI AND C. Y. KUO. A new model of granular flows over general topography with erosion and deposition. *Acta Mech.*, 199:71-96, 2008.

[33] Y. Y. NIU, Y. H. LIN, W. H. HUI AND C. C. CHANG. Development of a moving artificial compressibility solver on unified coordinates. 65:1029-1052, 2009.

[34] W.H. HUI AND S. KUDRIAKOV. The role of coordinates in the computation of discontinuities in one-dimensional flow. *Computational Fluid Dynamics Journal*, 8:495-510, 2000.

[35] W.H. HUI AND S. KUDRIAKOV. On wall overheating and other computational difficulties of shock-capturing methods. *Computational Fluid Dynamics J.*, 10:192-209, 2001.

[36] C.Y. LEPAGE AND W.H. HUI. A shock-adaptive Godunov scheme based on the generalized Lagrangian formulation. *J. Comput. Phys.*, 122:291-299, 1995.

[37] R.J. LEVEQUE. Nonlinear conservation laws and finite volume methods for astrophysical fluid flow. *Computational Methods for Astrophysical Flow*, Edited by O. Steiner and A. Gautschy. Springer-Verlag, 1998.

[38] E. GODLEWSKI AND P.A. RAVIART. *Numerical Approximation of Hyperbolic Systems of Conservation Laws.* New York, Springer-Verlag, 1996.

[39] E.F. TORO. *Riemann Solvers and Numerical Methods for Fluid Dynamics.* Springer, 1999.

Chapter 2

Derivation of Conservation Law Equations

All fluid motions, no matter how complex, are governed by the conservation laws of physics, namely, conservation of mass, momentum and energy. This chapter is devoted to the derivation of these conservation law equations and discussing their mathematical properties.

2.1 Fluid as a Continuum

Fluids, like all matter, are made up of molecules. The properties of a fluid in motion may be studied on the basis of the mechanics of the molecules composing the fluid. Although such a procedure may appear feasible in principle, it will indeed be a formidable task to achieve solutions to practical problems. Apart from this consideration, we are generally not interested in the details of motion of the molecules. What we intend to do in fluid dynamics is to establish relations between various macroscopically observable properties, e.g., pressure, density and velocity, pertaining to a fluid in motion. These macroscopically observable quantities are the mean values, in space and time, obtained by taking the average over a sufficiently long time compared to the time scale related to the motion of the molecules. For example, the mass density ρ in a small volume Ω_0 is taken as

$$\rho = \lim_{\Omega \to \Omega_0} \frac{\text{number of molecules} \times \text{mass of molecules in } \Omega}{\Omega}, \qquad (2.1)$$

where Ω_0 is macroscopically small so that the value of ρ as defined in (2.1) represents how dense the mass is locally, but is microscopically large so that it represents a mean density free of molecular fluctuations. Such a definition is possible as, for instance, at normal pressure and temperature, a volume of $(10^{-4}\text{cm})^3$ of air contains 2.7×10^7 of molecules. This being the case, it is a reasonable assumption

to regard a fluid as a continuous medium or a continuum. With such a picture of the fluid in mind, we call an infinitesimal fluid element a fluid particle, which in fact contains millions of molecules.

The continuum hypothesis breaks down in some extreme situations, e.g., space flight at high altitude where the air is rarefied. In this situation, we invoke the kinetic theory instead (see Chapter 12).

2.2 Derivation of Conservation Law Equations in Fixed Coordinates

Let's consider a control volume Ω of fluid which is fixed in space (Figure 2.1).
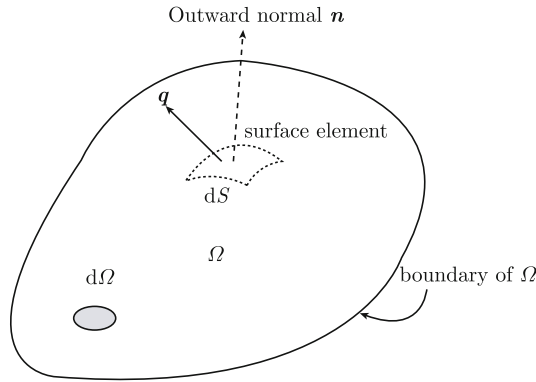
Figure 2.1   Control volume showing notation

(1) Conservation of mass. For conservation of mass, we have

rate of increase of mass in Ω=mass influx through $\partial\Omega$ per unit time.

Mathematically,

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega}\rho\mathrm{d}\Omega = -\int_{\partial\Omega}\rho\boldsymbol{q}\cdot\boldsymbol{n}\mathrm{d}S. \qquad (2.2)$$

Here t is time, ρ and $\boldsymbol{q}$ are the density and velocity of the fluid, and the right hand side is the inward mass flux across the surface of Ω per unit time[①].

(2) Conservation of momentum. In a similar fashion, the equation for the rate of increase of momentum in Ω is

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega}\rho\boldsymbol{q}\mathrm{d}\Omega = -\int_{\partial\Omega}\boldsymbol{q}\rho\boldsymbol{q}\cdot\boldsymbol{n}\mathrm{d}S + \int_{\partial\Omega}\boldsymbol{\tau}\mathrm{d}S + \int_{\Omega}\rho\boldsymbol{F}\mathrm{d}\Omega, \qquad (2.3)$$

---

① When chemical reaction takes place, the conservation of mass must be applied to each species separately.

where the left hand side is the rate of increase of momentum inside Ω, while on the right hand side, the first term is the rate of inward flux of momentum carried across $\partial\Omega$ by the flow, the second term is the total surface force produced by the stress $\boldsymbol{\tau}$ on $\partial\Omega$, and the third term is the total body force on Ω, where $\boldsymbol{F}$ denotes the body force per unit mass, e.g., the gravitation force.

(3) Conservation of energy. For the conservation of energy, we have

rate of increase of energy in Ω

$=$ influx of energy per unit time through $\partial\Omega$

$+$work done by (surface and body) forces per unit time

$+$heat transfer due to thermal conduction per unit time.

Here we neglect the heat transfer due to radiation, as it is very small in most cases, except when the temperature is extremely high.

Now the total energy per unit mass is equal to $\boldsymbol{q}\cdot\boldsymbol{q}/2+i$, where i is the specific internal energy per unit mass. Mathematically, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega}\left(\frac{1}{2}\boldsymbol{q}\cdot\boldsymbol{q}+i\right)\rho\mathrm{d}\Omega=-\int_{\partial\Omega}\left(\frac{1}{2}\boldsymbol{q}\cdot\boldsymbol{q}+i\right)\rho\boldsymbol{q}\cdot\boldsymbol{n}\mathrm{d}S$$
$$+\int_{\Omega}\rho\boldsymbol{F}\cdot\boldsymbol{q}\mathrm{d}\Omega+\int_{\partial\Omega}\boldsymbol{\tau}\cdot\boldsymbol{q}\mathrm{d}S-\int_{\partial\Omega}\boldsymbol{\kappa}\cdot\boldsymbol{n}\mathrm{d}S, \qquad (2.4)$$

where $\boldsymbol{\kappa}$ is the heat flux per unit area per unit time.

With the body force $\boldsymbol{F}$ given—and from now on the body force will be assumed negligible—this system of five equations (2.2)$\sim$(2.4) (the vector equation (2.3) is equivalent to three scalar equations) contains more than 5 unknowns, namely ρ, $\boldsymbol{q}$, $\boldsymbol{\tau}$, $\boldsymbol{\kappa}$ and i. Hence the system is not closed.

We consider two flow models.

(1) Inviscid flow. For a fluid in hydrostatic equilibrium in the absence of body force, it is known that

(i) the stress on any surface element is normal to the surface and independent of its orientation. Hence

$$\boldsymbol{\tau}=-p\boldsymbol{n}, \qquad (2.5)$$

where $p>0$ is the scalar pressure.

(ii) The heat conduction

$$\boldsymbol{\kappa}=\boldsymbol{0}. \qquad (2.6)$$

(iii) The internal energy is a function of the pressure and density,

$$i = i(p, \rho). \tag{2.7}$$

The form of this function is established by experiment and various thermodynamic arguments.

When the gas is non-uniform or in motion, none of these is strictly true. However, provided that time and space derivatives are not too large, they are still good approximations for many purposes. With these as assumptions, the basic conservation equations become a closed set for the five flow quantities ρ, p and three components of velocity $\boldsymbol{q}$. They are, when written in integral form,

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \rho \mathrm{d}\Omega = - \int_{\partial\Omega} \rho \boldsymbol{q} \cdot \boldsymbol{n} \mathrm{d}S, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \rho q_j \mathrm{d}\Omega = - \int_{\partial\Omega} [\rho q_j \boldsymbol{q} + p\boldsymbol{j}] \cdot \boldsymbol{n} \mathrm{d}S, \quad j = 1, 2, 3, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \rho \left(\frac{1}{2} \boldsymbol{q} \cdot \boldsymbol{q} + i \right) \mathrm{d}\Omega = - \int_{\partial\Omega} \left[\rho \left(\frac{1}{2} \boldsymbol{q} \cdot \boldsymbol{q} + i \right) \boldsymbol{q} + p\boldsymbol{q} \right] \cdot \boldsymbol{n} \mathrm{d}S, \end{cases} \tag{2.8}$$

where the vector $\boldsymbol{j}$ is the unit vector in the $x_j (j = 1, 2, 3)$ direction. The five equations in (2.8) are called *inviscid flow equations*, and the flows they describe are called *inviscid flows*.

(2) Viscous flow. In the general case, we have

(i) Fourier law for heat-conduction:

$$\boldsymbol{\kappa} = -k\nabla T, \tag{2.9}$$

where $k > 0$ is the coefficient of thermo-conductivity of the fluid, and T denotes the temperature.

(ii) Cauchy formula for stress components

$$\tau_i(\boldsymbol{n}) = \tau_{ji} n_j, \quad i, j = 1, 2, 3. \tag{2.10}$$

(iii) Symmetry of stress tensor:

$$\tau_{ij} = \tau_{ji}. \tag{2.11}$$

(iv) For Newtonian fluid,

$$\tau_{ij} = -p\delta_{ij} + \mu \left[\left(\frac{\partial q_i}{\partial x_j} + \frac{\partial q_j}{\partial x_i} \right) - \frac{2}{3} \delta_{ij} \frac{\partial q_k}{\partial x_k} \right], \tag{2.12}$$

where δ_{ij} is the Kronecker delta and μ is the coefficient of viscosity of the fluid. Details of derivation of (i)$\sim$(iv) can be found in most texts on fluid mechanics, e.g., [1] and [2].

Using $(2.9) \sim (2.12)$, we rewrite the conservation laws equations $(2.2)\sim(2.4)$ in their integral forms as follow:

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}\displaystyle\int_\Omega \rho\,\mathrm{d}\Omega + \int_{\partial\Omega}\rho q_j n_j\mathrm{d}S = 0, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t}\displaystyle\int_\Omega \rho q_i\,\mathrm{d}\Omega + \int_{\partial\Omega}(\rho q_i q_j - \tau_{ij})n_j\mathrm{d}S = 0, \quad i = 1,2,3, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t}\displaystyle\int_\Omega \rho\left(\dfrac{q_k q_k}{2}+i\right)\mathrm{d}\Omega + \int_{\partial\Omega}\left[\rho\left(\dfrac{q_k q_k}{2}+i\right)q_j - k\dfrac{\partial T}{\partial x_j} - \tau_{ij}q_i\right]n_j\mathrm{d}S = 0, \end{cases} \quad (2.13)$$

where Einstein's summation convention is used over the range of $j, k = 1, 2, 3$. The five equations in (2.13) are the *viscous flow equations* for Newtonian fluid.

Remark   (1) With the equation of state of the fluid $p = \rho RT$ appended, where R is the gas constant and $i = i(p, \rho)$ is the specific internal energy, the system of equations (2.13) forms a closed system for the five unknowns $\rho, p, q_i(i = 1, 2, 3)$.

(2) Inviscid flow is a special case of viscous flow when the fluid viscosity and thermo-conductivity are negligible, i.e., when $\mu = k = 0$.

(3) Inviscid flow may become discontinuous (in space and in time), even if the initial flow is smooth.

(4) Inviscid flow is reversible, since the inviscid flow equations (2.8) are obviously invariant under the transformation

$$\begin{cases} t \to -t, \\ \boldsymbol{q} \to -\boldsymbol{q}. \end{cases} \quad (2.14)$$

(5) This is not the case for the viscous flow equations (2.13) with $\mu, k > 0$. Consequently, viscous flow is irreversible.

For most part of the monograph, we shall be concerned mainly with inviscid flow, which is much simpler than viscous flow. It is generally recognized that the convective terms in the inviscid flow equations (2.8), i.e., the fluxes across the surface of the control volume, are the terms that pose the most stringent requirements on the discretisation techniques, but that the viscous and thermo-conduction terms in the viscous flow equations (2.13) can be handled relatively easily using central discretisation. Viscous flow computation will be discussed in Chapters 11 and 12.

2.3   Conservation Law Equations in Moving Coordinates

If the coordinate system used to describe fluid flow moves itself with velocity $\boldsymbol{Q}$, the conservation law equations derived in the previous section need be modified by replacing the fluid velocity $\boldsymbol{q}$ by its relative velocity $(\boldsymbol{q} - \boldsymbol{Q})$ in the flux terms. Accordingly, for example, the inviscid flow equations (2.8) become

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{Q}(t))} \rho \mathrm{d}\Omega = - \int_{\partial\Omega} \rho(\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{n} \mathrm{d}S, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{Q}(t))} \rho q_j \mathrm{d}\Omega = - \int_{\partial\Omega} [\rho q_j(\boldsymbol{q} - \boldsymbol{Q}) + p\boldsymbol{j}] \cdot \boldsymbol{n} \mathrm{d}S \\[1mm] \qquad\qquad\qquad\qquad +\text{inertial forces}, \quad j = 1,2,3, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{Q}(t))} \rho e \mathrm{d}\Omega = - \int_{\partial\Omega} [\rho e(\boldsymbol{q} - \boldsymbol{Q}) + p\boldsymbol{q}] \cdot \boldsymbol{n} \mathrm{d}S. \end{cases} \qquad (2.15)$$

Here $e = (1/2)\boldsymbol{q}^2 + i(p, \rho)$ is the specific total energy of the fluid.

In the special case of Lagrangian coordinates, $\boldsymbol{Q} = \boldsymbol{q}$, and (2.15) simplifies to

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{q}(t))} \rho \mathrm{d}\Omega = 0, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{q}(t))} \rho q_j \mathrm{d}\Omega = - \int_{\partial\Omega} p\boldsymbol{j} \cdot \boldsymbol{n} \mathrm{d}S + \text{inertial forces}, \quad j = 1,2,3, \\[2mm] \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_{\Omega(\boldsymbol{q}(t))} \rho e \mathrm{d}\Omega = - \int_{\partial\Omega} p\boldsymbol{q} \cdot \boldsymbol{n} \mathrm{d}S. \end{cases} \qquad (2.16)$$

On the other hand, in the special case of Eulerian coordintes, $\boldsymbol{Q} = \boldsymbol{0}$ and hence Ω is independent of time t, yielding the system (2.8), which may also be written symbolically as

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega} \boldsymbol{E} \mathrm{d}\Omega = - \int_{\partial\Omega} \boldsymbol{F}(\boldsymbol{E}) \cdot \boldsymbol{n} \mathrm{d}S, \qquad (2.17)$$

where $\boldsymbol{E} = (\rho, \rho\boldsymbol{q}, \rho e)^{\mathrm{T}}$.

The issues of computation complications of using conservation law equations (2.15) for moving coordinates, including (2.16) for Lagrangian coordinates, over (2.8) for fixed coordinates (also called Eulerian coordinates) will be discussed in later chapters.

2.4   Integral Equations versus Partial Differential Equations

To illustrate concepts, we use conservation law equations for fixed coordinates. As shown in Section 2.2, conservation laws of physics lead directly to integral form

equations (2.13) (including the inviscid flow equations (2.8) as special case), which thus describe the actual physical phenomena properly; in particular, they admit piece-wise smooth solution. On the further assumption that the flow is smooth everywhere, i.e., p, ρ, i and $\boldsymbol{q}$ have continuous first derivatives, these integral equations can be converted to PDE by noting that for any smooth functions $f(x, y, z, t)$ and $\boldsymbol{F}(x, y, z, t)$, we have

(1)
$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega f(x, y, z, t) \mathrm{d}\Omega = \int_\Omega \frac{\partial f}{\partial t} \mathrm{d}\Omega. \tag{2.18}$$

(2)
$$\int_{\partial\Omega} \boldsymbol{F} \cdot \boldsymbol{n} \mathrm{d}S = \int_\Omega \nabla \cdot \boldsymbol{F} \mathrm{d}\Omega \quad \text{(Gauss Divergence Theorem)}. \tag{2.19}$$

(3)
$$\text{If } \int_\Omega f \, \mathrm{d}\Omega = 0 \quad \text{for all } \Omega \subset \mathbf{R}^3 \text{ and } f \text{ is continuous, then } f \equiv 0. \tag{2.20}$$

Making use of (2.18)$\sim$(2.20), we obtain from (2.13) the differential form of the conservation law equations

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial(\rho q_j)}{\partial x_j} = 0, \\[2mm] \dfrac{\partial(\rho q_i)}{\partial t} + \dfrac{\partial}{\partial x_j}(\rho q_i q_j + p\delta_{ij}) \\[2mm] \quad = \dfrac{\partial}{\partial x_j}\left[\mu\left(\dfrac{\partial q_i}{\partial x_j} + \dfrac{\partial q_j}{\partial x_i}\right) - \dfrac{2}{3}\mu\delta_{ij}\dfrac{\partial q_k}{\partial x_k}\right], \quad i = 1, 2, 3, \\[2mm] \dfrac{\partial}{\partial t}\left[\rho\left(\dfrac{q_k q_k}{2} + i\right)\right] + \dfrac{\partial}{\partial x_j}\left[\rho q_j\left(\dfrac{q_k q_k}{2} + i\right) + p q_j\right] \\[2mm] \quad = \dfrac{\partial}{\partial x_j}\left[\mu q_i\left(\dfrac{\partial q_i}{\partial x_j} + \dfrac{\partial q_j}{\partial x_i}\right) - \dfrac{2}{3}\mu\dfrac{\partial q_k}{\partial x_k}q_j + k\dfrac{\partial T}{\partial x_j}\right]. \end{cases} \tag{2.21}$$

This system of equations is called the Navier-Stokes Equations for viscous flow.

In the special case of inviscid flow for which $k = \mu = 0$, the Navier-Stokes equations reduce to the Euler equations

$$\frac{\partial \boldsymbol{E}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{E}) = \boldsymbol{0}, \tag{2.22}$$

which is said to be in conservation form. Written out in details, (2.22) becomes

$$
\frac{\partial}{\partial t}
\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{pmatrix}
+ \frac{\partial}{\partial x}
\begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ \rho u \left(e + \dfrac{p}{\rho} \right) \end{pmatrix}
+ \frac{\partial}{\partial y}
\begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ \rho v \left(e + \dfrac{p}{\rho} \right) \end{pmatrix}
+ \frac{\partial}{\partial z}
\begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ \rho w \left(e + \dfrac{p}{\rho} \right) \end{pmatrix}
= \mathbf{0},
$$

$$(2.23)$$

where u, v, w are the x-, y- and z-component of the velocity $\boldsymbol{q}$, respectively, and

$$
e = \frac{1}{2}(u^2 + v^2 + w^2) + i.
$$

Note   The Euler equations for inviscid flow (2.23) can also be derived directly from the integral form equations (2.8).

Remarks on the Navier-Stokes equations for viscous flow

(1) It is generally believed, though not yet proven, that all solutions to the Navier-Stokes equations for laminar flow are smooth. If so, the system of integral equations (2.13) for viscous flow is equivalent to the Navier-Stokes equations (2.21) which are in PDE form.

(2) For unsteady flow, the Navier-Stokes equations are hyperbolic if $k \approx 0, \mu \approx 0$; parabolic, otherwise.

(3) It is not known what general initial and boundary conditions will make the Navier-Stokes equations well-posed. The situation is even more uncertain when these equations are solved numerically.

(4) For many practical viscous flows, the fluid viscosity and thermo-conductivity are very small, resulting in a high Reynolds number. For high Reynolds number flow, the viscous scale is several orders of magnitude smaller than a typical length scale of the problem. Without special treatment, large number of mesh points would be needed to get reasonable resolution of the numerical solution, requiring very large storage and long computing time.

Remarks on the Euler equations for inviscid flow

(1) The inviscid flow equations (2.23) obtained by neglecting fluid viscosity and thermo-conductivity in the Navier-Stokes equations are of hyperbolic type and have only one scale. These render numerical computation much easier than the viscous flow equations. This is why great efforts are spent on computing inviscid

flow solution (see [3–5]), hoping to get a good first approximation to high Reynolds number flow.

(2) In smooth flow region, the Euler equations (2.23) hold and are equivalent to its integral counterparts (2.8).

(3) Across discontinuities, only the integral form equations (2.8) are valid.

Therefore, the starting point for studying inviscid flow should be the integral equations (2.8), not the Euler PDE (2.23) as is done quite commonly. In this regard, we note that it is the integral form equations (2.8), not the Euler equations (2.23), that are actually solved in numerical computation (see Section 3.4). The reason why most text books discuss the Euler PDE (2.23) at great length, rather than the integral equations (2.8), is because the mathematical theories of system of hyperbolic PDE are quite well developed, e.g., characteristics and Riemann invariants, but we know very little about the mathematical theory of integral-differential equations (2.8). The mathematical properties revealed by the PDE, (2.23), though only valid for smooth flow, are found to be very helpful in computing piece-wise smooth solution to the integral equations (2.8). See Sections 3.4∼3.7.

2.5   The Entropy Condition for Inviscid Flow Computation

As pointed out in Section 2.2 the inviscid flow model neglects all irreversible mechanisms, such as viscosity and thermo-conductivity of the fluid. Consequently, the solution to the inviscid flow equations (2.8) (or (2.23)) is not unique: once it admits one solution it also simultaneously admits its reverse flow solution. If the conservation law equations (2.8) are to model the real flow which typically has very small but positive viscosity and thermo-conductivity, then clearly only one of these solutions can be physically correct.

In order to pick out the physically correct solution, one general approach is to add a diffusive term so as to obtain a system with a unique smooth solution and then let the coefficient of this term tend to zero. However, this "vanishing viscosity" method requires studying a more complicated system of equations, somewhat similar to the Navier-Stokes equations. But this is what we wanted to avoid in the first place when we try to get a good approximation to viscous flow by solving the inviscid equations because of its simplicity. For this reason, we would like to derive other conditions that can be applied directly on the solutions of this (hyperbolic)

inviscid system to filter out the physically incorrect solutions. For gas dynamics, we can appeal to the second law of thermodynamics, which states that entropy is non-decreasing. This very simple condition, called *entropy condition*, enables us to select the physically correct inviscid solution that well approximates the solution of flow with small viscosity. More will be discussed about entropy condition in Section 3.3.

For other systems, it is frequently possible to derive similar conditions, generally called entropy conditions, by analogy with gas dynamics.

References

[1] G.K. BATCHELOR. *An introduction to fluid dynamics*. Cambridge: Cambridge University Press, 1973.

[2] C.S. YIH. *Fluid Mechanics: A concise introduction to the theory*. Ann Arbor, Michigan, West River Press, 1988.

[3] C. HIRSCH. *Numerical Computation of Internal and External Flows, Vol. II: Computational Methods for Inviscid and Viscous Flows*. New York: Wiley, 1990.

[4] R.J. LEVEQUE. Nonlinear conservation laws and finite volume methods for astrophysical fluid flow. *Computational Methods for Astrophysical Flow*, Edited by O. Steiner and A. Gautschy, New York: Springer-Verlag, 1998.

[5] E.F. TORO. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. New York: Springer-Verlag, 1999.

Chapter 3

Review of Eulerian Computation for 1-D Inviscid Flow

3.1 Flow Discontinuities and Rankine-Hugoniot Conditions

3.1.1 Special Case: Stationary Surface of Discontinuity

Let σ be a stationary surface of discontinuity and $\boldsymbol{n}$ be a unit normal of σ (Figure 3.1). We take a rectangular volume Ω for which σ cuts across Ω as shown in the figure. Let S_+ denote the surface of Ω which lies in the positive side of σ, S_- that lies in the negative side, and S_l denote the lateral surfaces of Ω.
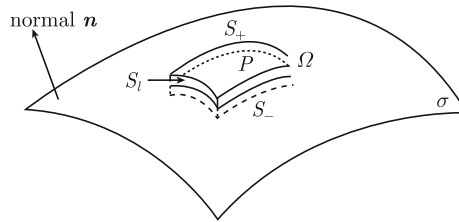
Figure 3.1   Control volume Ω in relation to surface of discontinuity σ

Recall the integral form equations (2.8) for inviscid flow,

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \rho \mathrm{d}\Omega + \displaystyle\int_{\partial\Omega} \rho \boldsymbol{q} \cdot \boldsymbol{n} \mathrm{d}S = 0, \\ \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \rho \boldsymbol{q} \mathrm{d}\Omega + \displaystyle\int_{\partial\Omega} (\rho \boldsymbol{q}(\boldsymbol{q}\cdot\boldsymbol{n}) + p\boldsymbol{n}) \mathrm{d}S = 0, \\ \dfrac{\mathrm{d}}{\mathrm{d}t} \displaystyle\int_\Omega \left(\dfrac{1}{2}q^2 + i\right)\rho \mathrm{d}\Omega + \displaystyle\int_{\partial\Omega} \left(\dfrac{1}{2}q^2 + i + \dfrac{p}{\rho}\right)\rho \boldsymbol{q} \cdot \boldsymbol{n} \mathrm{d}S = 0, \end{cases} \tag{3.1}$$

where $\boldsymbol{n}$ is the outward unit normal of $\partial\Omega$. We restrict, for the moment, our attention to the case when σ is stationary and apply the following identity to (3.1):

$$\int_{\partial\Omega} f \mathrm{d}S = \int_{S_+} f \mathrm{d}S + \int_{S_l} f \mathrm{d}S + \int_{S_-} f \mathrm{d}S \tag{3.2}$$

for any integrable function f on $\partial \Omega$. Let d be the maximum distance between S_+ and S_- . Then

$$S_l \to 0 \quad \text{as} \quad d \to 0. \tag{3.3}$$

Hence

$$\lim_{d \to 0} \int_{\partial \Omega} f \mathrm{d}S = \int_{S_+} f \mathrm{d}S + \int_{S_-} f \mathrm{d}S. \tag{3.4}$$

The terms involving volume integrals in (3.1) also vanish as $d \to 0$. We then let S shrink to a point P so that at the point P on σ, each equation in (3.1) reduces to the form

$$f^+(P) + f^-(P) = 0. \tag{3.5}$$

Noting that the outward normal on the surfaces S_+ and S_- are opposite to each other, we get from (3.1)

$$\begin{cases} (\rho q_n)_+ = (\rho q_n)_-, \\ (\boldsymbol{q}\rho q_n + p\boldsymbol{n})_+ = (\boldsymbol{q}\rho q_n + p\boldsymbol{n})_-, \\ \left[\rho q_n \left(\dfrac{\boldsymbol{q}^2}{2} + h \right) \right]_+ = \left[\rho q_n \left(\dfrac{\boldsymbol{q}^2}{2} + h \right) \right]_-, \end{cases} \tag{3.6}$$

where $q_n = \boldsymbol{q} \cdot \boldsymbol{n}$, $h = i + p/\rho$ and

$$i = \frac{1}{\gamma - 1} \frac{p}{\rho}$$

for a perfect gas with gamma-law. These five equations relate the flow variables on the opposite sides of a surface of discontinuity σ and are called the Rankine-Hugoniot conditions.

To sum up the analysis above, we see that the integral form conservation law equations (3.1) admit discontinuous solution, but on crossing a stationary surface of discontinuity the flow variables must satisfy the Rankine-Hugoniot conditions (3.6).

3.1.2   General Case: Moving Surface of Discontinuity

Suppose σ moves with velocity $\boldsymbol{s}$. We take the reference frame to move with velocity $\boldsymbol{s}$ and replace the fluid velocity $\boldsymbol{q}$ in by the relative velocity $\boldsymbol{q} - \boldsymbol{s}$. D'Alembert inertial force arises in the momentum equation—the second of (3.1), or (2.15)—as the reference frame is not an inertial one. However, the D'Alembert force is

proportional to the volume of Ω and hence disappears as its thickness d tends to zero (Figure 3.1). So we can make use of the results for the case of stationary σ by simply replacing $\boldsymbol{q}$ by the relative velocity $(\boldsymbol{q} - \boldsymbol{s})$ in the Rankine-Hugoniot conditions (3.6).

3.2   Classification of Flow Discontinuities

3.2.1   Special Case: Stationary Surface of Discontinuity

1. Case 1: $(\rho q_n)_+ = 0$

Let $\boldsymbol{\tau}_1$ and $\boldsymbol{\tau}_2$ be two independent orthogonal unit tangential vectors to σ (Figure 3.2).
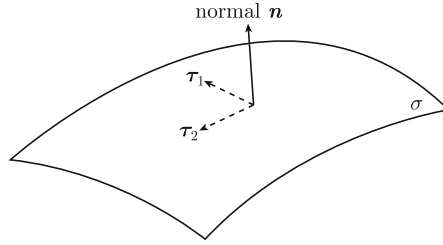
Figure 3.2   Tangential directions and normal direction of surface of discontinuity

From (3.6) we have $(\rho q_n)_+ = (\rho q_n)_- = 0$. This implies

$$(q_n)_+ = (q_n)_- = 0. \tag{3.7}$$

From (3.7) and the second of (3.6), we get

$$p_+ = p_-. \tag{3.8}$$

On the other hand, the flow variables ρ, q_{τ_1} and q_{τ_2} may be discontinuous, while p is continuous across σ. In this case, σ is called a *contact discontinuity*, or a slip surface.

2. Case 2: $(\rho q_n)_+ \neq 0$

From the first equation of (3.6), we have $(\rho q_n)_- \neq 0$. Then from the second of (3.6), we get

$$(\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_-,$$
$$(\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_-. \tag{3.9}$$

This means that the tangential velocities are continuous across the surface σ. Now if the normal velocity is also continuous, i.e., $(q_n)_+ = (q_n)_-$, then $\rho_+ = \rho_-$ and

$p_+ = p_-$ from the first and the second of (3.6). In this case, the flow is continuous and is a trivial one. For discontinuity to occur, we must have $(q_n)_+ \neq (q_n)_-$.

In the non-trivial case when $(q_n)_+ \neq (q_n)_-$, from the third of (3.6) and using $|\boldsymbol{q}|^2 = q_{\tau_1}^2 + q_{\tau_2}^2 + q_n^2$, we get

$$\left(\frac{1}{2}q_n{}^2 + h\right)_+ = \left(\frac{1}{2}q_n{}^2 + h\right)_- . \tag{3.10}$$

Taking dot product of both sides of the second of (3.6) with $\boldsymbol{n}$, we obtain

$$(\rho q_n{}^2 + p)_+ = (\rho q_n{}^2 + p)_-. \tag{3.11}$$

In this case, the surface of discontinuity is called a *shock wave.* We sum up the conditions relating the flow variables on the opposite sides of a shock as follows:

$$\begin{cases} (\rho q_n)_+ = (\rho q_n)_-, \quad (\rho q_n)_+ \neq 0, \\ (\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_-, \\ (\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_-, \\ (\rho q_n{}^2 + p)_+ = (\rho q_n{}^2 + p)_-, \\ \left(\dfrac{1}{2}q_n{}^2 + h\right)_+ = \left(\dfrac{1}{2}q_n{}^2 + h\right)_- . \end{cases} \tag{3.12}$$

Given the flow state upstream of a shock (the "+" state), we can calculate the downstream flow state (the "−" state) by using (3.12), which are also called the Rankine-Hugoniot conditions across a shock. There are three possible solutions.

(1) Solution 1:

$$\begin{cases} \rho_- = \rho_+, \\ p_- = p_+, \\ \boldsymbol{q}_- = \boldsymbol{q}_+, \end{cases} \tag{3.13}$$

which represents a continuous flow.

(2) Solution 2:

$$\begin{cases} \rho_- = \dfrac{\rho_+^2 q_n^2(\gamma + 1)}{\gamma \rho_+ q_{n+}^2 - \rho_+ q_{n+}^2 + 2\gamma p_+}, \\ p_- = \dfrac{2\rho_+ q_{n+}^2 - \gamma p_+ + p_+}{\gamma + 1}, \\ q_{n-} = \dfrac{\gamma q_{n+}^2 \rho_+ - \rho_+ q_{n+}^2 + 2\gamma p_+}{(\gamma + 1)\rho_+ q_{n+}}, \\ (q_{\tau_1})_- = (q_{\tau_1})_+, \\ (q_{\tau_2})_- = (q_{\tau_2})_+. \end{cases} \tag{3.14}$$

For this solution the entropy p/ρ^γ, pressure p, density ρ and temperature T of a fluid particle, increase on crossing the shock from the "+" side to the "−" side. This is the physically correct solution. We note that this is the way the entropy condition is applied.

(3) Solution 3: this solution is obtained from Solution 2 in (3.14) by interchanging the "+" with the "−". For this solution the entropy p/ρ^γ , pressure p, density ρ and temperature T of a fluid particle decrease on crossing the shock from the "+" side to the "−" side. This is physically incorrect and should be rejected according to the second law of thermodynamics. This is another way the entropy condition is applied.

3.2.2   General Case: Moving Surface of Discontinuity

Suppose σ moves with velocity $\boldsymbol{s}$. The Rankine-Hugoniot conditions are obtained, as explained in Section 3.1, by simply replacing the fluid velocity $\boldsymbol{q}$ in (3.6) by the relative velocity $(\boldsymbol{q}-\boldsymbol{s})$. Classification of flow discontinuities then follows similarly. The results are summed up as follows:

   1. Case 1: contact discontinuity or slip surface

$$p_+ = p_-,$$
$$(q_n - s_n)_+ = (q_n - s_n)_- = 0. \tag{3.15}$$

Hence

$$(q_n)_+ = (q_n)_- = s_n,$$

where $s_n = \boldsymbol{s} \cdot \boldsymbol{n}$, ρ, q_{τ_1} and q_{τ_2} may be discontinuous.

   2. Case 2: shock

The Rankine-Hugoniot conditions are

$$\begin{cases} [\rho(q_n - s_n)]_+ = [\rho(q_n - s_n)]_-, \\ \left[\rho(q_n - s_n)^2 + p\right]_+ = \left[\rho(q_n - s_n)^2 + p\right]_- , \\ (\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_1)_-, \\ (\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_+ = (\boldsymbol{q} \cdot \boldsymbol{\tau}_2)_-, \\ \left[\frac{1}{2}(q_n - s_n)^2 + h\right]_+ = \left[\frac{1}{2}(q_n - s_n)^2 + h\right]_- , \end{cases} \tag{3.16}$$

which involve the normal component of the velocity of the shock. Given the flow state upstream of a shock and its normal velocity, we can calculate the downstream flow state by using (3.16).

In the remaining of this chapter, we restrict our discussions to 1-D flow only. Here we list some useful results for 1-D flow. For a system of conservation laws

$$\frac{\partial \boldsymbol{E}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{E})}{\partial x} = \boldsymbol{0}, \tag{3.17}$$

the Rankine-Hugoniot shock jump conditions are

$$s(\boldsymbol{E}_+ - \boldsymbol{E}_-) = \boldsymbol{F}(\boldsymbol{E}_+) - \boldsymbol{F}(\boldsymbol{E}_-), \tag{3.18}$$

where $s = \mathrm{d}x/\mathrm{d}t$ is the speed of the shock. On the other hand, the speed s of a contact (or slipline) discontinuity is equal to the characteristic speed λ of (3.17), i.e.,

$$s = \lambda_i(\boldsymbol{E}_+) = \lambda_i(\boldsymbol{E}_-). \tag{3.19}$$

For the simpler case of a scalar conservation law $\partial E/\partial t + \partial F/\partial x = 0$, the shock speed is

$$s = \frac{F_+ - F_-}{E_+ - E_-}. \tag{3.20}$$

As an example, we consider the inviscid Burgers equation in three different PDE forms:

(a)

$$\frac{\partial u}{\partial t} + \frac{\partial (u^2/2)}{\partial x} = 0, \tag{3.21}$$

whose shock speed is

(b)

$$s_1 = \frac{\frac{1}{2}u_+^2 - \frac{1}{2}u_-^2}{u_+ - u_-} = \frac{1}{2}(u_+ + u_-);$$

$$\frac{\partial u^2}{\partial t} + \frac{\partial 2u^3/3}{\partial x} = 0, \tag{3.22}$$

whose shock speed is

$$s_2 = \frac{\frac{2}{3}u_+^3 - \frac{2}{3}u_-^3}{u_+^2 - u_-^2} = \frac{2}{3}\frac{u_+^2 + u_+ u_- + u_-^2}{u_+ + u_-} \neq s_1;$$

(when $u_+ \neq u_-$) and

(c)

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0. \tag{3.23}$$

We see that although the three equations are equivalent to each other for smooth solution, when a shock is present, it has different speeds according to (3.20). In particular, (3.21) and (3.22) are in conservation form, hence their shock speeds can be easily calculated using (3.20), and they are different!

This raises a serious question in computing non-linear conservation laws that does not exist for linear equations. The question is: which equation, amongst the many that are equivalent for smooth solution, should we use as the starting point for computing numerical solution which may be discontinuous? Of the three equations above, (3.23) is not in conservation form and therefore, according to Lax-Wendroff theorem ([1], see also [2]), will not capture shocks correctly and must be rejected. Although (3.21) and (3.22) both are in conservation form and can be used in numerical computation to capture the shock, each one correctly with respect to their form of equation. But if the Burgers equation is to represent some physical phenomena, the solutions obtained from (3.21) and (3.22) cannot be both correct as they have different shock speeds. Does any of these solutions correctly describe the physical phenomenon? If so, which one? These questions cannot be answered if one uses a PDE as the starting point for computation. To answer these questions, one must go back to the physical phenomena and derive its corresponding conservation law equations, which are usually in integral form, similar to (2.8). These integral conservation law equations can then be converted to PDE, with the assumption of solution smoothness. If the PDE coincides with (3.21), then the solution to (3.21) will correctly describe the physical phenomena. But if the PDE coincides with (3.22), then the solution to (3.22) will correctly describe the physical phenomena. If the PDE does not coincide with (3.21), nor (3.22), (this implies that (3.21) and (3.22) are obtained from some PDE by mathematically equivalent manipulations), then the solutions obtained from them do not describe the physical phenomena.

This example serves to emphasize the importance of using the integral form conservation law equations, not PDE, as the starting point of computation; in doing so, the difficulties, or confusions, mentioned above are avoided.

3.3   Riemann Problem and its Solution

3.3.1   Elementary Waves

The Riemann problem for a system of conservation laws in one space dimension is the special initial value problem

$$
\text{RP} : \begin{cases} \dfrac{\partial \boldsymbol{E}}{\partial t} + \dfrac{\partial \boldsymbol{F}(\boldsymbol{E})}{\partial x} = 0, & t > 0, x \in \mathbf{R}, \\ \boldsymbol{E}(x,0) = \begin{cases} \boldsymbol{E}_l, & x < 0, \\ \boldsymbol{E}_r, & x > 0, \end{cases} \end{cases} \tag{3.24}
$$

where $\boldsymbol{E} = (u_1, \cdots, u_m)$, and we assume that $\boldsymbol{F}(\boldsymbol{E}) = (F_1(\boldsymbol{E}), \cdots, F_m(\boldsymbol{E}))$ is smooth in a neighborhood $M \subset \mathbf{R}^m$, and the Jacobian matrix $\boldsymbol{A}(\boldsymbol{E}) = \dfrac{\mathrm{d}\boldsymbol{F}(\boldsymbol{E})}{\mathrm{d}\boldsymbol{E}}$ has m real and distinct eigenvalues $\lambda_1(\boldsymbol{E}) < \cdots < \lambda_m(\boldsymbol{E})$ in M. The initial data $\boldsymbol{E}_l$ and $\boldsymbol{E}_r$ are constant.

The Riemann problem plays a central role in the theory of systems of conservation laws as well as in numerical methods for computing solution to the general initial value problem of conservation laws. Indeed, its solution reveals all the important physics of wave propagation, and is the building brick for the two most important numerical methods: the Godunov method and the random choice method. It is also of crucial importance in the shock fitting (front tracking) method.

The simplicity of the Riemann problem, as against the problem with general initial data, derives from its property that both the equation and Riemann data are invariant under the transformation

$$
t \to \alpha t, \quad x \to \alpha x, \quad \alpha > 0. \tag{3.25}
$$

Consequently, solution to RP exists in the form of similarity solution, i.e.,

$$
\boldsymbol{E} = \boldsymbol{E}(\xi), \quad \xi = \frac{x}{t}. \tag{3.26}
$$

Indeed, using (3.26) in (3.24) yields

$$
(\boldsymbol{A} - \xi \boldsymbol{I})\frac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} = \boldsymbol{0}. \tag{3.27}
$$

Solution to the Riemann problem (3.24) may possibly be continuous or be discontinuous.

Continuous solutions are to be found from (3.27). There are two possibilities:

$$\frac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} = \boldsymbol{0} \text{ or } \frac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} \neq \boldsymbol{0}.$$

(1) $\dfrac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} = \boldsymbol{0}$. This gives a constant solution

$$\boldsymbol{E}\left(\frac{x}{t}\right) = \text{const..} \tag{3.28}$$

(2) $\dfrac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} \neq \boldsymbol{0}$, then ξ is an eigenvalue of $\boldsymbol{A}$ and $\dfrac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi}$ is parallel to its right eigenvector, i.e.,

$$\begin{cases} \lambda_i(\boldsymbol{E}) = \xi, \\[2mm] \dfrac{\mathrm{d}\boldsymbol{E}}{\mathrm{d}\xi} = C\boldsymbol{r}^{(i)}(\boldsymbol{E}), C = \text{const..} \end{cases} \tag{3.29}$$

The solution is called a centered rarefaction wave.

Discontinuous solutions of conservation law, as mentioned in last subsection, include a shock wave and a contact wave.

(1) Shock wave. Across a shock wave, the states $\boldsymbol{E}_+$ and $\boldsymbol{E}_-$ on its two sides must satisfy

$$s(\boldsymbol{E}_+ - \boldsymbol{E}_-) = \boldsymbol{F}(\boldsymbol{E}_+) - \boldsymbol{F}(\boldsymbol{E}_-), \tag{3.30}$$

where $s = \dfrac{\mathrm{d}x}{\mathrm{d}t}$ is the speed of the shock wave.

(2) Contact wave. The speed of a contact wave must be equal to the characteristic speed, i.e.,

$$s = \lambda_i(\boldsymbol{E}_+) = \lambda_i(\boldsymbol{E}_-). \tag{3.31}$$

Shock wave, contact wave and rarefaction wave are called elementary waves.

It is known that a linearly degenerated λ-field gives rise to a contact wave, whereas a genuinely non-linear λ-field gives rise to either a shock wave or a centered rarefaction wave, i.e., see Chapter 17, Theorems 17.9, 17.11 and 17.17 in Smoller[3].

3.3.2   Existence and Uniqueness of Solution

Using these elementary waves, a solution to the Riemann problem (3.24) can be constructed provided that $|\boldsymbol{E}_\ell - \boldsymbol{E}_r|$ is small. In particular, we have the following theorem (Smoller [3], Chapter 17):

Theorem 3.1 (Local existence and uniqueness of solution)   Let $\boldsymbol{E}_\ell \in M$ and suppose that the system (3.24) is strictly hyperbolic and that each characteristic field is either genuinely nonlinear or linearly degenerate in M. Then there is a

neighborhood $\widetilde{M} \subset M$ of $\boldsymbol{E}_\ell$ such that if $\boldsymbol{E}_r \in \widetilde{M}$, the Riemann problem (3.24) has a solution. This solution consists of at most $(m+1)$-constant states separated by shocks, centered rarefaction waves or contact discontinuities. There is precisely one solution of this kind in $\widetilde{M}$.

3.3.3   Structure of Riemann Solution for 1-D Flow

The conservation law Euler equations for 1-D flow of a perfect gas with a γ-law are

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ \rho e \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(\rho e + p) \end{pmatrix} = \boldsymbol{0}, \tag{3.32}$$

where

$$e = \frac{1}{2}u^2 + \frac{1}{\gamma - 1}\frac{p}{\rho}. \tag{3.33}$$

Here u is velocity, p is pressure, ρ is density and e is specific total energy. System (3.32) is hyperbolic and in conservation form.

Since across a contact the pressure p and velocity u are continuous, the Riemann problem for 1-D Euler equation in Eulerian coordinates is better formulated using primative variables $\boldsymbol{W} = (\rho, u, p)$. Thus

$$\text{RP:} \begin{cases} \dfrac{\partial \boldsymbol{W}}{\partial t} + \boldsymbol{A}(\boldsymbol{W})\dfrac{\partial \boldsymbol{W}}{\partial x} = 0, \quad t > 0, \\ \boldsymbol{W}(x, t = 0) = \begin{cases} \boldsymbol{W}_\ell, & x < 0, \\ \boldsymbol{W}_r, & x > 0, \end{cases} \end{cases} \tag{3.34}$$

where the Jacobian matrix is

$$\boldsymbol{A} = \begin{pmatrix} u & \rho & 0 \\ 0 & u & 1/\rho \\ 0 & \rho a^2 & u \end{pmatrix} \tag{3.35}$$

and $a = (\gamma p/\rho)^{1/2}$ is the speed of sound. The wave structure for the Riemann problem is shown in Figure 3.3.

The following theorem[3] gives the global existence of solution to the Riemann problem (3.34):

Theorem 3.2(Global existence and uniqueness of solution)   There is a unique solution to the Riemann problem (3.34) if and only if

$$u_\ell - u_r < \frac{2}{\gamma - 1}(a_\ell + a_r). \tag{3.36}$$

If (3.36) is violated, then a vacuum is present in the solution. The solution is composed by at most 4 constant states (including the states $\boldsymbol{W}_\ell$ and $\boldsymbol{W}_r$) separated by a shock (or a rarefaction), a contact, and a rarefaction (or a shock). The solution is sketched in Figure 3.3.
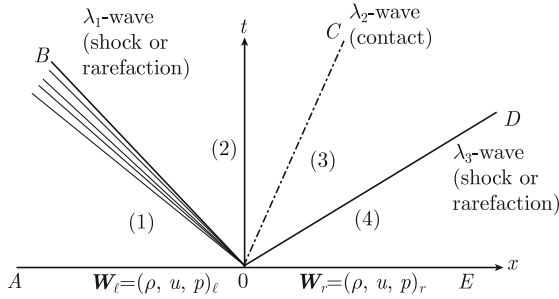
Figure 3.3   Structure of Riemann solution for 1-D Euler equations

3.3.4   Solution of the Elementary Waves

The eigenvalues of $\boldsymbol{A}$ and their corresponding right eigenvectors are

$$\lambda_1 = u - a, \quad \lambda_2 = u, \quad \lambda_3 = u + a, \tag{3.37}$$

$$\boldsymbol{r}^{(1)} = \begin{pmatrix} 1 \\ -\dfrac{a}{\rho} \\ a^2 \end{pmatrix}, \quad \boldsymbol{r}^{(2)} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{r}^{(3)} = \begin{pmatrix} 1 \\ \dfrac{a}{\rho} \\ a^2 \end{pmatrix}. \tag{3.38}$$

The λ_2-field is linearly degenerate, giving rise to a contact wave. On the other hand, both λ_1-field and λ_3-field are genuinely non-linear, each of which giving rise to a shock or a rarefaction wave.

(1) The λ_2-field. Across the contact

$$p_2 = p_3 = p^*,$$

$$u_2 = u_3 = u^*, \tag{3.39}$$

but there is no relation between ρ_2 and ρ_3.

(2) The λ_1-field.

Case 1: $p_2 > p_\ell$. In this case, the wave is a shock, because only on crossing (time t increases, see Figure 3.3) a shock wave can pressure of a particle increases; it cannot do so on crossing a rarefaction wave. We also pointed out earlier that as pressure increases crossing a shock, so does entropy of the fluid particle. So we

have implicitly applied the entropy condition at this stage. The relations between the state $\boldsymbol{W}_\ell$ and the state $\boldsymbol{W}_2$ in region (2) (Figure 3.3) are thus given by the Rankine-Hugoniot conditions

$$
\begin{cases}
\rho_2(u_2 - s) = \rho_l(u_l - s), \\
\rho_2(u_2 - s)^2 + p_2 = \rho_l(u_l - s)^2 + p_l, \\
\dfrac{1}{2}(u_2 - s)^2 + \dfrac{\gamma}{\gamma - 1}\dfrac{p_2}{\rho_2} = \dfrac{1}{2}(u_l - s)^2 + \dfrac{\gamma}{\gamma - 1}\dfrac{p_l}{\rho_l},
\end{cases}
\tag{3.40}
$$

where s is the shock speed. With the state $\boldsymbol{W}_\ell$ given, there are two solutions of (3.40) for the state $\boldsymbol{W}_2$: one with entropy-increasing, the other decreasing. The entropy-increasing solution is the physically acceptable one and is given in terms of the pressure ratio $\alpha = p_2/p_\ell > 1$ as follows:

$$
\begin{cases}
\dfrac{\rho_2}{\rho_\ell} = \dfrac{\alpha + \dfrac{\gamma - 1}{\gamma + 1}}{\dfrac{\gamma - 1}{\gamma + 1}\alpha + 1}, \\
u_2 = u_\ell - \dfrac{2(\alpha - 1)a_\ell}{(\gamma + 1)\alpha + \gamma - 1}\sqrt{\dfrac{\gamma + 1}{2\gamma}\alpha + \dfrac{\gamma - 1}{2\gamma}}, \\
s = u_\ell - a_\ell\sqrt{\dfrac{\gamma + 1}{2\gamma}\alpha + \dfrac{\gamma - 1}{2\gamma}},
\end{cases}
\tag{3.41}
$$

where $a_\ell = \sqrt{\gamma p_\ell/\rho_\ell}$.

Note that as the shock strength tends to zero, the pressure ratio $\alpha = p_2/p_\ell$ tends to unity and the shock speed s approaches the characteristic speed $\lambda_1 = u_\ell - a_\ell$.

Case 2: $p_2 \leqslant p_\ell$. In this case, the wave is a rarefaction, because only on crossing (time t increases, see Figure 3.3) a rarefaction wave can pressure of a particle decreases; it cannot do so on crossing a shock wave. The solution in the rarefaction wave is to be found by solving the following system of ordinary differential equations:

$$
\begin{cases}
\dfrac{\mathrm{d}\boldsymbol{W}}{\mathrm{d}\xi} = C\boldsymbol{r}^{(1)}(\boldsymbol{W}), C = \text{const.}, \\
\lambda_1(\boldsymbol{W}) = \xi.
\end{cases}
\tag{3.42}
$$

With $\boldsymbol{W} = (\rho, u, p)^\mathrm{T}$ and (3.38), we may write the first equation of (3.42) as

$$
\frac{\mathrm{d}}{\mathrm{d}\xi}\begin{pmatrix} \rho \\ u \\ p \end{pmatrix} = C\begin{pmatrix} 1 \\ -\dfrac{a}{\rho} \\ a^2 \end{pmatrix},
\tag{3.43}
$$

hence

$$\begin{cases} \dfrac{\mathrm{d}\rho}{\mathrm{d}p} = \dfrac{1}{a^2} = \dfrac{\rho}{\gamma p}, \\[3mm] \dfrac{\mathrm{d}u}{\mathrm{d}p} = -\dfrac{1}{\rho a} = -\dfrac{1}{\sqrt{\gamma p\rho}}. \end{cases} \tag{3.44}$$

Integrating the first equation of (3.44) gives $p/\rho^\gamma = $ const., i.e., the entropy is constant. Hence

$$\frac{\rho}{\rho_\ell} = \left(\frac{p}{p_\ell}\right)^{1/\gamma}. \tag{3.45}$$

Substituting (3.45) in the second equation of (3.44) and integrating, we get

$$u = u_\ell + \frac{2a_\ell}{\gamma - 1}\left[1 - \left(\frac{p}{p_\ell}\right)^{(\gamma-1)/2\gamma}\right]. \tag{3.46}$$

(3.45) and (3.46) express the flow variables ρ and u in terms of the pressure ratio p/p_ℓ . We now use the second equation of (3.42) to express p/p_ℓ as a function of ξ. Since $a = \sqrt{\gamma p/\rho}$, we get $a/a_\ell = (p/p_\ell)^{(\gamma-1)/2\gamma}$. Using (3.37), the second equation of (3.42) becomes

$$\begin{aligned} \xi &= u - a \\ &= u_\ell + \frac{2a_\ell}{\gamma - 1}\left[1 - \left(\frac{p}{p_\ell}\right)^{(\gamma-1)/2\gamma}\right] - a_\ell\left(\frac{p}{p_\ell}\right)^{(\gamma-1)/2\gamma} \\ &= u_\ell + \frac{2a_\ell}{\gamma - 1} - \frac{\gamma + 1}{\gamma - 1}a_\ell\left(\frac{p}{p_\ell}\right)^{(\gamma-1)/2\gamma}. \end{aligned}$$

Therefore

$$\frac{p}{p_\ell} = \left[\frac{\gamma - 1}{\gamma + 1}\left(\frac{u_\ell}{a_\ell} + \frac{2}{\gamma - 1} - \frac{\xi}{a_\ell}\right)\right]. \tag{3.47}$$

Equations (3.45)~(3.47) give the complete solution of the flow inside the rarefaction wave. Of particular interest is the velocity u_2,

$$u_2 = u_\ell + \frac{2a_\ell}{\gamma - 1}\left[1 - \left(\frac{p_2}{p_\ell}\right)^{(\gamma-1)/2\gamma}\right]. \tag{3.48}$$

Also, the position of the first ray (upstream) is

$$\frac{x}{t} = \xi_\ell = u_\ell - a_\ell, \tag{3.49}$$

and the position of the last ray (downstream) is

$$\begin{aligned} \frac{x}{t} = \xi_2 &= u_\ell + \frac{2a_\ell}{\gamma - 1} - \frac{\gamma + 1}{\gamma - 1}a_\ell\left(\frac{p_2}{p_\ell}\right)^{(\gamma-1)/2\gamma} \\ &= u_2 - a_2. \end{aligned} \tag{3.50}$$

The solutions $(3.45)\sim(3.47)$ are to be used for

$$u_\ell - a_\ell \leqslant \xi = \frac{x}{t} \leqslant u_2 - a_2, \tag{3.51}$$

where

$$a_2 = a_\ell \left(\frac{p_2}{p_\ell}\right)^{(\gamma-1)/2\gamma}. \tag{3.52}$$

(3) The λ_3-field. This can be studied similarly to the λ_1-field.

Case 1: $p_3 > p_r$. This results in a shock and we have

$$\begin{cases} \dfrac{\rho_3}{\rho_r} = \dfrac{\alpha + \dfrac{\gamma-1}{\gamma+1}}{\dfrac{\gamma-1}{\gamma+1}\alpha + 1}, \quad \alpha = \dfrac{p_3}{p_r}, \\[4mm] u_3 = u_r - \dfrac{2(\alpha-1)a_r}{(\gamma+1)\alpha + \gamma - 1}\sqrt{\dfrac{\gamma+1}{2\gamma}\alpha + \dfrac{\gamma-1}{2\gamma}}, \\[4mm] s = u_r + a_r\sqrt{\dfrac{\gamma+1}{2\gamma}\alpha + \dfrac{\gamma-1}{2\gamma}}. \end{cases} \tag{3.53}$$

Again, as the shock strength tends to zero, the pressure ratio $\alpha = p_3/p_r$ tends to unity and the shock speed s approaches the characteristic speed $\lambda_3 = u_r + a_r$.

Case 2: $p_3 \leqslant p_r$. This results in a rarefaction wave, of which the solution is

$$\begin{cases} \dfrac{\rho}{\rho_r} = \left(\dfrac{p}{p_r}\right)^{1/\gamma}, \\[4mm] u = u_r + \dfrac{2a_r}{\gamma-1}\left[1 - \left(\dfrac{p}{p_r}\right)^{(\gamma-1)/2\gamma}\right], \\[4mm] \dfrac{p}{p_r} = \left[\dfrac{\gamma-1}{\gamma+1}\left(\dfrac{u_r}{a_r} + \dfrac{2}{\gamma-1} - \dfrac{\xi}{a_r}\right)\right]^{(\gamma-1)/2\gamma}. \end{cases} \tag{3.54}$$

In particular,

$$u_3 = u_r + \dfrac{2a_r}{\gamma-1}\left[1 - \left(\dfrac{p_3}{p_r}\right)^{(\gamma-1)/2\gamma}\right]. \tag{3.55}$$

Solution (3.54) holds for

$$u_3 + a_3 \leqslant \xi = \frac{x}{t} \leqslant u_r + a_r, \tag{3.56}$$

where

$$a_3 = a_r \left(\frac{p_3}{p_r}\right)^{(\gamma-1)/2\gamma}. \tag{3.57}$$

3.3.5   Solution Algorithm

We are now in the position to give an algorithm for constructing the solution to the
Riemann problem (3.34). We first note that the pressure and velocity in regions
(2) and (3) are the same, and denote them with (*), i.e.,

$$p_2 = p_3 = p^*, \tag{3.58}$$

$$u_2 = u_3 = u^*. \tag{3.59}$$

We shall make use of (3.58) and (3.59) to derive a single equation for p^* and then
solve it.

   We already know (Figure 3.3) that if $p^* \leqslant p_\ell$, the λ_1-field is a rarefaction wave
and $u^*(= u_2)$ is given by (3.48). But if $p^* > p_\ell$, the λ_1-field is a shock and u^* is
given by the second equation of (3.41). We therefore define a function

$$\psi_1(p^*) = \begin{cases} u_\ell + \dfrac{2a_\ell}{\gamma - 1}\left[1 - \left(\dfrac{p^*}{p_\ell}\right)^{(\gamma-1)/2\gamma}\right], & p^* \leqslant p_\ell, \\[4ex] u_\ell - \dfrac{2\left(\dfrac{p^*}{p_\ell} - 1\right)a_\ell}{(\gamma+1)\dfrac{p^*}{p_\ell} + \gamma - 1}\sqrt{\dfrac{\gamma+1}{2\gamma}\dfrac{p^*}{p_\ell} + \dfrac{\gamma-1}{2\gamma}}, & p^* > p_\ell. \end{cases} \tag{3.60}$$

Similarly, for the λ_3-field we define a function

$$\psi_3(p^*) = \begin{cases} u_r + \dfrac{2a_r}{\gamma - 1}\left[1 - \left(\dfrac{p^*}{p_r}\right)^{(\gamma-1)/2\gamma}\right], & p^* \leqslant p_r, \\[4ex] u_r - \dfrac{2\left(\dfrac{p^*}{p_r} - 1\right)a_r}{(\gamma+1)\dfrac{p^*}{p_r} + \gamma - 1}\sqrt{\dfrac{\gamma+1}{2\gamma}\dfrac{p^*}{p_r} + \dfrac{\gamma-1}{2\gamma}}, & p^* > p_r. \end{cases} \tag{3.61}$$

We further define a function

$$\phi(p^*) = \psi_1(p^*) - \psi_3(p^*). \tag{3.62}$$

Equation (3.59) then requires

$$\phi(p^*) = 0. \tag{3.63}$$

This is the required equation for p^* which is to be solved numerically, using New-
ton's method of iteration, say.

With p^* found, the solution in the whole flow filed, i.e., (ρ, u, p) in regions (1)$\sim$(4), as well as the flow inside the rarefaction wave, can be found using the formulae derived early in this section. Of course, $\boldsymbol{W}_1 = \boldsymbol{W}_\ell$ in region (1) and $\boldsymbol{W}_4 = \boldsymbol{W}_r$ in region (4). A good computer program for solving the Riemann problem can be found in [4].

3.4   Preliminary Considerations of Numerical Computation

3.4.1   The Question of Weak Solution to Partial Differential Equations

In many CFD text books, the starting point is the Euler equations (3.32), which are of hyperbolic type. For such a system, no matter how smooth the initial data, the solution may become discontinuous when the characteristics of the same family intersect. Consequently, solution to the Euler equations in the classical sense— the functions (ρ, u, p) are smooth (having continuous first derivatives) and satisfy (3.32)—does not exist. It is then necessary to extend the classical concept of solution to *weak solution*, for which the functions (ρ, u, p) are piece-wise smooth such that they satisfy (3.32) in smooth flow region and satisfy the Rankine-Hugoniot conditions on crossing discontinuities. The analytic solution to the Riemann problem given in Section 3.3 is indeed a weak solution. Numerical solution to the Euler equations means to compute their weak solution. One way of doing this is to revert the Euler equations back to the integral form conservation equations

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \boldsymbol{E}\mathrm{d}\Omega = - \int_{\partial\Omega} \boldsymbol{F}(\boldsymbol{E}) \cdot \boldsymbol{n}\mathrm{d}S, \tag{3.64}$$

where $\boldsymbol{E} = (\rho, \rho u, \rho e)^{\mathrm{T}}$.

On the other hand, we see from Section 2.4 that the integral form conservation equations (3.64) is equivalent to the Euler PDE (3.32) in smooth flow region. We also see from Section 3.1 that the integral form equations automatically contain the Rankine-Hugoniot conditions on crossing a discontinuity. It is therefore clear that solving the integral form equations (3.64) will automatically gives the weak solution to the Euler equations (3.32) without even introducing the concept of weak solution, which belongs to partial differential equations. For this reason, and for the example discussed at the end of Section 3.2, and also for the reason to be give presently, we emphasige that the starting point of computing inviscid flow should be the integral form conservation equations.

3.4.2   Finite Volume Formulation

The main task in CFD is to compute numerical solution to the following initial value problem, i.e., in 1-D case,

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega} \boldsymbol{E} \mathrm{d}\Omega = -\int_{\partial\Omega} \boldsymbol{F}(\boldsymbol{E}) \cdot \boldsymbol{n} \mathrm{d}S, \tag{3.65}$$

with initial condition $\boldsymbol{E}(x,0) = \boldsymbol{E}_0(x)$.

If we take cell j to be the control volume $\Omega : x_{j-1/2} < x < x_{j+1/2}$, (3.65) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t} \boldsymbol{E}_j = -\frac{\boldsymbol{F}(\boldsymbol{E}(x_{j+1/2},t)) - \boldsymbol{F}(\boldsymbol{E}(x_{j-1/2},t))}{h}, \tag{3.66}$$

where

$$\boldsymbol{E}_j(t) = \frac{1}{h} \int_{x_{j-1/2}}^{x_{j+1/2}} \boldsymbol{E}(x,t) \mathrm{d}x \tag{3.67}$$

is cell average and $h = x_{j+1/2} - x_{j-1/2}$ is cell width. Integrating (3.66) from time t^n to t^{n+1} gives

$$\boldsymbol{E}_j^{n+1} = \boldsymbol{E}_j^n - \frac{1}{h} \left[\int_{t^n}^{t^{n+1}} \boldsymbol{F}(\boldsymbol{E}(x_{j+1/2},t)) \mathrm{d}t - \int_{t^n}^{t^{n+1}} \boldsymbol{F}(\boldsymbol{E}(x_{j-1/2},t)) \mathrm{d}t \right]. \tag{3.68}$$

We note that in deriving (3.68) there has been no approximation made, hence it remains exact. We also note that (3.68) belongs to the class of conservative numerical schemes which ensure correct capturing of shock waves, according to Lax-Wendroff theorem[1].

Different methods of evaluating the flux integrals on the right hand side of (3.68), particularly $\boldsymbol{E}$ at the interfaces, constitute different numerical schemes. For example, in the Glimm's random choice scheme[5, 6], $\boldsymbol{E}$ at the interface between cell j and cell $j+1$ is to be obtained from the solution of the Riemann problem with $\boldsymbol{E}_j$ and $\boldsymbol{E}_{j+1}$ at time t_n as Riemann data, where $\boldsymbol{E}_j$ is the value of $\boldsymbol{E}$ at some point in cell j which is chosen randomly. Although random choice scheme works well for 1-D flow, it is difficult to extend to multi-dimensional flow. Another scheme is the famous Godunov scheme to be discussed in the next section.

3.5   Godunov Scheme

In 1959, Godunov[7] proposed a finite volume scheme, which used the Riemann solution as a building brick for the numerical solution of compressible Euler equations. Under the assumption of constant state inside each control volume at the

beginning of each time step t^n, i.e.,

$$\boldsymbol{E}(x, t^n) = \boldsymbol{E}_j(x, t^n) = \text{const}, \quad \text{for} \quad x_{j-1/2} < x \leqslant x_{j+1/2},$$

the initial condition around a cell interface becomes

$$\boldsymbol{E}(x, t^n) = \begin{cases} \boldsymbol{E}_j, & x \leqslant x_{j+1/2}, \\ \boldsymbol{E}_{j+1}, & x > x_{j+1/2}, \end{cases} \tag{3.69}$$

and there is a corresponding exact Riemann solution for the Euler equations

$$\boldsymbol{E}_{j+1/2}(x, t) = \boldsymbol{E}_{\text{RP}} \left(\frac{x - x_{j+1/2}}{t - t^n} \right).$$

As a result, the fluxes in (3.68) can be evaluated based on the above solution, i.e.,

$$\boldsymbol{F}(\boldsymbol{E}(x_{j+1/2}, t)) = \boldsymbol{F}(\boldsymbol{E}_{\text{RP}}(0)). \tag{3.70}$$

The above scheme based on the exact Riemann solution is called Godunov scheme.

Godunov scheme is composed of two phases: cell averaging phase followed by solving a sequence of Riemann problems for every pair of adjacent cells using the cell averages as Riemann data. The second phase is exact, and the only error arises from cell averaging in the first phase. Godunov scheme has the following desirable properties:

(1) It is conservative and consistent and hence, according to Lax-Wandroff theorem[1], it can capture shocks correctly whenever it converges, which requires numerical stability of the scheme.

(2) It can always be made stable by suitably adjusting the time step in the marching computation, according to the CFL stability condition.

(3) For linear systems of conservation law equations it reduces to the upwind scheme, which is the least dissipative scheme among all first-order schemes.

(4) It is applicable to non-uniform cells.

(5) It is entropy-satisfying: the entropy condition is applied in obtaining the Riemann solution.

However, due to cell averaging Godunov scheme is only first order accurate for smooth flow, and zeroth order accurate across discontinuities (shocks and contacts). For 1-D flow, the Godunov-averaging errors across a shock can be avoided by using the shock-adaptive Godunov scheme ([8, 9]; see Section 4.2). On the other hand,

averaging errors across a contact is inherent in Eulerian coordinates which do not, in general, align with the contact. Errors of this type can only be avoided by using coordinates of the Lagrangian-type (see Chapter 4).

In comparison with traditional central difference methods for the Euler equations, one of the merit of the Godunov scheme is its clear physical picture, where gas dynamics process gets involve in the construction of the numerical scheme through the flux evaluation. The Riemann solution tells us exactly what is happening at a cell interface if the initial condition is distributed piecewise constant. Therefore, this methodology can be naturally extended to more complicated system once the dynamics under simple initial condition can be obtained.

The Euler equations present a shock discontinuity with zero thickness. Numerically, this sharp transition cannot be properly resolved by any refined cell size. Therefore, the real equations the Godunov method solves numerically with a limited cell resolution, must be viscous governing equations, where any flow structure should be compatible with the cell size. In other words, an additional dissipation is implicitly added in the Godunov method. Certainly, if the shock discontinuity or slip surface can be well resolved by shock or contact fitting methods, the additional dissipation can be avoided (see Chapter 4).

The use of the Riemann solution somehow helps to introduces the "appropriate" numerical dissipation needed in the discontinuity region. However, contrary to many statements in the literatures, the amount of dissipation in the Godunov method is not coming from the characteristic waves of the Riemann solution or the so-called upwinding mechanism, but is coming from the preparation of the initial data at the beginning of each time. For example, the preparation of piecewise constant state inside each cell is based on the conservation of mass, momentum, and total energy. In this process, the kinetic energy is not conserved and the lost kinetic energy is automatically transferred into thermal one[10]. As a consequence, in multi-dimensional flow simulation the amount of dissipation in the Godunov method is closely related to the relative orientation of the shock discontinuity and the mesh distribution[11]. Therefore, the dissipation in the Godunov method is a passive one, but it may not be adequate in flow computation. This observation can be used to analyze many defects of the Godunov method in the flow simulations (see Section 3.7). Also, the assumption of discontinuity at a cell interface in the

Godunov method is valid only for the cases where the cell resolution is not enough to resolve the flow structure. In the well-resolved continuum flow regime, this assumption is not necessary and the high-resolution scheme is basically trying to reduce such a discontinuity jump.

3.6   High Resolution Schemes and Limiters

In order to extend the Godunov method to high resolution, we have to modify the piecewise constant initial condition first. A natural extension is to introduce slopes inside each cell for the preparation of initial condition. However, a naive slope construction can easily introduce over- or under-shoot in the discontinuity regions. On the other hand, if we simply connect the cell averages, the initial data in the Godunov scheme will be a continuous function across the cell interface, therefore it goes back to the traditional central difference schemes, where the post-shock oscillations are intrinsically rooted. Therefore, in order to properly construct a high resolution scheme, the following criteria have to be satisfied. Firstly, in the smooth region, the initial reconstruction should be as close as possible to a continuous one across a cell interface; Secondly, in the discontinuity region the large jump at the cell interface has to be kept; Thirdly, the reconstruction has to avoid over- and under-shoot in all situations. The reconstruction method, which could satisfy all above requirements, must use non-linear limiters to limit or choose the appropriate one from all possibly reconstructed slopes. In other words, even for linear equation a non-linear reconstruction mechanism has to be involved. With the inclusion of slopes, the Riemann solution should theoretically be replaced by the generalized Riemann solution[12]. However, there are simpler ways to achieve high resolution, and one of them is the monotonic upstream-centered scheme for conservation laws (MUSCL) of van Leer[13].

The reconstruction can be further replaced by more accurate approximations as well. For example, the reconstruction can use quadratic, as in the piecewise parabolic method (PPM)[14] or even higher order reconstructions as in the ENO (essentially non-oscillatory) or WENO (weighted ENO) methods[15−17].

For any high resolution scheme, it is basically a choice between the use of the continuous and discontinuous initial data for the flux evaluation. Currently, most high resolution schemes make the choices between the continuous and discontinu-

ous ones based heavily on the numerics. In other words, it is the stencil and their coefficients that determine the final reconstruction. For complicated flow problems, such as the viscous flow computation inside a boundary layer of the Navier-Stokes equations, a purely numerical choice may trigger serious problems when there are only a few points inside such a layer, and the solution may strongly depend on the order of the reconstruction. For a successful scheme solving the hypersonic viscous heat conducting flow equations, the choice between the continuous and discontinuous initial data reconstruction should not only be based on the numerics, such as different stencil or limiters, but also be based on the flow physics to choose the appropriate one. For example, an ideal case is the following. Inside the viscous heat conducting boundary layer, even with the possible "inappropriate" discontinuity initial reconstruction, according to local Reynolds number the flow physics will quickly smear the discontinuity and generate a continuous one for the flux evaluation. Otherwise, the high resolution scheme will automatically generate waves which don't exist in the boundary layer, to poison the numerical solution. One good example to use flow physics to make the choices is the gas-kinetic BGK scheme, where both discontinuous and continuous flow distributions are included in the final gas distribution function at a cell interface, and the weight between them depends on the flow physics[18, 19]. Therefore, the boundary solution is not sensitive to the limiters used at all.

High resolution schemes generally give more accurate numerical results in the smooth flow region, but different combinations of approximate Riemann solvers and limiters give slightly different results[4]. Although they also improve resolution of shocks and contacts, they remain of zeroth order accuracy there. In all Eulerian computations presented in this book, we use Godunov scheme with the exact Riemann solver and upgrade it to high resolution via the MUSCL methodology, employing the MINMOD limiter. Some of the computed results are reported in Section 4.5 for comparison with the unified coordinates computation.

3.7   Defects of Eulerian Computation

Godunov scheme and high resolution schemes have enjoyed great success in CFD[4]; in particular the shock can be resolved with 2 to 3 computational cells in high resolution scheme. However, Eulerian computation using Godunov scheme or higher

resolution schemes with various combinations of approximate Riemann solvers and limiters have inherent defects. The following list represents known defects[20−23] of Eulerian computation for solving (3.32):

(1) Contact smearing;

(2) Slow moving shocks;

(3) Sonic point glitch;

(4) Start-up errors;

(5) Low pressure flow;

(6) Wall-overheating;

(7) Strong rarefaction waves.

Defects (4)∼(7) are also shared in Lagrangian computation. In Section 4.5, we will detail out these defects, point out their causes and show how they all can be cured or avoided by using the unified coordinates.

References

[1] P.D. LAX AND B. WENDROFF. Systems of conservation laws. *Comm. Pure and Applied Mathematics*, 13: 217-237, 1960.

[2] T.Y. HOU AND P.G. LEFLOCH. Why nonconservative schemes converge to wrong solutions: error analysis. *Math. Comput.*, 62: 497-530, 1994.

[3] J. SMOLLER. *Shock Waves and Reaction-Diffusion Equations*. New York: Springer-Verlag, 1994.

[4] E.F. TORO. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. New York: Springer-Verlag, 1999.

[5] J. GLIMM. Solution in the large for nonlinear hyperbolic systems of equations. *Comm. Pure. Appl. Math.*, 18: 697-715, 1965.

[6] A. J. CHORIN. Random choice solutions of hyperbolic systems. *J. Comput. Phys.*, 22: 517-533, 1976.

[7] S.K. GODUNOV. A difference scheme for numerical computation of discontinuous Solutions of Hydrodynamic Equations. *Math. Sbornik*, 47: 271, 1959.

[8] W.H. HUI AND C.Y. LOH. A new Lagrangian method for steady supersonic flow computation, Part III: strong shocks. *J. Comput. Phys.*, 103: 465-471, 1992.

[9] C.Y. LEPAGE AND W.H. HUI. A shock-adaptive Godunov scheme based on the generalized Lagrangian formulation. *J. Comput. Phys.*, 122: 291-299, 1995.

[10] K. XU AND J.S. HU. Projection dynamics in Godunov-type schemes *J. Comput. Phys.*, 142: 412-427, 1998.

[11] K. XU AND Z.W. LI. Dissipative mechanism in Godunov-type schemes. *Int. J. Numer. Methods in Fluids*, 37: 1-22, 2001.

[12] M. BEN-ARTZI AND J. FALCOVITZ. Generalized Riemann problems in computational fluid dynamics. Cambridge: *Cambridge University Press*, 2003.

[13] B. VAN LEER. Towards the ultimate conservative difference scheme IV, a new approach to numerical convection. *J. Comput. Phys.*, 23: 276-299, 1977.

[14] P. COLELLA AND P. WOODWARD. The piecewise-parabolic method (PPM) for gas-dynamical simulations. *J. Comput. Phys.*, 54: 174-201, 1984.

[15] A. HARTEN, B. ENGQUIST, S. OSHER AND S. R. CHAKRAVARTHY. Uniformly high order accuracy essentially non-oscillatory schemes III. *J. Comput. Phys.*, 71: 231-303, 1987.

[16] X.D. LIU, S. OSHER AND T. CHAN. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115: 200-212, 1994.

[17] G.S. JIANG AND C.W. SHU. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126: 202-228, 1996.

[18] K. XU. Dissipative mechanism in Godunov method. *Computational Fluid Dynamics for the 21st Century*, M. Hafez, K. Morinishi, and J. Periaux (Eds), 309-321, 2001.

[19] K. XU, M.L. MAO AND L. TANG. A multidimensional gas-kinetic BGK scheme for hypersonic viscous flow. *J. Comput. Phys.*, 203: 405-421, 2005.

[20] J. QUIRK. A contribution to the great Riemann solver debate. *Int. J. Num. Met. Fluids*, 18: 555-574, 1994.

[21] R.J. LEVEQUE. Nonlinear conservation laws and finite volume methods for astrophysical fluid flow. *Computational Methods for Astrophysical Flow*, Edited by O. Steiner and A. Gautschy, New York: Springer-Verlag, 1998.

[22] H.Z. TANG AND T.G. LIU. A note on the conservative schemes for the Euler equations. *J. Comput. Phys.*, 218: 451-459, 2006.

[23] S. KUDRIAKOV AND W.H. HUI. On a new defect of shock-capturing methods. *J. Comput. Phy.*, 227: 2105-2117, 2008.

Chapter 4

1-D Flow Computation Using the Unified Coordinates

4.1 Gas Dynamics Equations Based on the Unified Coordinates

The gas dynamics equations in Eulerian coordinates (t, x) are written in conservation PDE form as

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ \rho e \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(\rho e + p) \end{pmatrix} = \mathbf{0}, \tag{4.1}$$

where

$$e = \frac{1}{2} u^2 + \frac{1}{\gamma - 1} \frac{p}{\rho}.$$

Under the transformation to classical Lagrangian coordinates

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = u \mathrm{d}\lambda + \frac{1}{\rho} \mathrm{d}\xi, \end{cases} \tag{4.2}$$

(4.1) become

$$\frac{\partial}{\partial \lambda} \begin{pmatrix} 1/\rho \\ u \\ e \end{pmatrix} + \frac{\partial}{\partial \xi} \begin{pmatrix} -u \\ p \\ up \end{pmatrix} = \mathbf{0}. \tag{4.3}$$

The system of equations (4.3) is seen to be in conservation PDE form.

We now introduce the transformation to the unified coordinates (λ, ξ) as follows:

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = U \mathrm{d}\lambda + A \mathrm{d}\xi, \end{cases} \tag{4.4}$$

where $U(\lambda, \xi)$ is an arbitrary function. It can easily be shown that

$$\frac{\partial \xi}{\partial t} + U \frac{\partial \xi}{\partial x} = 0. \tag{4.5}$$

This means that the coordinate ξ is invariant following a *pseudo-particle* whose velocity is U. Consequently, computational cells move with the pseudo-particles, instead of the fluid particles as in Lagrangian coordinates. As we can see from the above transformation, the case $U = 0$ gives us the Eulerian coordinates in which ξ is independent of the flow; in particular, $\xi = x$ if $A = 1$. On the other hand, the case $U = u$ gives the Lagrangian coordinates, since the pseudo-particles move with the fluid particles. The classical Lagrangian coordinate corresponds to the special case when $A = 1/\rho$ in addition to $U = u$. Thus the unified coordinate ξ introduced by (4.4) with $U = u$ may be regarded as a generalized Lagrangian coordinate; it has the flexibility in choosing the stretching factor A. One advantage of this unified coordinate system is its ability to show the effect of coordinate systems on the behaviour of the solution by simply varying the parameter U from 0 to u.

Under transformation (4.4), the gas dynamics equations (4.1) become

$$\frac{\partial \boldsymbol{E}}{\partial \lambda} + \frac{\partial \boldsymbol{F}}{\partial \xi} = \boldsymbol{0},\tag{4.6}$$

where

$$\boldsymbol{E} = \begin{pmatrix} \rho A \\ \rho A u \\ \rho A e \\ A \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} (u - U)\rho u \\ (u - U)\rho u^2 + p \\ (u - U)\rho u e + u p \\ -U \end{pmatrix}.$$

The last equation in (4.6) is the compatibility condition of the differential relation (4.4) for dx. We note that like the Eulerian system (4.1), the UC system (4.6) and its Lagrangian special case (4.3) are also in conservation PDE form. It is well-known that Eulerian system (4.1) is hyperbolic in t-direction, and so is the Lagrangian system (4.3) in λ-direction. We now prove that the UC system (4.6) is also hyperbolic in λ-direction, despite the fact that the transformation involves the unknown function v. We choose $\boldsymbol{Q} = (p, \rho, u, A)^{\mathrm{T}}$ to be the vector of state variables, then the eigenvalues and corresponding eigenvectors are [1]

$$\begin{cases} (1) & \sigma = 0, \\[2mm] (2) & \sigma = \dfrac{u - U}{A}, \\[2mm] (3) & \sigma_{\pm} = \dfrac{(u - U) \pm a}{A}, \end{cases}\tag{4.7}$$

where $a = \left(\dfrac{\gamma p}{\rho}\right)^{1/2}$ is the speed of sound. The corresponding right eigenvectors

are

$$\begin{cases} \boldsymbol{r}_1 = (0,0,0,1)^{\mathrm{T}}, \\ \boldsymbol{r}_2 = (0,1,0,0)^{\mathrm{T}}, \\ \boldsymbol{r}_\pm = \left(1, a^{-2}, \pm\dfrac{1}{a\rho}, \mp\dfrac{U/u}{a\rho\sigma_\pm}\right)^{\mathrm{T}}. \end{cases} \tag{4.8}$$

Since these eigenvectors are evidently linearly independent, we conclude that system (4.6) is **hyperbolic in λ-direction**.

Since the UC system (4.6) and its special case (4.3) are, like their Eulerian counterpart (4.1), both hyperbolic and in conservation form, these systems can be solved similarly to the Eulerian one. In particular, any shock-capturing methods developed for conservation laws (as summarized in Chapter 3) apply.

We further note that Lagrangian gas dynamics equations (4.3) and the Eularian ones (4.1) are both strictly hyperbolic, having distinct eigenvalues $(-\rho a, 0, +\rho a)$ and $(u-a, u, u+a)$, respectively. It is remarkable that Wagner[2] proved that they are theoretically equivalent to each other, meaning the existence of a one-to-one map between the two sets of weak solutions obtained by using the two systems. This important result does not, however, carry over to the case of multi-dimensional flow, as shown in Chapter 8.

Computationally, Lagrangian method is superior to the Eulerian in that contact discontinuities are sharply resolved in Lagrangian computation as they coincide with the coordinate lines (cell interfaces) and hence avoid the cell averaging errors across them, whereas contacts are badly smeared in Eulerian computation because they do not align with the fixed Eulerian coordinates and hence errors arise due to cell averaging across the contacts. We shall show that Lagrangian computation can be further improved by using the shock adaptive Godunov scheme to replace the classical Godunov scheme.

4.2   Shock-Adaptive Godunov Scheme

As we pointed out earlier, the only source of error in the Godunov scheme arises from averaging the conservative flow quantities over a computational cell when representing the flow in all cells by piecewise constant states. To minimize this error, we distinguish three cases.

(1) In smooth flow regions, the error due to cell averaging is small and can be

reduced by using a high resolution schemes, e.g., FCT[3], MUSCL[4], ENO[5], or WENO[6] etc..

(2) On crossing a contact discontinuity the error is large, as in Eulerian computation. However, it is minimized using Lagrangian or the unified coordinates (generalized Lagrangian). Since the contact coincides with the Lagrangian coordinate line and hence with a cell interface, there is no error arising from averaging across it.

(3) On crossing a shock discontinuity, we shall use the shock-adaptive Godunov scheme to replace the classical Godunov scheme. In this new scheme, there also is no error arising from averaging across the shock. This is described as follows[1, 7, 8].

The basic idea of the shock-adaptive Godunov scheme consists of splitting a *shock-cell*, i.e., a computational cell containing a shock wave, along the trajectory of the shock. The split shock-cell becomes two sub-cells: one entirely upstream of the shock and the other entirely downstream. In this way, the cell averaging procedure across the shock discontinuity, and the errors associated with it, are avoided, resulting in infinite shock resolution. The fictitious cell boundary separating the two sub-cells and moving through the regular mesh at the local speed of the shock shall be called a *partition*. With this abstraction, the two sub-cells and the other (regular) cells can be treated on an equal footing in the Godunov scheme (recall that Godunov scheme applies to non-uniform cells as well as uniform ones).

The shock-adaptive Godunov scheme is complemented by a shock-cell splitting criterion which triggers the splitting of a shock-cell in the presence of a shock in the flow field. In the approach considered here, a shock-cell is split if the pressure jump across an elementary shock wave, as obtained from the exact solution to the Riemann problem, is larger than some critical shock strength threshold, say if

$$\frac{p_u}{p_d} < \delta_{\text{Shock}}, \quad 0 < \delta_{\text{Shock}} < 1, \tag{4.9}$$

where the subscripts d and u refer to the downstream and upstream flow states, respectively. This desirable self-adaptivity feature permits for the automatic detection of a shock wave without a priori knowledge of its position nor its existence.

The value of $\delta_{\text{Shock}} = 1$ corresponds to splitting all cells along elementary shock waves, whereas $\delta_{\text{Shock}} = 0$ corresponds to no splitting and the scheme simply reduces to the classical Godunov scheme. In our calculations, $\delta_{\text{Shock}} = 0.6$,

corresponding to the entropy jump of 16%, was found to be satisfactory.

We note that the shock-adaptive Godunov scheme is, in effect, doing a shock fitting in the overall shock-capturing environment, using the information from the exact Riemann solution with no extra cost. We also note that a similar idea was proposed in [9] for Eulerian computation. However, it did not work well (for example, see [8]), due to convolution of averaging errors across a shock and those across a contact. In Lagrangian computation, contact resolution is taken care of by its alignment with cell interface, leaving only the shock resolution to be handled by the shock- adaptive Godunov scheme.

4.3   The Use of Entropy Conservation Law for Smooth Flow Computation

Now with the use of the shock-adaptive Godunov scheme in the Lagrangian coordinates, the errors arising from averaging across both discontinuities—shocks and contacts—are avoided, and the numerical computation of unsteady flow needs be done only in the smooth flow regions. The utmost importance of the conservation form of the governing equations[10] for the correct capturing of shocks is no longer necessary as shocks are now fitted exactly. It is therefore possible to freely rewrite the governing equations in any other form that is advantageous for computing smooth flow. For this purpose, an entropy-conserving reformulation of the governing equations in the smooth flow region is proposed.

One motivation for the use of an alternative set of governing equations arises from the desire to eliminate the overshoots commonly observed in Lagrangian computation in the vicinity of contacts (see Refs. [1, 11], also Figures 4.3 and 4.9), despite the fact that in the case $U = u$ the flow quantities are not averaged across a slip line. Another motivation is to compute entropy accurately.

Since the flow is isentropic along a path line in a smooth flow region, the energy equation (the 3rd equations of (4.3) and (4.6)) can be replaced by the law of conservation of entropy,

$$\frac{\partial S}{\partial \lambda} = 0 \tag{4.10}$$

along a path line, where $S = p/\rho^\gamma$. Accordingly, (4.3) and (4.6) are simplified. For example, (4.3) becomes

$$\frac{\partial}{\partial \lambda}\begin{pmatrix} 1/\rho \\ u \\ p/\rho^\gamma \end{pmatrix} + \frac{\partial}{\partial \xi}\begin{pmatrix} -u \\ p \\ 0 \end{pmatrix} = \mathbf{0}. \tag{4.11}$$

We caution that the law of conservation of entropy (4.10), and hence (4.11), applies only in regions of smooth flow, i.e., upstream and downstream of the shock. Across a shock discontinuity, the exact entropy jump is imposed using the Riemann solution.

To summarize, our UC method for 1-D flow computation consists of using

(1) the generalized Lagrangian coordinates given by (4.4), or its special case, the classical Lagrangian coordinates given by (4.2);

(2) the shock-adaptive scheme in place of the classical Godunov scheme;

(3) Equations (4.11) instead of (4.3) for computing flow in the smooth flow regions.

Accordingly, UC gives exact entropy everywhere, fits shocks exactly and resolves contacts sharply. It will be shown in Section 4.5 that all the defects listed in Section 3.7, except the defect (7), can be cured or avoided by the UC method using the classical Lagrangian coordinate plus shock adaptive Godunov scheme for (4.11). To avoid the defect (7), which has a large variation of density, it is necessary to use (4.6) (in place of (4.3)) with its energy equation replaced by (4.10).

4.4   The Unified Computer Code

With the Riemann solution given in Appendix A, where we have used a parameter h such as $U = hu$, and the shock-adaptive Godunov scheme explained in Section 4.2, we are now in the position to describe the numerical procedure.

A pseudo-particle in the physical space marching in the flow direction of a fluid particle corresponds to a cell marching in the λ-direction in the computational (λ, ξ) space. The superscript n refers to the marching time step number and the subscript i refers to the cell index number on a time line $\lambda = \mathrm{const.}$.

The following steps describe the numerical procedure of the shock-adaptive Godunov scheme upgraded by MUSCL. This scheme is applied only for $U = u$ (Lagrangian coordinates). By putting the shock-detection parameter δ_{Shock} equal to zero (no cells are splitted) the described scheme reduces to the classical Godunov-MUSCL scheme, and it can then be applied for $0 \leqslant U \leqslant u$.

Step 1: initialization at λ^0. Assume the initial conditions of a flow problem are given at $t = 0$ ($\lambda = 0$) in the physical space. Then a ξ-coordinate mesh is laid on the x line (for instance, we take ξ equal to x) with

$$\xi = \xi_0, \xi_1, \xi_2, \cdots, \xi_m. \tag{4.12}$$

Hence the flow variables $\boldsymbol{Q}_i^0 = (p_i^0, \rho_i^0, u_i^0, A_i^0)(i = 1, 2, \cdots, m)$ are known as initial conditions. Subsequently, $\boldsymbol{E}_i^0$ are available for all i, $i = 1, 2, \cdots, m$.

Step 2: construction of interface fluxes. With all $\boldsymbol{E}_i^n$ and $\boldsymbol{Q}_i^n$ known at step n ($n = 0, 1, 2, \cdots$), the $\boldsymbol{Q}$ data are first upgraded or reconstructed via MUSCL procedure by nonlinear interpolation in the following way. Let f be any of the above variables p, ρ, u or A, then in the cell i and $i + 1$, we assume linearly distributed states in both cells, and use nonlinear extrapolation to determine cell interface flow variables as follows:

$$f_r = f_{i+1} - 0.5(f_{i+2} - f_{i+1})\phi(r^+), \tag{4.13}$$

$$r^+ = \frac{f_{i+1} - f_i}{f_{i+2} - f_{i+1}}, \tag{4.14}$$

$$f_l = f_i + 0.5(f_i - f_{i-1})\phi(r^-), \tag{4.15}$$

$$r^- = \frac{f_{i+1} - f_i}{f_i - f_{i-1}}, \tag{4.16}$$

where $\phi(r) = \max(0, \min(1, r))$ is the minmod limiter and subscripts r and l correspond to the right and the left state, respectively. We emphasize that the MUSCL upgrading does not apply across shocks: the shocks are already captured exactly by the shock-adaptive Godunov scheme based on the Lagrangian coordinates, there is no room for further improvement by upgrading.

A local Riemann problem is solved at all cell interfaces, including partitions, for all adjacent cell pairs i and $i + 1$. A Newton iterative method is employed (see Use reference [12] of chapter 3) for finding the values of pressure p^* and velocity u^* at the cell interface $\xi_{i+1/2}$. The numerical fluxes are obtained in the following.

Let us consider two cells, i and $i + 1$, with partition between them having the slope $\dfrac{\mathrm{d}\xi}{\mathrm{d}\lambda} = \sigma_{i+1/2}^n > 0$(Figure 4.1) (The case $\sigma_{i+1/2}^n < 0$ can be derived similarly). Applying the divergence theorem to the cells i and $i + 1$, we obtain

$$\Delta\xi_i^{n+1}\boldsymbol{E}_i^{n+1} = \Delta\xi_i^n\boldsymbol{E}_i^n + \sigma_{i+1/2}^n \cdot \Delta\lambda \cdot \boldsymbol{E}_{i+1/2}^*$$

$$-\Delta\lambda \cdot (\boldsymbol{F}(p^*_{i+1/2}, u^*_{i+1/2}) - \boldsymbol{F}(p^*_{i-1/2}, u^*_{i-1/2})), \qquad (4.17)$$

$$\Delta\xi^{n+1}_{i+1}\boldsymbol{E}^{n+1}_{i+1} = \Delta\xi^n_{i+1}\boldsymbol{E}^n_{i+1} - \sigma^n_{i+1/2} \cdot \Delta\lambda \cdot \boldsymbol{E}^n_{i+1}$$

$$-\Delta\lambda \cdot (\boldsymbol{F}(p^*_{i+3/2}, u^*_{i+3/2}) - \boldsymbol{F}(p^n_{i+1}, u^n_{i+1})), \qquad (4.18)$$
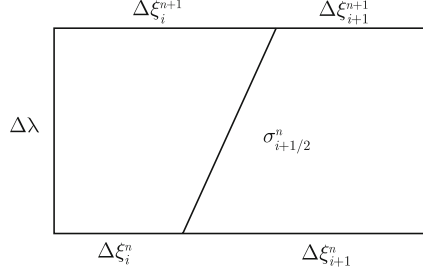
Figure 4.1   Computational cells i and $i+1$ with
partition in between having slope $\sigma^n_{i+1/2}$

where $\boldsymbol{E}^*_{i+1/2}$ is the averaged conserved flow state along the inner (downstream) side of the shock having speed $\sigma^n_{i+1/2}$. Using $\Delta\xi^{n+1}_i = \Delta\xi^n_i + \sigma^n_{i+1/2} \cdot \Delta\lambda$ and $\Delta\xi^{n+1}_{i+1} = \Delta\xi^n_{i+1} - \sigma^n_{i+1/2} \cdot \Delta\lambda$, the above equations can be written in the form

$$\boldsymbol{E}^{n+1}_i = \boldsymbol{E}^n_i - \frac{\Delta\lambda}{\Delta\xi^{n+1}_i} \left(\boldsymbol{F}(p^*_{i+1/2}, u^*_{i+1/2}) \right.$$

$$\left. +\sigma^n_{i+1/2} \cdot (\boldsymbol{E}^n_i - \boldsymbol{E}^*_{i+1/2}) - \boldsymbol{F}(p^*_{i-1/2}, u^*_{i-1/2}) \right), \qquad (4.19)$$

$$\boldsymbol{E}^{n+1}_{i+1} = \boldsymbol{E}^n_{i+1} - \frac{\Delta\lambda}{\Delta\xi^{n+1}_{i+1}} \left(\boldsymbol{F}(p^*_{i+3/2}, u^*_{i+3/2}) - \boldsymbol{F}(p^n_{i+1}, u^n_{i+1}) \right), \qquad (4.20)$$

which reproduce the numerical fluxes

$$(\Im^{\pm})^n_i = \begin{cases} \boldsymbol{F}(p^n_i, u^n_i), & \pm\sigma^n_{i\pm1/2} < 0, \\ \boldsymbol{F}(p^*_{i\pm1/2}, u^*_{i\pm1/2}) + \sigma^n_{i\pm1/2}(\boldsymbol{E}^n_i - \boldsymbol{E}^*_{i\pm1/2}), & \pm\sigma^n_{i\pm1/2} \geqslant 0 \end{cases} \qquad (4.21)$$

are evaluated at $\xi_{i\pm1/2}$, where $\sigma^n_{i\pm1/2}$ are the slopes $d\xi/d\lambda$ of the interface $\xi_{i\pm1/2}$ at λ^n and $\boldsymbol{E}^*_{i\pm1/2}$ is the averaged conserved flow state along the inner (downstream) side of the shock relative to the shock-subcell being updated. The speeds $\sigma^n_{i\pm1/2}$ of the partitions are updated based on the speeds of the corresponding elementary shock waves in the local Riemann problems. The presence of a new shock is tested at all cell interfaces. If the splitting criterion is satisfied, a new partition is introduced to account for the incoming shock and the cell is split.

Step 3: determination of the step size $\boldsymbol{\Delta\lambda^n}$. To satisfy the stability condition of the scheme, the step size is determined as the minimum of the step

sizes

$$\Delta\lambda^n = \frac{\text{CFL}}{\max_i \left\{ \frac{|\sigma_{i\pm1/2}^n|}{\Delta\xi_i} \right\}} \tag{4.22}$$

for the regular cells and

$$\Delta\lambda^n = \min_i \left\{ \frac{\Delta\xi_i}{|\sigma_{i+1/2}^n| + |\sigma_{i-1/2}^n|} \right\} \tag{4.23}$$

for the shock-subcells which represents the intersection point of the $\pm$ waves in the shock-subcell, or equivalently, the intersection point of the incoming elementary wave with the opposing cell interface, also an elementary wave.

For an elementary rarefaction wave, $\sigma_{i\pm1/2}^n$ is replaced by the speed of the leading Mach line (fastest characteristic). For an elementary shock wave, $\sigma_{i\pm1/2}^n$ is replaced by the shock speed, since any disturbance propagating in the shock direction is bounded by the shock, with the flow upstream of the shock being undisturbed.

Step 4: advancement of the average cell states from λ^n to $\lambda^{n+1} = \lambda^n + \Delta\lambda^n$. The average states E_i^{n+1} for all non-terminating cells, with domain

$$\{(\lambda,\xi) \in \mathbf{R}^2 | \lambda^n < \lambda < \lambda^{n+1}, \xi_{i-1/2}^n + \sigma_{i-1/2}^n(\lambda - \lambda^n) < \xi < \xi_{i+1/2}^n + \sigma_{i+1/2}^n(\lambda - \lambda^n)\} \tag{4.24}$$

are obtained at λ^{n+1} using the difference equation

$$E_i^{n+1} = E_i^n - \frac{\Delta\lambda^n}{\Delta\xi_i^{n+1}}((\Im^+)_i^n - (\Im^-)_i^n) \tag{4.25}$$

with the composite fluxes given by (4.21). The cell width $\Delta\xi_i^{n+1}$ of a shock-subcell is updated using

$$\Delta\xi_i^{n+1} = \Delta\xi_i^n + (\sigma_{i+1/2}^n - \sigma_{i-1/2}^n)\Delta\lambda^n \tag{4.26}$$

prior to advancing the conserved flow state. For a stationary coordinate contact line, $\sigma_{i+1/2}^n = 0$; for a partition, $\sigma_{i+1/2}^n = \sigma_{\text{Shock}}^n$, as obtained from the solution to the local Riemann problem. For a regular rectangular cell, the above difference equation reduces to the familiar updating formula used in the classical Godunov scheme.

Step 5: decoding to get Q_i^{n+1}. The decode procedure is straightforward. From the expression for $E = (e_1, e_2, e_3, e_4)^{\text{T}}$, where $e_1 = \rho A$, $e_2 = \rho Au$, $e_3 = \rho Ae$

or $e_3 = \dfrac{p}{\rho^{\gamma}}$ (for shock-adaptive Godunov scheme) and $e_4 = A$, we have

$$u = \frac{e_2}{e_1}, \tag{4.27}$$

$$\rho = \frac{e_1}{e_4}, \tag{4.28}$$

$$p = \begin{cases} \left(\dfrac{e_3}{e_1} - \dfrac{u^2}{2}\right)(\gamma - 1)\rho, & U \neq u, \\[2ex] e_3 \cdot \rho^{\gamma}, & U = u, \quad \text{for shock-adaptive Godunov scheme,} \end{cases} \tag{4.29}$$

$$A = e_4. \tag{4.30}$$

Step 6: Elimination of the terminating shock-subcells at $\boldsymbol{\lambda^{n+1}}$ (Only for shock-adaptive Godunov scheme). The shock-subcells terminating at λ^{n+1} are eliminated by removing the corresponding partitions and the upstream shock-subcells. This renders the downstream subcell a regular cell at λ^{n+1}.

Step 7: Mapping results onto the physical space. To map results from computational space onto the physical space, trapezoidal integration is applied to get

$$x_i^{n+1} = x_i^n + 0.5 \cdot \Delta\lambda \cdot (U_i^n + U_i^{n+1}). \tag{4.31}$$

This step in the program is optional since the grid in the physical space does not participate in the computation; all computations are carried out in the transformed (λ, ξ) space. It is obvious that we will have some errors during mapping of computational results onto physical domain, but the errors are small as seen in the test examples.

Now the numerical procedure for advancing one time step is completed. To march forward further, one goes back to Step 2 and repeats Steps 2~6.

A detailed computer code is given in Appendix B, where we have used a parameter h such that $U = hu$. It can be used to perform an Eulerian computation by specifying $h = 0$, or classical Lagrangian computation by specifying $h = 1$ and $A = 1/\rho$, or the UC computation. The code may also be used to do computer experiments to see the effects of h on the flow.

4.5 Cure of Defects of Eulerian and Lagrangian Computation by the UC Method

We now explain how to use the UC method to cure, or avoid, the following defects of the Eulerian computation:

(1) Contact smearing;

(2) Slow moving shocks;

(3) Sonic point glitch;

(4) Start-up errors;

(5) Low pressure flow;

(6) Wall-overheating;

(7) Strong rarefaction waves.

Note that defects (4) to (7) are also shared in the classical Lagrangian computation.

To do the UC computation for cases (1)$\sim$(6) it is adequate, as we shall do, to use $U = u$ and $A = 1/\rho$. For case (7), the density variation is great, and it is necessary to use the coordinates given by (4.4) with $U = u$, while leaving A to be determined as part of the solution. This allows us to have a uniform mesh initially by specifying $A = 1.0$ at $\lambda = 0$.

4.5.1   Contact smearing

Figure 4.2 shows the Eulerian computation for the Riemann problem with initial data,

$$(\rho, u, p) = \begin{cases} (5.9924, 19.5975, 460.894), & 0 \leqslant x < 1.0, \\ (5.99242, -6.19633, 46.0950), & 1.0 \leqslant x \leqslant 2.0, \end{cases} \quad (4.32)$$

Figure 4.2   Contact smearing for Riemann problem (4.32): Eulerian computation employing Godunov MUSCL method with minmod limiter

which consists of two shocks and a contact line in between, where ρ is density and i internal energy. While the shocks are resolved to within 2 cells, which is

a fortuitous fact due to characteristics converging to the shocks, the contact line is badly smeared; the smearing actually increases with the number of marching time steps. As pointed out earlier, the numerical smearing of the contact is due to non-alignment of Eulerian coordinate line, $x =$const., with the contact line. It is inherent in the Eulerian computation and cannot be substantially reduced, except with artificial techniques which are not always reliable.

Figure 4.3 shows the classical Lagrangian computation. We see that the shock resolution is similar to Eulerian computation, and that the contact is more sharply resolved. We also note the overshoot (undershoot) near the contact.

Figure 4.4 shows the UC computation, i.e., classical Lagrangian plus shock adaptive Godunov scheme and the use of equation (4.11). It reproduces the exact analytic solution, as predicted.
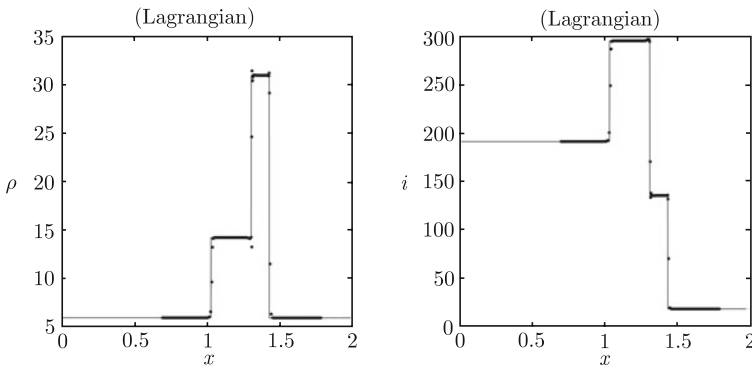
Figure 4.3   Contact smearing for Riemann problem (4.32): Lagrangian computation employing Godunov MUSCL method with minmod limiter

4.5.2   Slow moving shock

In computing an isolated shock, if the shock is slow-moving the disturbances due to the shock is of low frequencies (long waves). The numerical dissipation near the shock is small for slow-moving shocks and may not be sufficient to damp the long waves down. The test case is the Riemann problem with the initial data,[12]

$$(\rho, u, p) = \begin{cases} (3.86, 0.0, 10.33), & 0 \leqslant x < 1.0, \\ (1.0, -2.63, 1.0), & 1.0 \leqslant x \leqslant 2.0. \end{cases} \tag{4.33}$$
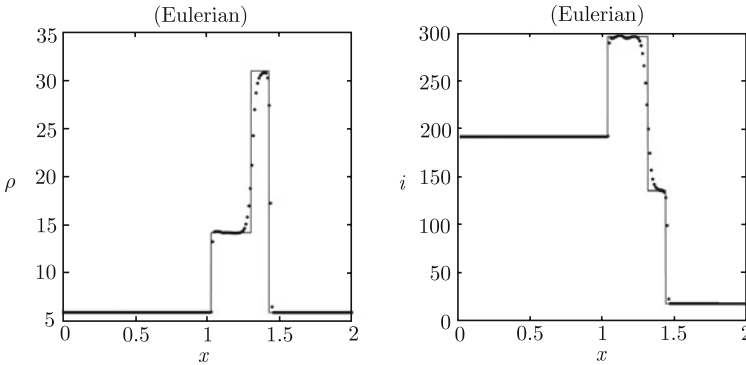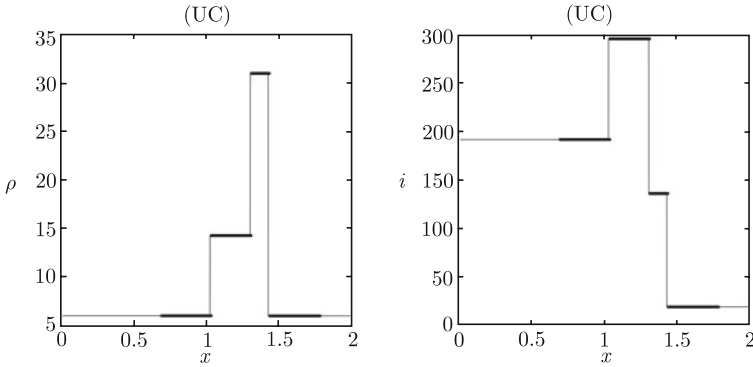
Figure 4.4   Contact smearing for Riemann problem (4.32): UC computation employing
shock adaptive Godunov MUSCL method with minmod limiter

In these cases, the strong acoustic and entropy waves generated will give rise to some oscillations in the pressure as well as density[12], where the shock speed is 0.11.

In the Lagrangian coordinates the speed of a shock wave is always greater (in magnitude) than the characteristic speed, $a\rho$, and hence the shock cannot be slow-moving. Therefore, the errors due to slow-moving shocks that occur in Eulerian computation will never occur in Lagrangian computation. This is clearly seen by comparing the Lagrangian results in Figure 4.6 with the Eulerian ones in Figure 4.5 for the same flow problem. With UC method, both errors are eliminated, as shown in Figure 4.7.
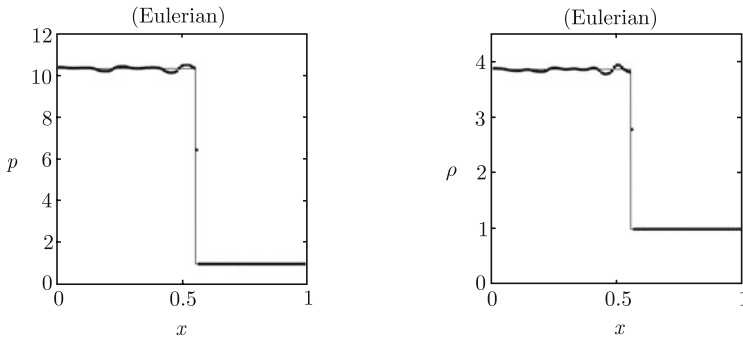
Figure 4.5   Slowly moving shock wave for Riemann problem (4.33): Eulerian
computation employing Godunov MUSCL method with minmod limiter

Figure 4.6   Slowly moving shock wave for Riemann problem (4.33): Lagrangian

computation employing Godunov MUSCL method with minmod limiter
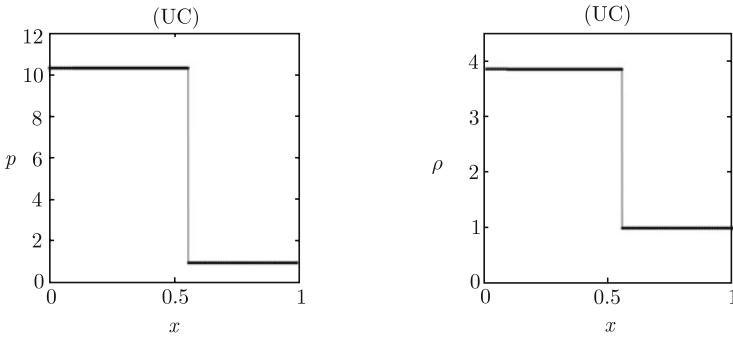
Figure 4.7   Slowly moving shock wave for Riemann problem (4.33): UC computation

employing shock adaptive Godunov MUSCL method with minmod limiter

4.5.3   Sonic point glitch

In Eulerian computation, many shock-capturing methods experience difficulties in the resolution of rarefaction waves. We consider the Riemann problem from[12] with initial data

$$(\rho, u, p) = \begin{cases} (1.0, 0.75, 1.0), & 0 \leqslant x < 1.0, \\ (0.125, 0.0, 0.1), & 1.0 \leqslant x \leqslant 2.0. \end{cases} \tag{4.34}$$

Figure 4.8, which uses the first order Godunov method, shows a "glitch" in the rarefaction waves region. This phenomenon is associated with the fact that at the point where the "glitch" situates the flow is sonic, hence the characteristic velocity for the genuinely nonlinear wave field is zero (see [13] and [14] for an explanation). In Lagrangian computation, the corresponding characteristic velocities are

$a\rho$ and $-a\rho$, and will never become zero. Consequently, the sonic point glitch can never occur in Lagrangian computation, as demonstrated in Figure 4.9 which was computed using the same Godunov method but based on the Lagrangian equations. We note that contact overheating (see also 4.5.6 Wall-overheating below) is greatly smeared in Eulerian computation (Figure 4.8), but is more pronounced in Lagrangian computation (Figure 4.9). The overheating is completely eliminated using UC method, as shown in Figure 4.10.
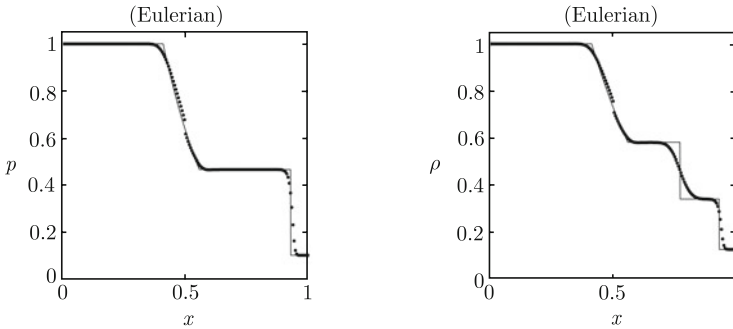
Figure 4.8   Sonic glitch test for Riemann problem (4.34): Eulerian computation
employing Godunov MUSCL method with minmod limiter
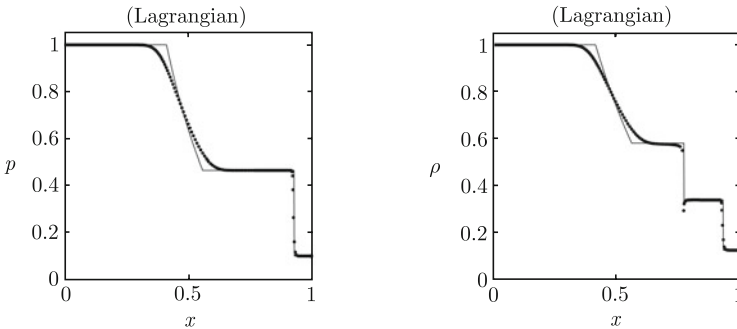
Figure 4.9   Sonic glitch test for Riemann problem (4.34): Lagrangian computation
employing Godunov MUSCL method with minmod limiter

4.5.4   Start-up errors[12]

Numerical errors are often observed in using conventional shock-capturing methods to compute the propagation of an isolated shock, especially when the initial data are chosen as a sharp discontinuity. These methods require averaging across a
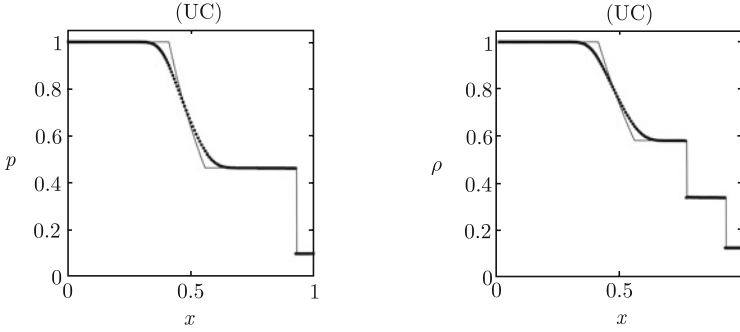
Figure 4.10   Sonic glitch test for Riemann problem (4.34): UC computation employing
shock-adpative Godunov MUSCL method with minmod limiter

shock, leading to a smeared shock profile and, in the process, generating acoustic
and entropy waves. While acoustic waves dissipates quickly the entropy waves
remain observable.

This is again the same Riemann problem as the slowly moving shock test (4.33),
of an isolated shock, initially located at $x = 1.0$, moving at a speed equal to 0.92.
Eulerian computations are shown in Figure 4.11, whereas Lagrangian computations
are shown in Figure 4.12, both using the conventional Godunov-MUSCL shock-
capturing method. In both cases, the start-up errors occur in the form of entropy
waves in the density profile and in the entropy profile.
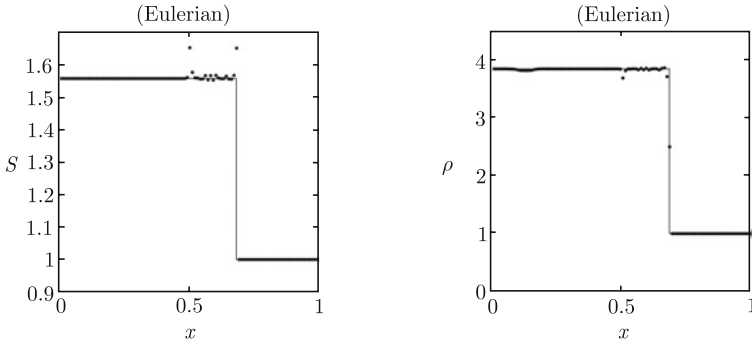
Figure 4.11   Start-uperror test for Riemann problem (4.33): Eulerian
computationemploying Godunov MUSCL method with minmod limiter

In our method, the shock is fitted using exact Riemann solution and is taken
to be a cell interface, avoiding cell-averaging across it. Consequently, the start-up
errors are avoided, as shown in Figure 4.13.
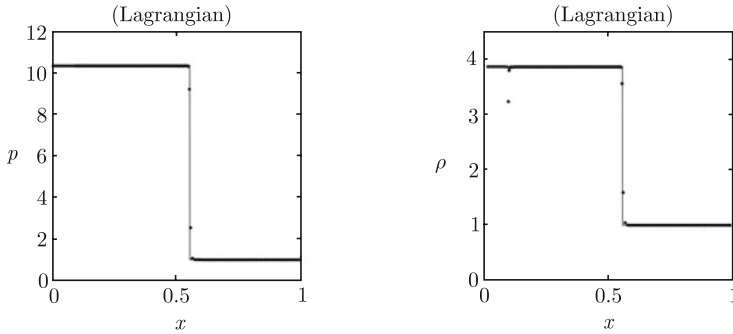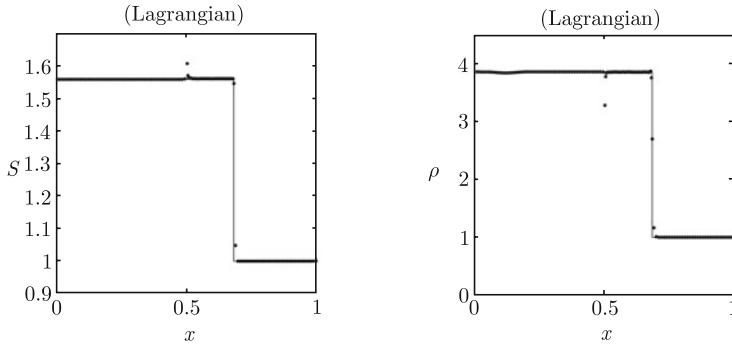
Figure 4.12   Start-up error test for Riemann problem (4.33): Lagrangian computation employing Godunov MUSCL method with minmod limiter
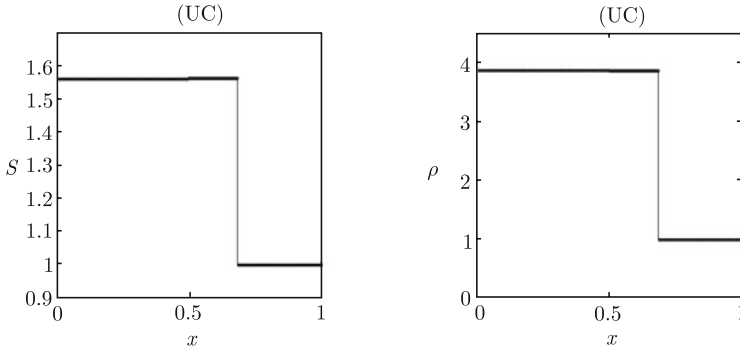
Figure 4.13   Start-up error test for Riemann problem (4.33): UC computation employing Godunov MUSCL method with minmod limiter

4.5.5   Low pressure flow

A difficulty occurs in some shock-capturing methods with a linearized Riemann solution that use conservation formulation[15]. In this formulation the total energy, density and momentum are the conserved variables, but the pressure p is computed from the total energy after subtracting the kinetic energy determined from the momentum and density. In regions of high speed flow, the kinetic energy dominates the internal energy and inaccuracies in the conserved quantities can easily lead to the kinetic energy exceeding the total energy, causing computational breakdown in the form of pressure being negative. It was shown in [15] that this will not happen with the (first order) Godunov method for the Euler equations but can happen if a linearized Riemann solution is used.

The difficulty of negative pressure appearing will never happen in UC method in which, for the smooth flow regions, pressure is determined from the constancy of the entropy and the conserved quantity $1/\rho$ (see (4.11)), and can never become negative. On crossing a shock, pressure is increased as determined by the exact Riemann solution and cannot become negative either. Figure 4.14 shows the UC computation for the expansion flow due to a piston withdrawing at speed equal to 5 from a gas at rest with a unit pressure and density. It is seen that even when the flow near the piston is almost vacuum, e.g., the pressure is $p = 1.6 \times 10^{-7}$, the flow is computed without any difficulty.
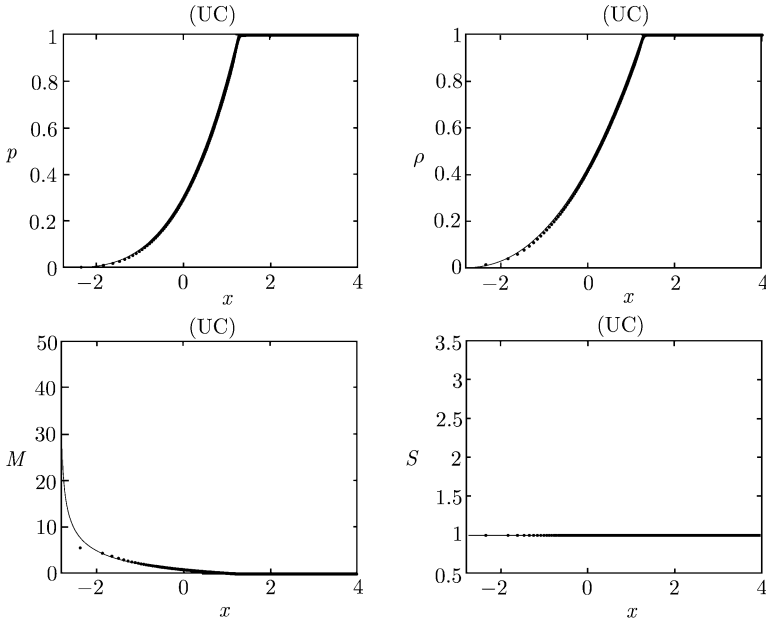
Figure 4.14   Sudden expansion of a gas; UC computation (dots) compared with exact solution (solid line) at $t = 1.0$

4.5.6   Wall-overheating

This notorious phenomenon of wall-overheating was first discovered by von Neumann[16], and has since been a hot topic of research[12,17−28]. It is now realized[12] that not only sudden compression of a gas due to shock reflection from a solid wall, but also sudden expansion due to an abrupt withdrawal of a piston from the gas, will produce this wall-overheating. Most conventional shock-capturing methods in Eulerian or Lagrangian computation can accurately predict pressure and ve-

locity in these situations, while under-predicting density, thus over-predicting the temperature near the wall, hence the name.

Consider the well known Noh's problem: the initial condition is a uniform flow with velocity $u = -1$, density $\rho = 1$ and pressure $p = 0$ in a unit domain $0 < x < 1$. This flow is reflected at time $t = 0$ from a solid wall situated at $x = 0$, and we want to compute the flow after the reflection. Eulerian computations of pressure, p velocity u, density ρ and i temperature T are shown in Figure 4.15, whereas the corresponding, Lagrangian computations are shown in Figure 4.16, both results are compared with the exact solutions. We see from these figures that the conventional Godunov-MUSCL method in Eulerian or Lagrangian coordinates predicts wall pressure and velocity correctly, but it fails to predict the density (and hence also temperature and entropy) correctly near the wall. The failure is worse in the Lagrangian computation. Figure 4.17 presents computational results for the same problem using UC method, i.e., (4.11) with the shock-adaptive Godunov scheme. The wall overheating is seen completely eliminated; indeed, the exact solution is reproduced by our method.

Figure 4.15   Noh's problem; Eulerian computation (dots) employing Godunov-MUSCL scheme, compared with exact solution (solid line) at $t = 0.6$
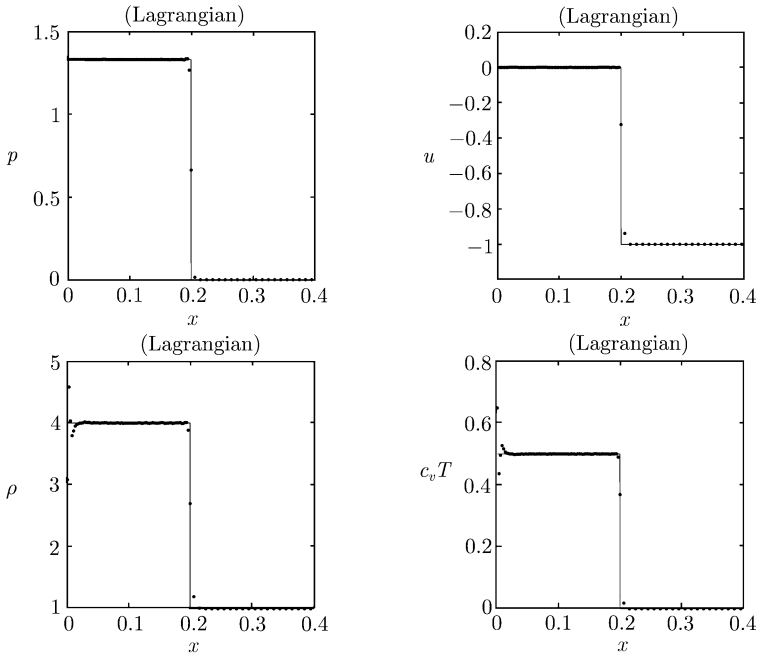
Figure 4.16   Noh's problem; Lagrangian computation (dots) employing
Godunov-MUSCL scheme, compared with exact solution (solid line) at $t = 0.6$
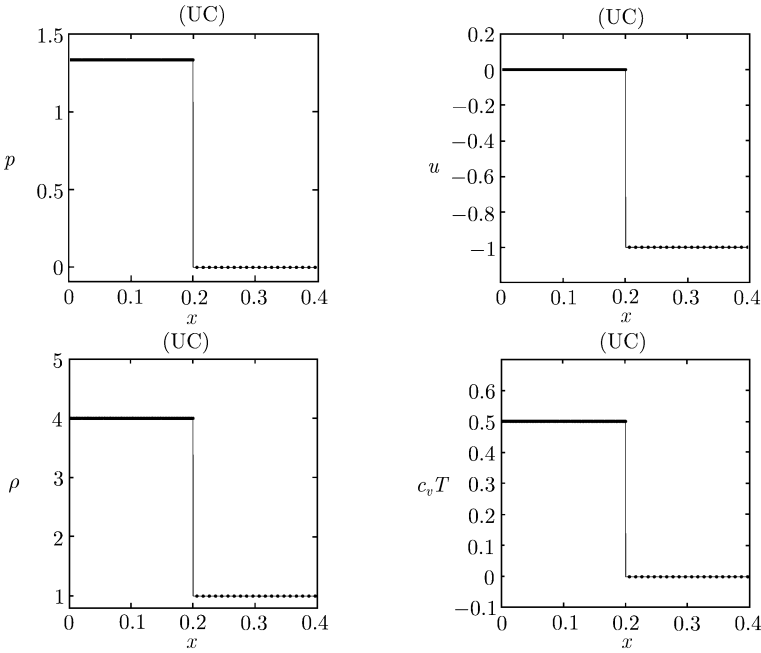
Figure 4.17   Noh's problem; UC computation (dots) employing shock-adaptive
Godunov-MUSCL scheme, compared with exact solution (solid line) at $t = 0.6$

An example of sudden expansion is the flow resulting from an abrupt with-drawal, at time $t = 0$, of a piston with a velocity $u = -2$ from the gas at rest having pressure $p = 1$ and density $\rho = 1$. If a Eulerian mesh is used, this is a moving boundary problem in that the piston position is not fixed with a mesh line, and some special treatment must be applied in order to correctly satisfy the bound-ary condition there. Such treatment would produce additional errors. In contrast, there is no difficulty in Lagrangian mesh, and we shall present only Lagrangian computations.

Figure 4.18 shows the computed results using Lagrangian method based on the conservation form equations (4.3). The method is seen capable of predicting the wall pressure and velocity (not shown here) correctly, but fails to predict the wall density, temperature T and entropy S correctly. We note that this failure is due to the inability of conventional shock-capturing methods to correctly simulate the flow near the start of the motion. This flow near the start of the motion is singular in nature. Therefore, spurious entropy is generated near the wall which remains unchanged with mesh refinement[12]. This is expected, because the flow near the start of the motion is a centered (rarefaction) waves which has no length scale. Using UC method, i.e., shock-adaptive Godunov method applied to (4.11), the wall overheating is completely eliminated while the whole flow field is correctly predicted, as seen in Figure 4.19.

In both examples, the UC is seen to completely overcome the difficulties of wall-overheating. We emphasize that the success of the UC method originates from the use of the entropy conservation equation to replace the energy one[12].

4.5.7   Strong rarefaction waves

Recently, a new defect of conventional shock-capturing methods was found in [29] due to the presence of strong rarefaction waves. The defect appears in Eulerian, Lagrangian, and gas-kinetic computation. As an example, consider the Riemann problem with initial data

$$(\rho, u, p) = \begin{cases} (1000.0, 0.0, 1000.0), & 0 \leqslant x < 0.3, \\ (1.0, 0.0, 1.0), & 0.3 \leqslant x \leqslant 1. \end{cases} \tag{4.35}$$
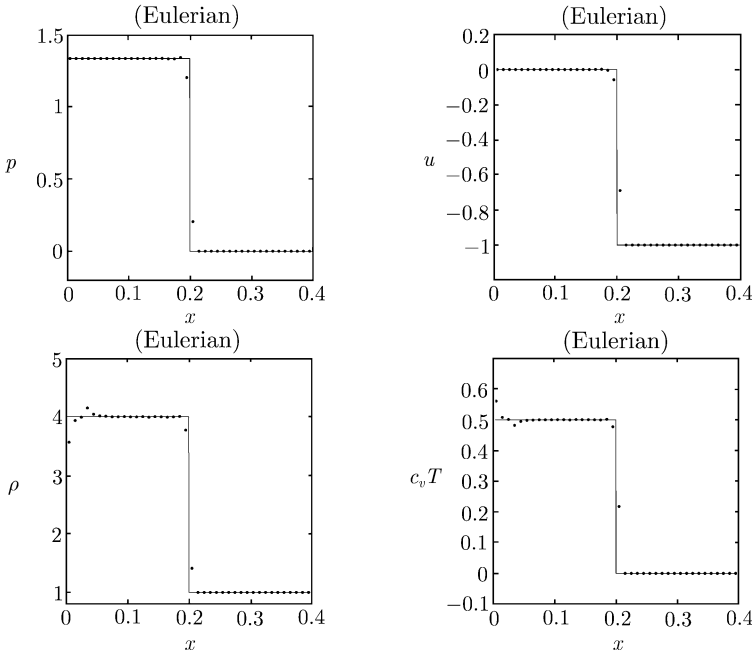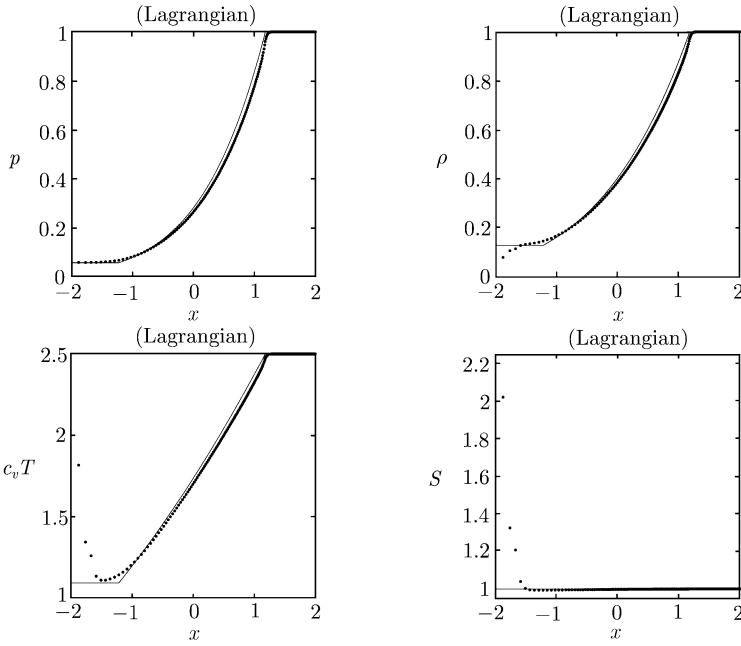
Figure 4.18   Sudden expansion of a gas; Lagrangian computation (dots) employing
Godunov-MUSCL scheme, compared with exact solution (solid line) at $t = 1.0$
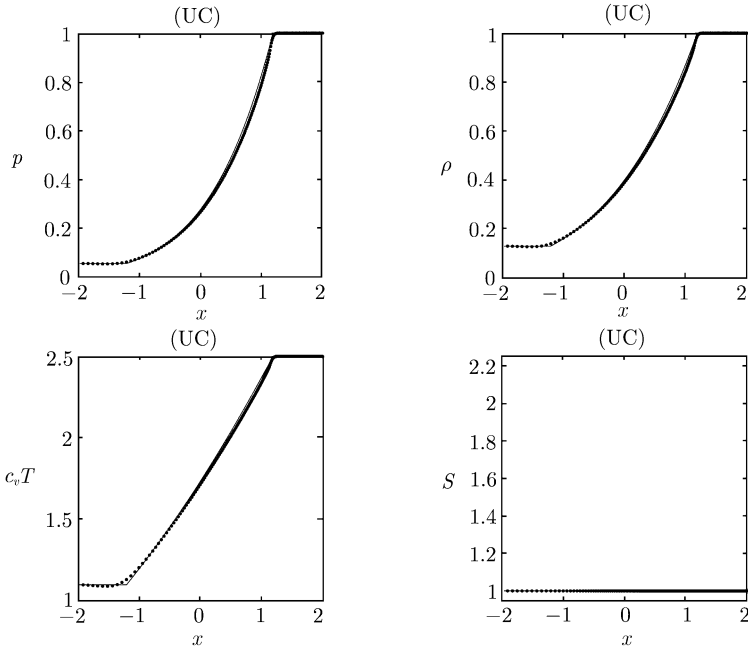
Figure 4.19   Sudden expansion of a gas; UC computation (dots) employing shock-
adaptive Godunov-MUSCL scheme, compared with exact solution (solid line) at $t = 1.0$

As shown in Figure 4.20, even with 500 cells in one unit, there is still significant error in the shock location. It is now realized[30] that this is due to the presence of a strong rarefaction wave which affects the location and strength of the adjacent contact line which, in turn, affects the location of the shock front. The UC method, again, can avoid this error as seen in Figure 4.21. We note that with the presence of a strong rarefaction wave, the density variation is great. In such a situation the classical Lagrangian method, which uses $\partial x/\partial \xi = 1/\rho$, will give large cell disparity and hence error. We use the UC based on the general equation (4.6) with the entropy equation (4.10) replacing the energy equation in (4.6).
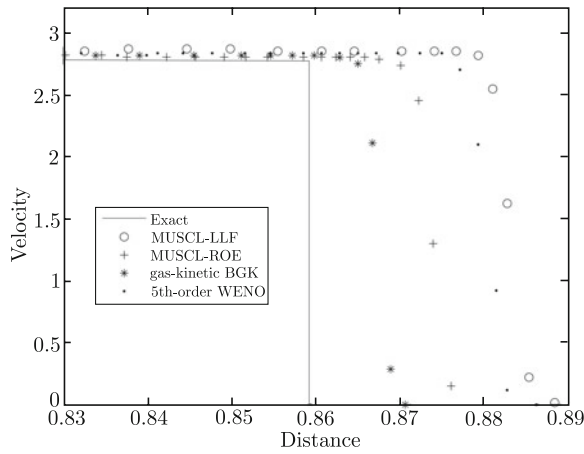
Figure 4.20   Velocities calculated by MUSCL-LLF, MUSCL-ROE, gas-kinetic BGK and 5th-order WENO methods with 500 cells (Tang and Liu[29])
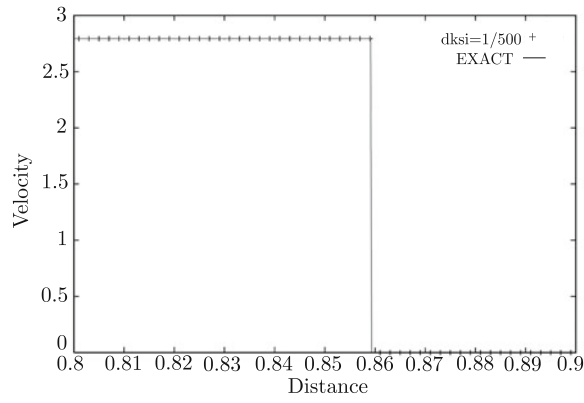
Figure 4.21   Velocities by the UC method with 500 cells

By putting $A = 1$ at $\lambda = 0$, we have a uniform mesh initially, and the difficulty with large cell disparity is avoided. In this way, the UC is, again, completely satisfactory, as shown in Figure 4.21.

4.6   Conclusions

From these examples, we conclude that for 1-D flow,

(1) Classical Lagrangian computation is superior to the Eulerian in its sharp resolution of contact discontinuities. In this regard, we note with pleasant interest that in the two seminal papers by Von Neumann[31] and by Godunov[32], the classical Lagrangian gas dynamics equations (4.3) were used, instead of their Eulerian counterpart (4.1).

(2) The UC computation, with its use of shock-adaptive Godunov scheme in place of the classical one, and with its use of the entropy conservation equation in place of the energy conservation equation for smooth flow, is superior to both Eulerian and Lagrangian computation.

(3) The UC computation is completely satisfactory for 1-D flow.

(4) The UC also applies to the system with real gas effect[33].

References

[1] W.H. Hui and S. Kudriakov. The role of coordinates in the computation of discontinuities in one-dimensional flow. *Computational Fluid Dynamics Journal*, 8: 495-510, 2000.

[2] D.H. Wagner. Equivalence of Euler and Lagrangian equations of gas dynamics for weak solutions. *J. Differ. Equations*, 68: 118-136, 1987.

[3] J.P. Boris and D.L. Book. Flux-corrected transport. I.SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, 11: 38-69, 1973.

[4] B. van Leer. Towards the ultimate conservative difference scheme IV, a new approach to numerical convection. *J. Comput. Phys.*, 23: 276-299, 1977.

[5] A. Harten, B. Engquist, S. Osher and S. R. Chakravarthy. Uniformly high order accuracy essentially non-oscillatory schemes III. *J. Comput. Phys.*, 71: 231-303, 1987.

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

[7] W.H. Hui and C.Y. Loh. A new Lagrangian method for steady supersonic flow computation, Part III: strong shocks. *J. Comput. Phys.*, 103: 465-471, 1992.

[8] C.Y. Lepage and W.H. Hui. A shock-adaptive Godunov scheme based on the generalized Lagrangian formulation. *J. Comput. Phys.*, 122: 291-299, 1995.

[9] A. Harten and J.M. Hyman. Self-adjusting grid method for one-dimensional hyperbolic conservation laws. *J. Comput. Phys.*, 50: 235-269, 1983.

[10] P.D. Lax and B. Wendroff. Syetems of conservation laws. *Comm. Pure and Applied Mathematics*, 13: 217-237, 1960.

[11] H.W. Hui and C.Y. Loh. A new Lagrangian method for steady supersonic flow computation, Part 2: slipline resolution. *J. Comput. Phys.*, 103: 450-464, 1992.

[12] W.H. Hui and S. Kudriakov. On wall overheating and other computational difficulties of shock-capturing methods. *Computational Fluid Dynamics J.*, 10: 192-209, 2001.

[13] K. Xu. Dissipative mechanism in Godunov method. *Computational Fluid Dynamics for the 21st Century*, M. Hafez, K. Morinishi, and J. Periaux (Eds), pp. 309-321, 2001.

[14] H.Z. Tang. On the sonic point glitch. *J. Comput. Phys.*, 202: 507-532, 2005.

[15] 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.

[16] R.E. Peierls. Theory on von Neumann's method of treating shocks. *Technical Report LA-332*, Los Alamos Scientific Laboratory, 1945.

[17] R. Landshoff. A numerical method for treating fluid flow in the presence of shocks. *Technical Report LA-1930*, Los Alamos Scientific Laboratory, 1955.

[18] I.G. Camerson. An analysis of the errors caused by using artificial viscosity terms to represent steady state shock waves. *J. Comput. Phys.*, 1: 1-20, 1966.

[19] L.G. Margolin, H.M. Ruppel and R.B. Demuth. Gradient Scaling for a Nonuniform Meshes. in *Numerical Methods for Laminar and Turbulent Flow*, Pineridge Press, Swansea, UK, 1985.

[20] W.F. Noh. Errors for calculations of strong shocks using an artificial viscosity and an artificial heat flux. *J. Comput. Phys.*, 72: 78-120, 1987.

[21] P. Glaister. An approximate linearized Riemann solver for the euler equations for real gases. *J. Comput. Phys.*, 74: 382-408, 1988.

[22] R. Sanders and A. Weiser. Higher resolution steggered mesh approach for nonlinear hyperbolic systems of conservation laws. *J. Comput. Phys.*, 101: 314-329, 1992.

[23] R. Menikoff. Errors when shock waves interact due to numerical shock width. *SIAM J. Sci. Comput.*, 15: 1227-1242, 1994.

[24] R. Donat and A. Marquina. Capturing shock reflections: An improved flux formula. *J. Comput. Phys.*, 125: 42-58, 1996.

[25] M. GEHMEYR, B. CHENG AND D. MIHALAS. Noh's constant-velocity shock problem revisited. *Shock Waves*, 7: 255-274, 1997.

[26] E.J. CARAMANA, M.J. SHASHKOV AND P.P. WHALEN. Formulations of artificial voscosity for multi-dimensional shock wave computations. *J. Comput. Phys.*, 144: 70-97, 1998.

[27] R.P. FEDKIW, A. MARQUINA AND R. MERRIMAN. An isobaric fix for the overheating problem in multimaterial compressible flows. *J. Comput. Phys.*, 148: 545-578, 1999.

[28] W.J. RIDER. Revisiting wall heating. *J. Comput. Phys.*, 162: 395-410, 2000.

[29] H.Z. TANG AND T.G. LIU. A note on the conservative schemes for the Euler equations. *J. Comput. Phys.*, 218: 451-459, 2006.

[30] S. KUDRIAKOV AND W.H. HUI. On a new defect of shock-capturing methods. *J. Comput. Phy.*, 227: 2105-2117, 2008.

[31] J. VON NEUMANN AND R.D. RICHTMYER. A method for the numerical calculation of hydrodynamic shocks. *J. Appl. Phys.*, 21: 232, 1950.

[32] S.K. GODUNOV. A Difference Scheme for Numerical Computation of Discontinuous Solutions of Hydrodynamic Equations. *Math. Sbornik*, 47: 271, 1959.

[33] C.Y. LOH AND M.S. LIOU. Lagrangian solution of supersonic real gas flows. *J. Comput. Phys.*, 104: 150-161, 1993.

Chapter 5

Comments on Current Methods for Multi-Dimensional Flow Computation

5.1 Eulerian Computation

The chief advantage of Eulerian computation is that the control volumes, i.e., the cells, are fixed in space. This facilitates writing computer codes and, if we discretize the integral equation (2.17), we automatically obtain a conservative numerical scheme which ensures correct capturing of shocks.

A list of defects of Eulerian computation for 1-D flow has been given in Section 3.7. These carry over to 2-D and 3-D flow, too; the most serious of these is the smearing of contact discontinuities. A good example is shown in Figure 5.1, where a Riemann problem for 2-D steady supersonic flow is studied. The large numerical error around the contact line (slipline) (Figure 5.2) is seen to also affect the accuracy of the smooth flow computation. As pointed out in Chapter 3, contact smearing is inherent in Eulerian computation because, as in 1-D flow, it does not align
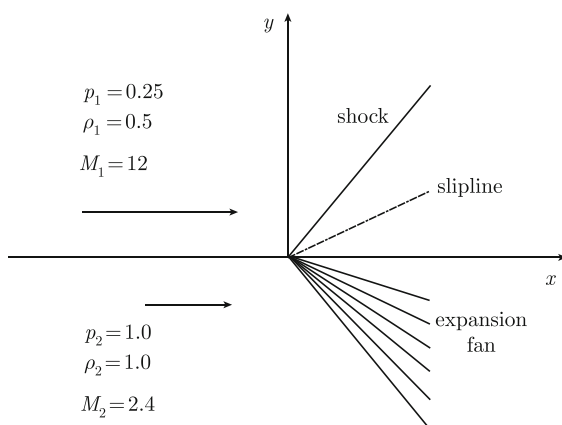
Figure 5.1   2-D steady Riemann problem: sketch

itself to Eulerian coordinates. Figure 5.3 shows the classical Lagrangian computation. Figure 5.4 shows results from the unified coordinate computation (for detail, see Chapter 9), which is completely satisfactory.
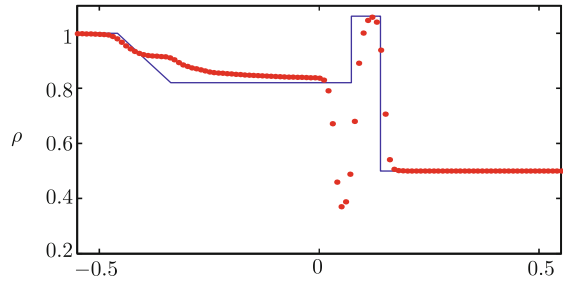
Figure 5.2   Density distribution in the 2-D steady Riemann problem; Eulerian computation (Godunov-MUSCL)
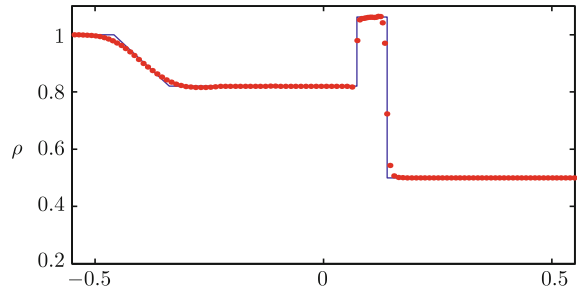
Figure 5.3   Density distribution in the 2-D steady Riemann problem; Lagrangian computation (Godunov-MUSCL)
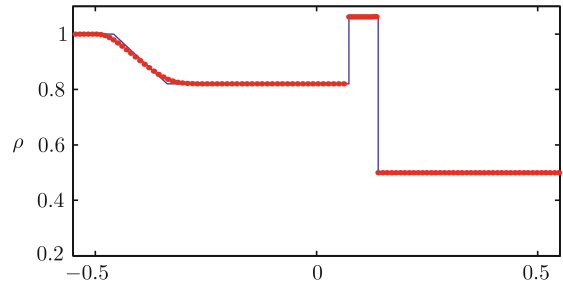
Figure 5.4   Density distribution in the 2-D steady Riemann problem; UC computation (Adaptive Godunov-MUSCL)

There are also defects that exist in multi-dimensional flows only. For example, shock instability and carbuncle phenomena can appear for the high speed

flow around a cylinder when the mesh is well aligned with the flow field[1]. The main reason for this is due to special dissipative mechanism in the Godunov method, where the cell averaging is the only source for the artificial dissipation[2].

Therefore, the anisotropy of the dissipative property triggers the shock instabilities. For example, a standard shock instability (carbuncle) phenomena from Roe's scheme is shown in Figure 5.5. Even though it is not totally conclusive at the current stage, there are many explanations existing in the literatures[3, 4]. For the UC method, due to the freedom in imposing the mesh velocity and the capability of shock fitting by moving the mesh, it may be possible to avoid or reduce the carbuncle phenomena.
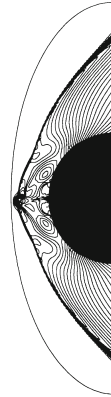
Figure 5.5　Shock instability[1]

　　Another serious drawback is that for computing flow past a body, which is the central problem in CFD, a body-fitted mesh must be generated prior to computing the flow field. But, mesh generation remains a tedious and time-consuming process, even after more than three decades of research.

5.2　Lagrangian Computation

For 1-D flow, classical Lagrangian computation was shown in Chapter 4 to be superior to the Eulerian. Its chief advantage is sharp resolution of contact discontinuities, and this arises from the flow property that a contact line is a material line and, therefore, aligns with a Lagrangian coordinate. This advantage carries over to multi-dimensional flow also as seen in Figure 5.3.

　　However, for multi-dimensional flow, Lagrangian coordinate system has two serious difficulties.

　　(1) Due to cell movement and deformation—both are unknown a priori—the integral equation (2.16) could not be easily written in conservation PDE form. Indeed, as late as 1999 D. Serre[5] stated that "Writing the equations of gas dynamics in Lagrangian coordinates is very complicated if (dimension) $D \geqslant 2$". In this re-

gard, 1-D flow is a fortuitous exception, see (4.3). This has serious consequences. To begin with, lack of conservation PDE form has prevented Lagrangian scheme to be a scheme on a moving mesh in Eulerian space; the design of such a scheme has great historical importance as well as practical application, as pointed out in[6, 7]. Although (2.16) looks very similar to, and indeed simpler than, (2.8) in Eulerian coordinates, the difficulty in using (2.16) as compared to (2.8) is that the control volumes (cells) move and deform with the fluid velocity, which is unknown and has to be determined as part of the solution. Therefore, without conservation form PDE, additional computation procedure is needed to tract the movement and deformation of the cells while using (2.16). This is usually done by employing staggered meshes for the velocity which controls the mesh movement and deformation. But switching between meshes requires interpolations of flow variables and geometries, producing numerical diffusion.

(2) Computation may break down due to cell deformation. This is because a Lagrangian computational cell is literally a fluid particle with finite size, no matter how small, and hence deforms with the fluid. This is illustrated in Figure 5.6 for a cylindrical explosion, where the computation breaks down after some finite time $t = 0.4$, due to severe cell deformation. The break down is typically associated with the appearance of negative pressure.

(a) a cylindrical explosion at $t=0$ (b) mesh at $t=0.4$ using Lagrangian method
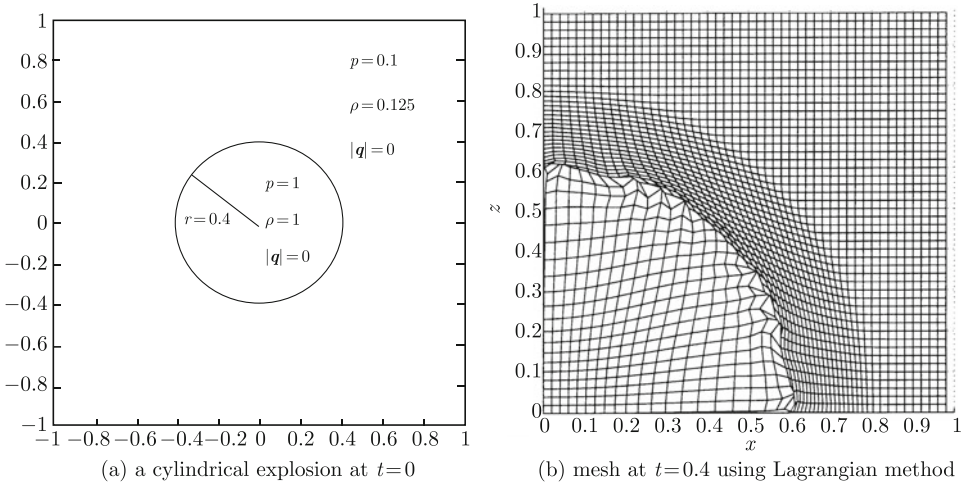
Figure 5.6   A typical Lagrangian mesh; Computation breaks down soon afterwards

Great efforts have been spent in Lagrangian computation to prevent it from breaking down by using special treatment; the Arbitrary-Lagrangian-Eulerian (ALE) method represents such a successful treatment.

5.3   The ALE Computation

Early efforts to combine the advantages of both Lagrangian and Eulerian systems have resulted in the famous Particle-in-Cell method[8, 9] and the Marker-and-Cell method[10−13].The highly original idea of Harlow developed in the Particle-in-Cell method of separating a computational cycle into a Lagrangian phase plus a convective, or remap/rezone, phase has been widely used in many hydrodynamic computer codes, in particular, in the celebrated ALE code[14−19].

In this method a computational cycle consists of a computation phase in Lagrangian space followed by a remap/rezone phase to the Eulerian or ALE space. In the Lagrangian phase, it has the advantages as well as the drawbacks of the Lagrangian computation mentioned above, namely, it resolves contact discontinuities sharply but, due to the lack of conservation PDE form, it uses staggered meshes and hence produces numerical diffusion arising from switching between the meshes. Moreover, numerical diffusion is also introduced in the remap/rezone phase of the computation, because it too requires interpolations of the flow variables and of geometries. Indeed, it was demonstrated by Hall[20] that rezoning results in the type of errors encountered in Eulerian solutions and that with continuous remapping/rezoning, ALE computation is equivalent to Eulerian computation. The remap/rezone phase is, however, needed in order to prevent computational breakdown.

While ALE has achieved greatly in resolving contact discontinuities, an important phase of its computation—the remap/rezone phase—requires the intervention of the user[18], although new ideas have recently been proposed[19] to avoid it.

5.4   Moving Mesh Methods

In methods of this type, meshes are re-distributed statically, or moved dynamically, according to flow properties so as to increase computational accuracy and efficiency. These typically require solution to an elliptic equation at every step

of the re-distribution or movement. Many moving mesh methods have been developed[21−33].

To account for the effects of re-distribution and movement of the mesh, it is necessary to have one or more space (or geometric) conservation equations, which are either derived mathematically or given intuitively, in addition to the physical conservation laws. While these may be necessary condition(s), it is not certain whether the effects due to mesh movement on the flow are fully taken into account.

In a recent paper[34], another defect of moving mesh methods has been found that in a supersonic flow past a blunt nosed airfoil, although mesh re-distribution may increase the shock resolution, the accuracies for some integral physical properties, e.g., lift and drag, can decrease.

Most above moving mesh methods are static ones, where the mesh is re-generated according to an elliptic solver within every or several time steps. Another type of moving mesh methods, which have the similarity with the Lagrangian method, are called dynamic ones. There is a defined governing equation to determine the mesh moving velocity and the original governing equations are transformed into the computational space, such as coordinate transformation or mapping. Many applications for the dynamic moving mesh methods are those related to the scalar equations with the development of possible singularities[35−39], but are not immediately applicable to the Euler equations of gas dynamics. In engineering, moving mesh methods have also been successfully developed for the problems with moving boundaries and interfaces[40, 41].

5.5   Optimal Coordinates

We see from the discussions above, each of the different coordinate systems have advantages as well as drawbacks, and they tend to be complementary. We should thus try to find, or construct, an "optimal" system of coordinates.

For compressible flow, we want such a system to possess the following desirable properties:

(1) Conservation form PDE exists, as in Eulerian system;

(2) Contact discontinuities are sharply resolved, as in Lagrangian system;

(3) Body-fitted mesh can be generated automatically, and, if possible, we may

also wish our system to be orthogonal and uniform. It is the aim of this book to show that the unified coordinate system possesses these desirable properties.

References

[1] J. QUIRK. A contribution to the great Riemann solver debate. *Int. J. Num. Meth. Fluids*, 18: 555-574, 1994.

[2] K. XU AND Z.W. LI. Dissipative mechanism in Godunov-type schemes. *Int. J. Numer. Methods in Fluids*, 37: 1-22, 2001.

[3] M.S. LIOU. Probing numerical fluxes, positivity, and entropy-satisfying property. *AIAA 97-2035*, 1997.

[4] J.C. ROBINET, J. GRESSIER, G. CASALIS AND J.M. MOSCHETTA. Shock wave instability and the carbuncle phenomenon: Same intrinsic origin? *J. Fluid Mech.*, 417: 237-263, 2000.

[5] D. SERRE. Systems of Conservation Laws. Cambridge: *Cambridge University Press,* 1999.

[6] B. DESPRES AND C. MAZERAN. Lagrangian gas dynamics in two dimensions and Lagrangian systems. *Arch. Rathion. Mech. Anal.*, 178: 327-372, 2005.

[7] B. DESPRES AND C. MAZERAN. Symmetrization of Lagrangian gas dynamics in dimensions two and multidimensional solvers. *Comptes Rendus (Mecanique)*, 331: 475, 2003.

[8] F.H. HARLOW. LAMS-1956, *Los Alamos Scientic Laboratory Report*, 1955.

[9] A.A. AMSDEN. LAMS-3466, *Los Alamos Scientic Laboratory Report*, 1966.

[10] F.H. HARLOW AND J.E. WELCH. Numericalcal culation of time-dependent viscous incompressible fluid with free surface. *Phys. Fluids*, 8: 2182, 1965.

[11] J.E. WELCH, F.H. HARLOW, J.P. SHANNON AND B.J.DALY. LA-3425, *Los Alamos Scientic Laboratory Report*, 1966.

[12] A.J. CHORIN. Numerical solution of Navier-Stokes equations. *Math. Comput.*, 22: 745-762, 1968.

[13] F.H. HARLOW AND A.A. AMSDEN. An umerical fluid dynamics calculation for all flow speeds. *J. Comput. Phys.*, 8: 197-213, 1971.

[14] J.G. TRULIO. Air force weapons laboratory.*Kirtland Air Force Base Report No. AFWL-TR-66-19,* 1966.

[15] C. W. HIRT, A. A. AMSDEN AND J. L. COOK. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *J. Comput. Phys.*, 14: 227-253, 1974.

[16] W. PRACHT. Calculating three-dimensional fluid flows at all speeds with an Eulerian Lagrangian computing mesh. *J. Comput. Phys.*,17: 132-159, 1975.

[17] L.G. MARGOLIN. Introduction to an arbitrary Lagrangian-Eulerian computing method for all flow speeds. *J. Comput. Phys.*, 135: 198-202, 1997.

[18] D. J. BENSON. Computational methods in Lagrangian and Eulerian hydrocodes. *Comput. Method. Appl. Mech. Engrg.*, 99: 235-394, 1992.

[19] P. KNUPP, L.G. MARGOLIN AND M.SHASHKOV. Reference Jacobian optimization-based rezone strategies for arbitrary Lagrangian Eulerian methods. *J. Comput. Phys.*, 176: 93-128, 2002.

[20] M. S. HALL. A comparison of the first and second order rezoned and Lagrangian Godunov solutions. *J. Comput. Phys.*, 90: 458-485, 1990.

[21] A. WINSLOW. Numerical solution of the quasi-linear Poisson equation. *J. Comput. Phys.* , 1: 149-172, 1967.

[22] K. MILLER AND R.N. MILLER. Moving finite element. *SIAM J. Numer Anal.*, 18: 1019-1032, 1981.

[23] S.F. DAVIS AND J.E. FLAHERTY. An adaptive finite element method for initial-boundary value problems for partial differential equations. *SIAM J. Sci. Stat. Comput.*, 3: 6-27, 1982.

[24] J.U. BRACKBILL AND J.S. SALTZMAN. Adaptive zoning for singular problems in two dimensions. *J. Comput. Phys.* , 46: 342-368, 1982.

[25] A.S. DVINSKY. Adaptive grid generation from harmonic maps on Riemannian manifolds. *J. Comput. Phys.*, 95: 221-244, 1991.

[26] J.U. BRACKBILL. An adaptive grid with directional control. *J. Comput. Phys.*, 108: 38-50, 1993.

[27] S. LI AND L. PETZOLD. Moving mesh methods with upwinding schemes for time-dependent PDEs. *J. Comput. Phys.*, 131: 368-377, 1997.

[28] W.M. CAO, W.Z. HUANG AND R.D. RUSSELL. An r-adaptive finite element method based upon moving mesh PDEs. *J. Comput. Phys.*, 149: 221-244, 1999.

[29] W.Q. REN AND X.P. WANG. An iterative grid redistribution method for singular problems in multiple dimensions. *J. Comput. Phys.*, 159: 246-273, 2000.

[30] J.M. STOCKIE, J.A. MACKENZIE AND R.D. RUSSELL. A moving mesh method for one-dimensional hyperbolic conservation laws. *SIAM J. Sci. Comput.*, 22: 1791-1813, 2001.

[31] H.D. CENICEROS AND T.Y. HOU. An efficient dynamically adaptive mesh for potentially singular solutions. *J. Comput. Phys.*, 172: 609-639, 2001.

[32] R. LI, T. TANG AND P.W. ZHANG. Moving mesh methods in multiple dimensions based on harmonic maps. *J. Comput. Phys.*, 170: 562-588, 2001.

[33] H.Z. TANG AND T. TANG. Adaptive mesh methods for one-and two-dimensional hyperbolic conservation laws. *SIAM J. Numer. Anal.*, 41: 487-515, 2003.

[34] N.K. YAMALEEV AND M.H. CARPENTER. On accuracy of adaptive grid methods for captured shocks. *J. Comput. Phys.*, 181: 280-316, 2002.

[35] R.J. GELINAS, S.K. DOSS AND K. MILLER. The moving finite element method: application to general partial differential equations with multiple large gradients. *J. Comput. Phys.*, 40: 202-249, 1981.

[36] E.A. DORFI AND L. O'C. DRURY. Simple adaptive grids for 1-D initial value problems. *J. Comput. Phys.*, 69: 175-195, 1987.

[37] W. HUANG, Y. REN, AND R.D. RUSSELL. Moving mesh methods based upon moving mesh partial differential equations. *J. Comput. Phys.*, 11: 279-290, 1994.

[38] W. HUANG AND R.D. RUSSELL. Adaptive mesh movement — the MMPDE approach and its applications. *J. Comput. Appl. Math.*, 128: 383-398, 2001.

[39] W. HUANG, J. MA AND R.D. RUSSELL. A study of moving mesh PDE methods for numerical simulation of blowup in reaction diffusion equations. *J. Comput. Phys.*, (2008), doi: 10.1016/j.jcp.2008.03.024.

[40] W. SHYY. Computational Modeling for Fluid Flow and Interfacial Transport. Amsterdam: Elsevier, 1994.

[41] W. SHYY, R.W. SMITH, H.S. UDAYKUMAR AND M.M. RAO. Computational Fluid Dynamics with Moving Boundaries. Washingtom. D. C.: Taylor & Francis, Inc., 1996.

Chapter 6

The Unified Coordinates Formulation of CFD

6.1   Hui Transformation

We introduce arbitrary coordinates $(\lambda, \xi, \eta, \zeta)$ via a transformation from Cartesian (t, x, y, z) as follows:

$$
\begin{cases}
\mathrm{d}t = \mathrm{d}\lambda, \\
\mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta + P\mathrm{d}\zeta, \\
\mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta + Q\mathrm{d}\zeta, \\
\mathrm{d}z = W\mathrm{d}\lambda + C\mathrm{d}\xi + N\mathrm{d}\eta + R\mathrm{d}\zeta.
\end{cases}
\tag{6.1}
$$

From (6.1), we get

$$
\frac{D_{\boldsymbol{Q}}}{Dt}\begin{pmatrix}\xi\\\eta\\\zeta\end{pmatrix} = \boldsymbol{0},
\tag{6.2}
$$

where $D_{\boldsymbol{Q}}/Dt = \partial/\partial t + \boldsymbol{Q}\cdot\nabla_{\boldsymbol{x}}$. So the coordinates (ξ, η, ζ), and hence the computational cells, move with the pseudo-particle whose velocity is $\boldsymbol{Q} = (U, V, W)$.

The above coordinate system includes two important special cases: Eulerian when $\boldsymbol{Q} = \boldsymbol{0}$ and Lagrangian when $\boldsymbol{Q} = \boldsymbol{q}$, $\boldsymbol{q}$ being the fluid velocity. In the general case, we have a coordinate system with three degrees of freedom: U, V and W are arbitrary.

Special cases: For 1-D flow, (6.1) reduces to

$$
\begin{cases}
\mathrm{d}t = \mathrm{d}\lambda, \\
\mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi,
\end{cases}
\tag{6.3}
$$

which is identical to (4.4). For 2-D steady flow, (6.1) reduces to

$$
\begin{cases}
\mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi, \\
\mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi.
\end{cases}
\tag{6.4}
$$

For 2-D unsteady flow, (6.1) reduces to

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta. \end{cases} \tag{6.5}$$

For 3-D steady flow, (6.1) reduces to

$$\begin{cases} \mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta, \\ \mathrm{d}z = W\mathrm{d}\lambda + C\mathrm{d}\xi + N\mathrm{d}\eta. \end{cases} \tag{6.6}$$

6.2   Geometric Conservation Laws

Whereas the mesh velocity (U, V, W) is arbitrary, the nine coefficients $A, B, C, \cdots, R$ in transformation (6.1) are not arbitrary, but must satisfy a set of compatibility conditions for $\mathrm{d}x, \mathrm{d}y$ and $\mathrm{d}z$ to be total differentials. These conditions are

$$\begin{cases} \dfrac{\partial A}{\partial \lambda} = \dfrac{\partial U}{\partial \xi}, & \dfrac{\partial L}{\partial \lambda} = \dfrac{\partial U}{\partial \eta}, & \dfrac{\partial P}{\partial \lambda} = \dfrac{\partial U}{\partial \zeta}, \\[2mm] \dfrac{\partial B}{\partial \lambda} = \dfrac{\partial V}{\partial \xi}, & \dfrac{\partial M}{\partial \lambda} = \dfrac{\partial V}{\partial \eta}, & \dfrac{\partial Q}{\partial \lambda} = \dfrac{\partial V}{\partial \zeta}, \\[2mm] \dfrac{\partial C}{\partial \lambda} = \dfrac{\partial W}{\partial \xi}, & \dfrac{\partial N}{\partial \lambda} = \dfrac{\partial W}{\partial \eta}, & \dfrac{\partial R}{\partial \lambda} = \dfrac{\partial W}{\partial \zeta}. \end{cases} \tag{6.7}$$

$$\begin{cases} \dfrac{\partial A}{\partial \eta} = \dfrac{\partial L}{\partial \xi}, & \dfrac{\partial A}{\partial \zeta} = \dfrac{\partial P}{\partial \xi}, & \dfrac{\partial L}{\partial \zeta} = \dfrac{\partial P}{\partial \eta}, \\[2mm] \dfrac{\partial B}{\partial \eta} = \dfrac{\partial M}{\partial \xi}, & \dfrac{\partial B}{\partial \zeta} = \dfrac{\partial Q}{\partial \xi}, & \dfrac{\partial M}{\partial \zeta} = \dfrac{\partial Q}{\partial \eta}, \\[2mm] \dfrac{\partial C}{\partial \eta} = \dfrac{\partial N}{\partial \xi}, & \dfrac{\partial C}{\partial \zeta} = \dfrac{\partial R}{\partial \xi}, & \dfrac{\partial N}{\partial \zeta} = \dfrac{\partial R}{\partial \eta}. \end{cases} \tag{6.8}$$

We note that of the eighteen conditions in (6.7) and (6.8), only nine of them are independent. We shall take the first nine conditions in (6.7), which all involve λ-derivative and are called time-evolution, to be the independent conditions; the remaining nine in (6.8), are called free divergence constraints and hold for all time provided they hold initially. Equations (6.7) are also called geometric conservation laws.

6.3   Derivation of Governing Equations in Conservation Form

Consider conservation of mass equation, which can be written as

$$0 = \frac{\partial \rho}{\partial t} + \frac{\partial(\rho u_j)}{\partial x_j} \quad (j = 1, 2, 3)$$

$$= \frac{\partial(\rho u_\alpha)}{\partial x_\alpha} \quad (\alpha = 0, 1, 2, 3, x_0 = t, u_0 = 1) \tag{6.9}$$

$$= \oint_{\partial\Omega} \rho u_\alpha \widehat{\mathrm{d}x}_\alpha,$$

after using Gauss divergence theorem. Here $x_0 = t, u_0 = 1$, Ω is any control volume and the summation convention has been used.

We define

$$\begin{cases} \widehat{\mathrm{d}x}_0 = \mathrm{d}x_1 \mathrm{d}x_2 \mathrm{d}x_3, \\ \widehat{\mathrm{d}x}_1 = -\mathrm{d}x_2 \mathrm{d}x_3 \mathrm{d}x_0, \\ \widehat{\mathrm{d}x}_2 = \mathrm{d}x_3 \mathrm{d}x_0 \mathrm{d}x_1, \\ \widehat{\mathrm{d}x}_3 = -\mathrm{d}x_0 \mathrm{d}x_1 \mathrm{d}x_2 \end{cases} \tag{6.10}$$

and similarly for $\widehat{\mathrm{d}\xi}_\beta$. We further define

$$\begin{cases} \widehat{x}_0 = (x_1, x_2, x_3), \\ \widehat{x}_1 = (x_2, x_3, x_0), \\ \widehat{x}_2 = (x_3, x_0, x_1), \\ \widehat{x}_3 = (x_0, x_1, x_2) \end{cases} \tag{6.11}$$

and similarly for $\widehat{\xi}_\beta$. Then from the transformation we get

$$\widehat{\mathrm{d}x}_\alpha = U_{\alpha\beta} \widehat{\mathrm{d}\xi}_\beta, \tag{6.12}$$

where $\xi_0 = \lambda, \xi_1 = \xi, \xi_2 = \eta, \xi_3 = \zeta$ and

$$U_{\alpha\beta} = \frac{\partial \widehat{x}_\alpha}{\partial \widehat{\xi}_\beta}. \tag{6.13}$$

Hence,

$$0 = \oint_{\partial\Omega} \rho u_\alpha \widehat{\mathrm{d}x}_\alpha = \oint_{\partial\Omega} \rho u_\alpha U_{\alpha\beta} \widehat{\mathrm{d}\xi}_\beta. \tag{6.14}$$

Using Gauss divergence theorem, we get

$$\frac{\partial K_\beta}{\partial \xi_\beta} = 0, \tag{6.15}$$

where

$$K_\beta = \rho u_\alpha U_{\alpha\beta}. \tag{6.16}$$

(6.15) is the mass equation written in conservation form in the arbitrary co-ordinate system. We can similarly derive the momentum and energy equation in conservation PDE form.

To summarize, we have

$$\begin{cases} \dfrac{\partial(\rho u_\alpha)}{\partial x_\alpha} = 0, \quad \alpha = 0, 1, 2, 3, \\[3mm] \dfrac{\partial(\rho u_j u_\alpha)}{\partial x_\alpha} + \dfrac{\partial p}{\partial x_j} = 0, \quad j = 1, 2, 3, \\[3mm] \dfrac{\partial(\rho u_\alpha H)}{\partial x_\alpha} - \dfrac{\partial p}{\partial x_0} = 0 \end{cases} \tag{6.17}$$

in the Cartesian coordinates, and

$$\begin{cases} \dfrac{\partial K_\beta}{\partial \xi_\beta} = 0, \quad \beta = 0, 1, 2, 3, \\[3mm] \dfrac{\partial(K_\beta u_j + p U_{j\beta})}{\partial \xi_\beta} = 0, \quad j = 1, 2, 3, \\[3mm] \dfrac{\partial(K_\beta H - p U_{0\beta})}{\partial \xi_\beta} = 0 \end{cases} \tag{6.18}$$

in arbitrary coordinates. Here $H = e + p/\rho$.

The physical laws are now written in conservation PDE form (6.18) in arbitrary coordinates, including the Lagrangian. However, (6.18) is not a closed system, because it contains (through $U_{\alpha\beta}$) new unknowns: the coefficients $A, B, C, \cdots, R$ in the transformation (6.1)[①]. Although unknown, these coefficients are nevertheless related to the mesh velocity (U, V, W) via the compatibility conditions (6.7). To have a system of PDE that is closed and in conservation form it is, therefore, necessary and sufficient to append the time evolution equations (6.7) to the physical conservation laws (6.18). (6.7) are called geometric conservation laws (GCL). In this regard, Eulerian coordinates represent a degenerate case in that the geometric conservation laws reduce to triviality, and the physical conservation laws alone suffice as a closed system.

We conclude that (6.18) and (6.7) form a closed system of fourteen conservation PDE, containing fourteen unknowns: $\rho, p, \boldsymbol{q}, A, B, C, \cdots, R$ (the mesh velocities (U, V, W) in these equations will be given in Chapter 7). These are

---

[①] this explains why, as pointed out in Section 5.2, with Lagrangian coordinates, the physical laws alone cannot be written in closed conservation PDE form; the 1-D case (4.3) is just a fortuitous exception.

$$
\begin{cases}
\dfrac{\partial \rho \Delta}{\partial \lambda} + \dfrac{\partial (\rho X)}{\partial \xi} + \dfrac{\partial (\rho Y)}{\partial \eta} + \dfrac{\partial (\rho Z)}{\partial \zeta} = 0, \\[2mm]
\dfrac{\partial \rho \Delta q_j}{\partial \lambda} + \dfrac{\partial (\rho X q_j + p\boldsymbol{J}_1)}{\partial \xi} + \dfrac{\partial (\rho Y q_j + p\boldsymbol{J}_2)}{\partial \eta} + \dfrac{\partial (\rho Z q_j + p\boldsymbol{J}_3)}{\partial \zeta} = 0, \\[1mm]
\hspace{9cm} j = 1,2,3, \\[2mm]
\dfrac{\partial \rho \Delta e}{\partial \lambda} + \dfrac{\partial (\rho X e + p\boldsymbol{J}_1 \cdot \boldsymbol{q})}{\partial \xi} + \dfrac{\partial (\rho Y e + p\boldsymbol{J}_2 \cdot \boldsymbol{q})}{\partial \eta} + \dfrac{\partial (\rho Z e + p\boldsymbol{J}_3 \cdot \boldsymbol{q})}{\partial \zeta} = 0, \\[1mm]
\hspace{9cm} j = 1,2,3, \\[2mm]
\dfrac{\partial A}{\partial \lambda} = \dfrac{\partial U}{\partial \xi}, \\[2mm]
\dfrac{\partial L}{\partial \lambda} = \dfrac{\partial U}{\partial \eta}, \\[2mm]
\dfrac{\partial P}{\partial \lambda} = \dfrac{\partial U}{\partial \zeta}, \\[2mm]
\dfrac{\partial B}{\partial \lambda} = \dfrac{\partial V}{\partial \xi}, \\[2mm]
\dfrac{\partial M}{\partial \lambda} = \dfrac{\partial V}{\partial \eta}, \\[2mm]
\dfrac{\partial Q}{\partial \lambda} = \dfrac{\partial V}{\partial \zeta}, \\[2mm]
\dfrac{\partial C}{\partial \lambda} = \dfrac{\partial W}{\partial \xi}, \\[2mm]
\dfrac{\partial N}{\partial \lambda} = \dfrac{\partial W}{\partial \eta}, \\[2mm]
\dfrac{\partial R}{\partial \lambda} = \dfrac{\partial W}{\partial \zeta}.
\end{cases}
\tag{6.19}
$$

Here $X = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_1, Y = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_2, Z = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_3$ and

$$
\begin{cases}
\boldsymbol{J}_1 = \begin{pmatrix} MR - NQ \\ NP - LR \\ LQ - MP \end{pmatrix}, \quad \boldsymbol{J}_2 = \begin{pmatrix} CQ - BR \\ AR - CP \\ BP - AQ \end{pmatrix}, \\[6mm]
\boldsymbol{J}_3 = \begin{pmatrix} BN - CM \\ CL - AN \\ AM - BL \end{pmatrix}, \quad \Delta = \begin{vmatrix} A & L & P \\ B & M & Q \\ C & N & R \end{vmatrix}.
\end{cases}
\tag{6.20}
$$

We remark that the 14-equation system in arbitrary coordinates (6.19) appears to be much larger than the 5-equation system in Eulerian coordinates (6.17), but the eigenvalues corresponding to the additional nine equations—the geometric conservation laws—are found to be equal to zero (multiplicity 9), giving rise only to new linearly degenerated waves and no new nonlinear waves (shocks, rarefaction

waves, etc.). Details of these eigenvalues and their corresponding eigenvectors are given in [1] for 2-D flow and in [2] for 3-D flow. Therefore, system (6.19) does not generate spurious nonlinear waves that are not in the original system (6.17). Computationally (see Chapter 10 for details), the physical conservation laws are first solved, keeping the geometric variables $(A, B, C, \cdots, R)$ fixed, then the mesh velocity (U, V, W) are computed, and finally the geometric variables are updated using the geometric conservation laws alone. In this way, even the additional linearly degenerated waves do not appear (see [1, 2]). It is also interesting to observe that in the important special case of Lagrangian coordinates when $\boldsymbol{Q} = \boldsymbol{q}$, the linearly degenerated wave corresponding to the zero eigenvalue is not new but is already in the original system.

As to computing time requirement, the bulk of it is spent on solving the physical conservation laws, which is the same as in Eulerian computation. Additional time used in updating the geometric conservation laws adds about 3% to the total, while additional time used to compute the mesh velocity (U, V, W) adds another $5\% \sim 10\%$, depending on the problems at hand.

In the special case of 2-D flow, the system of governing equations (6.19) simplifies to

$$\frac{\partial \boldsymbol{E}}{\partial \lambda} + \frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta} = 0, \tag{6.21}$$

where

$$\boldsymbol{E} = \begin{pmatrix} \rho\Delta \\ \rho\Delta u \\ \rho\Delta v \\ \rho\Delta e \\ A \\ B \\ L \\ M \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} \rho X \\ \rho X u + pM \\ \rho X v - pL \\ \rho X e + p(uM - vL) \\ -U \\ -V \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{G} = \begin{pmatrix} \rho Y \\ \rho Y u - pB \\ \rho Y v + pA \\ \rho Y e + p(vA - uB) \\ 0 \\ 0 \\ -U \\ -V \end{pmatrix} \tag{6.22}$$

with $\Delta = AM - BL$, $X = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_1 = (u - U)M - (v - V)L$ and $Y = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_2 = (v - V)A - (u - U)B$, where $\boldsymbol{J}_1 = (M, -L)^{\mathrm{T}}$ and $\boldsymbol{J}_2 = (-B, A)^{\mathrm{T}}$. Here X and Y are the components of the relative velocity in the $\nabla\xi$ and $\nabla\eta$ directions with the length of the boundaries. The first four equations in (6.22) are the physical conservation laws and the last four the geometric conservation laws.

References

[1] W.H. HUI, P.Y. LI AND Z.W. LI. A unified coordinate system for solving the two-dimensional Euler equations. *J. Comput. Phys.*, 153: 596-637, 1999.

[2] W.H. HUI AND S. KUDRIAKOV. A unified coordinate system for solving the three-dimensional Euler equations. *J. Comput. Phys.*, 172: 235-260, 2001.

Chapter 7

Properties of the Unified Coordinates

We shall call the system of coordinates $(\lambda, \xi, \eta, \zeta)$ defined in (6.1) unified in the sense that it unifies the Eulerian system when $\boldsymbol{Q} = \boldsymbol{0}$ with the Lagrangian when $\boldsymbol{Q} = \boldsymbol{q}$, and also in the sense that the system of governing equations (6.19) unites the geometrical conservation laws with the physical ones to form a closed system of PDE in conservation form.

7.1   Relation to Eulerian Computation

The Eulerian computation is a special case of the unified coordinate (UC) approach by assigning $\boldsymbol{Q} = \boldsymbol{0}$, which makes the system simple. Since the UC system is in a conservative form, it may be regarded as a moving mesh method in the Eulerian space. As such, UC has advantages over the Eulerian system in the following aspects: ① The mesh velocity can be chosen to make the mesh align with the contact line or contact surface, see Section 7.4. ② The UC formulation naturally provides a foundation for mesh generation, where the mesh velocity can be used in this process. ③ By properly choosing mesh velocity $\boldsymbol{Q}$, UC can handle the interaction problems between flow and solid at ease (see example 4 in Section 12.5).

7.2   Relation to Classical Lagrangian Coordinates

We shall now prescribe conditions to be satisfied by the mesh velocity (U, V, W) so as to give the unified coordinates additional useful properties.

For 1-D flow, we require the coordinate ξ in (6.3) to be a material coordinate, i.e.,

$$\frac{D_q \xi}{Dt} = 0, \tag{7.1}$$

hence $U = u$ but A is arbitrary. This includes the classical Lagrangian coordinate

as a special case when $U = u$ and $A = 1/\rho$, or, we may say that the unified coordinate is a generalization of the Lagrangian one in that the coefficient A in (6.3) is arbitrary rather than equal to $1/\rho$.

We also use shock-adaptive Godunov scheme to replace the classical Godunov scheme and the entropy conservation equation to replace the energy conservation equation for computing smooth flow, as explained in Chapter 4.

For 2-D flow, we require a coordinate in (6.5), ξ say, to be a material coordinate, i.e.,

$$\frac{D_q\xi}{Dt} = 0, \tag{7.2}$$

plus a mesh angle-preserving (or Jacobian-preserving) condition to jointly determine the mesh velocity (U, V).

For 3-D flow, we require two coordinates in (6.1), ξ and η, to be material coordinates, i.e.,

$$\frac{D_q\xi}{Dt} = 0, \quad \frac{D_q\eta}{Dt} = 0, \tag{7.3}$$

plus a mesh skewness-preserving (or Jacobian-preserving) condition to jointly determine the mesh velocity (U, V, W).

We see from above that the unified coordinates can be regarded as a generalization of the classical Lagrangian coodinates, which requires all coordinates (ξ, η, ζ) to be material coordinates. By requiring only one of the two coordinates in 2-D flow, and only two of the three coordinates in 3-D, to be material the UC already possesses the most important advantage of Lagrangian system: sharp resolution of contact discontinuities (see Sections 7.4 and 7.5). To require the remaining coordinate also to be material would then inherit the defect of Lagrangian coordinates: computation breakdown due to cell deformation with the fluid. Instead, we choose the remaining coordinate in such a way as to preserve the mesh angles (and hence also mesh orthogonality) in 2-D or to preserve mesh-skewness in 3-D, thus preventing computation breakdown.

7.3   Relation to Arbitrary-Lagrangian-Eulerian Computation

The unified coordinates approach shares the same spirit of the Arbitrary-Lagrangian-Eulerian (ALE) approach in that it combines the best features of the Lagrangian

and Eulerian approach[1], or in that the coordinates move at an arbitrary velocity[2]. However, the strategies are quite different. In the ALE approach, "the general strategy is to perform a Lagrangian time step and to follow it with a re-map step that maps the solution from the distorted Lagrangian mesh on to the spatially fixed Eulerian mesh or the ALE mesh"[1, 2]. This is usually done by employing a staggered mesh. Furthermore, the rezoning/remapping strategies are not generally prescribed; instead, "rezoning requires the intervention of the user, · · ·, and the success of the method depends heavily on the skill and patience of the user"[2]. In our UC approach we propose a specific rule for mesh movement (see Sections 7.4, 7.5 and 7.8 below), and in our method all computations are done entirely in the transformed space without a staggered mesh and with no explicit rezoning/remapping to the Eulerian or ALE space; explicit re-mapping causes numerical diffusion.

7.4   Contact Resolution

We now explain in this and the next section how to determine the mesh velocity so as to get sharp resolution of contacts. Consider the 2-D case first. In this case there are two arbitrary functions, U and V, and we can prescribe two requirements. The first requirement is that coordinate η shall be a material coordinate, meaning

$$\frac{D_q \eta}{Dt} = 0. \tag{7.4}$$

Together with $D_{\boldsymbol{Q}} \eta / Dt = 0$ in (6.2), we get

$$Y = (v - V)A - (u - U)B = 0. \tag{7.5}$$

We observe that contact lines, being material lines, must coincide with coordinate lines and, therefore, can be resolved sharply. Moreover, a material interface (including a free surface) corresponds to $\eta =$const. and thus can be resolved sharply.

7.5   Mesh Orthogonality

The second requirement is that mesh angles, and hence mesh orthogonality, shall be preserved during the λ-marching computation. This means

$$\frac{\partial}{\partial \lambda} \arccos \left(\frac{\nabla \xi \cdot \nabla \eta}{|\nabla \xi||\nabla \eta|} \right) = \frac{\partial}{\partial \lambda} \arccos \left(\frac{AL + BM}{\sqrt{A^2 + B^2}\sqrt{L^2 + M^2}} \right) = 0, \tag{7.6}$$

which, after expressing V in terms of U from (7.5) and using the geometric conservation laws of (6.19), yields an ODE for U

$$\frac{\partial U}{\partial \eta} + P(\eta; \lambda, \xi)U = Q(\eta; \lambda, \xi), \tag{7.7}$$

$$U(\eta; \lambda, \xi) = U_0(\lambda, \xi) \quad \text{at} \quad \eta = \eta_0, \tag{7.8}$$

where $U_0(\lambda, \xi)$ can be prescribed arbitrarily, and

$$P(\eta; \lambda, \xi) = \frac{S^2}{T^2 \Delta}\left(A\frac{\partial B}{\partial \xi} - B\frac{\partial A}{\partial \xi}\right) - \frac{L}{A\Delta}\left(A\frac{\partial B}{\partial \eta} - B\frac{\partial A}{\partial \eta}\right),$$

$$Q(\eta; \lambda, \xi) = \frac{S^2 A}{T^2 \Delta}\left(B\frac{\partial u}{\partial \xi} - A\frac{\partial v}{\partial \xi}\right) + \frac{L}{\Delta}\left(A\frac{\partial v}{\partial \eta} - B\frac{\partial u}{\partial \eta}\right) + uP(\eta; \lambda, \xi)$$

with

$$\Delta = AM - BL, \quad S^2 = L^2 + M^2, \quad T^2 = A^2 + B^2.$$

We note that the mesh-angle preserving condition is not unique, but can be replaced by any reasonable condition, e.g., preserving the Jacobian Δ, which would be particularly suitable for incompressible flow computation.

Remark   (1) Condition (7.4) alone inherits the advantages of Lagrangian coordinates: sharp resolution of contacts. If we further require

$$\frac{D_q \xi}{Dt} = 0, \tag{7.9}$$

then the coordinates (ξ, η) are Lagrangian and computation may break down due to large cell deformation. This can be prevented by the mesh-angle preserving condition (7.7), or by the Jacobian-preserving condition. The UC system may be regarded as a generalization of the Lagrangian system in that we retain only one condition of Lagrangianess (7.4), while abandoning the other one (7.9) in favor of a mesh-angle preserving condition, or Jacobian preserving condition. The same may be said about the 3-D case (see remark (2) below).

(2) In the case of 3-D flow, there are three arbitrary functions U, V and W, so we should prescribe three requirements: We want η and ζ to be material coordinates, meaning

$$\frac{D_q \eta}{Dt} = 0 \tag{7.10}$$

and

$$\frac{D_q \zeta}{Dt} = 0. \tag{7.11}$$

(7.10) and the second equation of (6.2) combine to give

$$Y = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_2 = 0. \tag{7.12}$$

Similarly, (7.11) and the third equation of (6.2) combine to give

$$Z = (\boldsymbol{q} - \boldsymbol{Q}) \cdot \boldsymbol{J}_3 = 0. \tag{7.13}$$

As the third condition, we require that the mesh skewness

$$\kappa = \frac{|\boldsymbol{A}| \cdot |\boldsymbol{L}| \cdot |\boldsymbol{P}|}{\boldsymbol{A} \times \boldsymbol{L} \cdot \boldsymbol{P}} - 1 = \begin{cases} 0, & \text{orthogonal}, \\ \infty, & \text{degenerate} \end{cases} \tag{7.14}$$

be preserved during the λ-marching computation[3]. Here $\boldsymbol{A} = (A, B, C)^{\mathrm{T}}, \boldsymbol{L} = (L, M, N)^{\mathrm{T}}$ and $\boldsymbol{P} = (P, Q, R)^{\mathrm{T}}$. This condition is

$$\frac{\partial \kappa}{\partial \lambda} = 0. \tag{7.15}$$

It might seem a possible alternative (to the three conditions (7.10), (7.11) and (7.15)) to use (7.10) and to preserve the two mesh angles during the λ- marching computation. This would then have the desirable effect that mesh orthogonality can also be preserved for 3-D flow (as in the 2-D case) when the mesh is initially orthogonal. However, this is impossible in general, because it would contradict a theorem of [4] (see also Section 9.5) that for steady flow past a body, orthogonal body-fitted mesh is possible if and only if the flow belongs to a special class called complex-lamellar[5], for which $\boldsymbol{q} \cdot \nabla \times \boldsymbol{q} \equiv 0$. We also note that, as in the 2-D case, the condition of mesh skewness-preserving can be replaced by that of Jacobian-preserving.

7.6   Unified Coordinates for Steady Flow

In the special case of steady flow, (7.10) and (7.11) become $\boldsymbol{q} \cdot \nabla \eta = 0$ and $\boldsymbol{q} \cdot \nabla \zeta = 0$. Hence, fluid velocity vector $\boldsymbol{q}$ lies in the direction of the intersection line between the surfaces $\eta(x, y, z) =$const. and $\zeta(x, y, z) =$const.. At the same time, the second and the third equations of (6.2) also become $\boldsymbol{Q} \cdot \nabla \eta = 0$ and $\boldsymbol{Q} \cdot \nabla \zeta = 0$, and hence the mesh velocity vector $\boldsymbol{Q}$ must lie along the same intersection line. Therefore,

$$\boldsymbol{Q} = h\boldsymbol{q}, \tag{7.16}$$

where h can be an arbitrary function. This means the unified coordinates move in the direction of fluid velocity but not with its speed, in contrast to the Lagrangian coordinates which move with the fluid both in its direction and speed. This special case of the UC will be studied in details in Chapter 9.

7.7   Effects of Mesh Movement on the Flow

In Sections 7.4~7.6, we paid emphasis on sharp resolution of contacts. On the other hand, if contact resolution is not an important issue, as in some applications, there is no need to require any of the coordinates in UC to be material. In such cases we can freely choose the mesh velocity (U, V, W) to our benefits and regard the UC as a moving mesh method. As such, it distinguishes from existing moving mesh methods in that the effects of mesh movement on the flow are now fully accounted for through the geometric conservation laws which, together with the physical conservation laws, form a closed system of PDE in conservation form (equation (6.19)). An example is given in Chapter 12.

7.8   Relation to Other Moving Mesh Methods

For the numerical solution of partial differential equations, the adaptive or moving mesh methods can be roughly divided into two categories, i.e., the static and dynamic ones. For the static method, a new mesh is generated according to some monitor function after every or several time steps in the calculation and all flow variables are interpolated to the new mesh at a fixed time. By contrast, the UC uses only one mesh, and therefore does not need interpolation of flow variables and geometries between two meshes.

For the dynamic moving mesh methods, a mesh equation that involves mesh velocity is employed to move a mesh within each time step in such a way that the mesh becomes concentrated in regions of rapid variation of the solution. At the same time, the original governing equations are transformed into the the moving mesh space. In this regard, the dynamic moving mesh method is closely related to the UC method, even though they are targeting on the different equations. For example, the dynamic moving mesh method is mostly used for the scalar equations, and the UC method is solely solving the fluid system.

The mesh equation and the original differential equation are solved simultaneously for the dynamic moving mesh method. The moving finite element method[6] and moving finite difference method[7] have aroused considerable interest. Other representative research in this aspect is the work by Russell, Huang, et al[8−9]. Different from the UC method, the mesh movement in this approach is usually based on the error measure or error equi-distribution principle for the determination of moving mesh equation[11−15]. But, even with so many years research, it is still surprisingly difficult to describe consistently reliable moving mesh equations.

In 1-D case, in order to keep the equal-partition of the mesh, one of the criterion of the moving mesh method is to keep $\mathcal{M}\partial x/\partial \xi$ as a constant, where $\mathcal{M}$ is the monitor function to determine the distribution of the mesh. Therefore, we have

$$\frac{\partial}{\partial \xi}\left(\mathcal{M}\frac{\partial x}{\partial \xi}\right) = 0. \tag{7.17}$$

In order to determine the mesh movement, we can make a time derivative to the above equation,

$$\frac{\partial}{\partial \lambda}\left[\frac{\partial}{\partial \xi}\left(\mathcal{M}\frac{\partial x}{\partial \xi}\right)\right] = 0,$$

from which the mesh velocity can be obtained implicitly,

$$\frac{\partial^2}{\partial \xi^2}(\mathcal{M}U) = -\frac{\partial}{\partial \xi}\left(\frac{\partial \mathcal{M}}{\partial \lambda}\frac{\partial x}{\partial \xi}\right),$$

where $U = \partial x/\partial \lambda$ is the mesh velocity and $A = \partial x/\partial \xi$. With similar consideration and the introduction of a relaxation time τ to determine the time scale to settle down the optimal mesh with equal-distribution, a series of moving mesh velocity equations have been obtained, such as

$$\frac{\partial}{\partial \xi}\left(\mathcal{M}\frac{\partial U}{\partial \xi}\right) = -\frac{1}{\tau}\frac{\partial}{\partial \xi}(\mathcal{M}A),$$

$$U = \frac{1}{\tau}\frac{\partial}{\partial \xi}\frac{\partial}{\partial \xi}(\mathcal{M}A). \tag{7.18}$$

In principle, the UC method can adapt the above moving mesh equation as well in the determination of the mesh velocity U. In other words, the UC approach can be unified with the above dynamic moving mesh method. Based on the transformation between (t, x) and (λ, ξ) space in the UC approach, the compatibility condition (geometrical conservation laws) is given explicitly. However, for the dynamic moving mesh method, there is no distinction between the difference t and λ,

and the geometrical conservation law is implicitly satisfied for the scalar equation in 1-D case. But, solely designing a mesh velocity without explicitly conservation forms and geometrical conservation laws will meet great barriers in its development and applications for multidimensional fluid system.

Even though the above moving mesh methods cannot be applicable to gas dynamic equations, the definition of mesh velocity may become a supplement to the Lagrangian gas dynamics and be implemented in the UC system. For example, (7.18) is more likely to define a diffusive mesh velocity associated to the elliptic system, which can be used to prevent the mesh deformation. Therefore, with proper choice of the monitor function $\mathcal{M}$, and construct the diffusive mesh velocity in 2-D, such as

$$U_d = \frac{1}{\tau} \frac{\partial^2}{\partial \xi^2}(\mathcal{M}A), \quad V_d = \frac{1}{\tau} \frac{\partial^2}{\partial \eta^2}(\mathcal{M}M)$$

with $M = \partial y / \partial \eta$, we can define a generalized Lagrangian type mesh velocity

$$U = u + U_d, \quad V = v + V_d,$$

which not only follow the fluid velocity, but also adjust itself to smoothing the mesh irregularity. A recent study of moving mesh method with dissipative mesh velocity is presented in [16].

7.9   Relation to Mesh Generation and the Level-Set Function Method

The UC system can also be used in other frameworks, such as automatic mesh generation and implementation of level set function. As the body surface is a material surface, condition (7.5) guarantees that the mesh in UC is automatically a body-fitted mesh at all time. This provides the foundation for automatic mesh generation (see Section 10.2 for implementation details).

The function $\eta(x, y, t)$ given by (7.4) is a level set function, hence there is no need to introduce an extra level set function when using the level set method.

References

[1] L.G. MARGOLIN. Introduction to an arbitrary Lagrangian-Eulerian computing method for all flow speeds. *J. Comput. Phys.*, 135: 198-202, 1997.

[2] D. J. BENSON. Computational methods in Lagrangian and Eulerian hydrocodes. *Comput. Method. Appl. Mech. Engrg.*, 99: 235-394, 1992.

[3] W.H. HUI AND S. KUDRIAKOV. A unified coordinate system for solving the three-dimensional Euler equations. *J. Comput. Phys.*, 172: 235-260, 2001.

[4] W.H. HUI AND Y. HE. Hyperbolicity and optimal coordinates of the three-dimensional steady Euler equations. *SIAM Journal on Applied Mathematics*, 57: 893-928, 1997.

[5] C. ROGERS, W.K. SCHIEF AND W.H. HUI. On complex-lamellar motion of a Prim gas. *J. Math Anal. Appl.*, 266: 55-69, 2002.

[6] K. MILLER AND R.N. MILLER. Moving finite element. *SIAM J. Numer Anal.*, 18: 1019-1032, 1981.

[7] E.A. DORFI AND L. O'c. DRURY. Simple adaptive grids for 1-D initial value problems. *J. Comput. Phys.*, 69: 175-195, 1987.

[8] W. HUANG, Y. REN AND R.D. RUSSELL. Moving mesh methods based upon moving mesh partial differential equations. *J. Comput. Phys.*, 11: 279-290, 1994.

[9] W. HUANG AND R.D. RUSSELL. Adaptive mesh movement — the MMPDE approach and its applications. *J. Comput. Appl. Math.*, 128: 383-398, 2001.

[10] W. HUANG, J. MA AND R.D. RUSSELL. A study of moving mesh PDE methods for numerical simulation of blowup in reaction diffusion equations. *J. Comput. Phys.*, (2008), doi: 10.1016/j.jcp.2008.03.024.

[11] S. ADJERID AND J.E. FLAHERTY. A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations. *SIAM J. Numer. Anal.*, 23: 778-795, 1986.

[12] B.M. HERBST, S.W. SCHOOMBIE AND A.R. MITCHELL. Equidistributing principles in moving mesh finite element methods. *J. Comput. Appl. Math.*, 9: 377-389, 1983.

[13] S.F. DAVIS AND J.E. FLAHERTY. An adaptive finite element method for initial-boundary value problems for partial differential equations. *SIAM J. Sci. Stat. Comput.*, 3: 6-27, 1982.

[14] R.G. HINDMAN AND J. SPENCER. A new approach to truly adaptive grid generation. *AIAA Paper 83-0450*, 1983.

[15] Y. REN AND R.D. RUSSELL. Moving mesh techniques based upon equidistribution and their stability. *SIAM J. Sci. Statis. Comput.*, 13: 1265-1286, 1992.

[16] G. NI, S. JIANG AND K. XU. Remapping-free ALE-type kinetic method for flow computations. *J. Comput. Phys.*, 228: 3154-3171, 2009.

Chapter 8

Lagrangian Gas Dynamics

8.1　Lagrangian Gas Dynamics Equations

For simplicity we consider 2-D flow, when (6.1) simplifies to

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta, \end{cases} \tag{8.1}$$

the governing equations (6.19) also reduce to (6.21). In the special case of Lagrangian coordinates, (6.21) simplifies to

$$\frac{\partial \boldsymbol{E}}{\partial \lambda} + \frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta} = 0, \tag{8.2}$$

$$\boldsymbol{E} = \begin{pmatrix} \rho\Delta \\ \rho\Delta u \\ \rho\Delta v \\ \rho\Delta e \\ A \\ B \\ L \\ M \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} 0 \\ pM \\ -pL \\ p(uM - vL) \\ -u \\ -v \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{G} = \begin{pmatrix} 0 \\ -pB \\ pA \\ p(vA - uB) \\ 0 \\ 0 \\ -u \\ -v \end{pmatrix}. \tag{8.3}$$

The first four equations of (8.3) are the physical conservation laws and the last four are the geometric conservation laws.

We remark that this system of Lagrangian gas dynamics equations (8.2) in **conservation PDE form** was given for the first time for 2-D flow in [1], and for 3-D flow in [2]. It is this conservation form PDE that guarentees that shocks can be captured correctly and provides a foundation for designing schemes that are moving mesh schemes in Eulerian space. Indeed, the first such scheme has just been proposed by Despres and Mazeran[3], who ingeniously incorporate the remaining compatibility conditions

$$\frac{\partial A}{\partial \eta} = \frac{\partial L}{\partial \xi}, \quad \frac{\partial B}{\partial \eta} = \frac{\partial M}{\partial \xi} \tag{8.4}$$

to rewrite (8.2) into a canonical form.

8.2   Weak Hyperbolicity

Whereas system (6.21) is strongly hyperbolic like the Eulerian case, the Lagrangian system (8.2) is found in [1] to be only weakly hyperbolic, meaning that while all eight eigenvalues are real, there does not exist a complete set of eight linearly independent eigenvectors. This finding is also confirmed in [3].

Since this finding is quite striking, it is necessary here to say more about the eigen-systems of (8.2). To study the hyperbolicity of (8.2), we rewrite it as

$$A\frac{\partial U}{\partial \lambda} + B\frac{\partial U}{\partial \xi} + C\frac{\partial U}{\partial \eta} = S, \tag{8.5}$$

where

$$U = (\rho, p, u, v, A, B, L, M)^{\mathrm{T}},$$
$$A = \frac{\partial E}{\partial U}, \quad B = \frac{\partial F}{\partial U}, \quad C = \frac{\partial G}{\partial U},$$

and S is the source term due to the transformation (8.1), which has no contribution to the eigen-field. System (8.5) is said to be hyperbolic (also called strongly hyperbolic, or fully hyperbolic) in λ if

(1) all eigenvalues σ of

$$\det(\sigma A - \alpha B - \beta C) = 0$$

are real for every pair $(\alpha, \beta) \in \mathbf{R}^2, \alpha^2 + \beta^2 = 1$;

(2) associated with the eigenvalues there exists a complete set of eight linearly independent right eigenvectors in the state space.

Now, from direct computation we find that the eigenvalues of (8.5) are

$$\sigma_0 = 0 \quad (\text{multiplicity } 6),$$
$$\sigma_{\pm} = \pm a\sqrt{\alpha'^2 + \beta'^2}, \tag{8.6}$$

where multiplicity 6 means 6 eigerwalues have the same value, $\alpha' = (\alpha M - \beta B)/\Delta, \beta' = -(\alpha L - \beta A)/\Delta$. For $\sigma_{\pm}$, the associated eigenvectors are

$$r_{\pm} = \left(1, \frac{1}{a^2}, \frac{\alpha'}{\rho \sigma_{\pm}}, \frac{\beta'}{\rho \sigma_{\pm}}, \frac{-\alpha \alpha'}{\rho(\sigma_{\pm})^2}, \frac{-\alpha' \beta}{\rho(\sigma_{\pm})^2}, \frac{-\beta \beta'}{\rho(\sigma_{\pm})^2}, \frac{-\alpha \alpha'}{\rho(\sigma_{\pm})^2}\right)^{\mathrm{T}}. \tag{8.7}$$

Associated with the eigenvalue $\sigma_0 = 0$,

$$\text{rank}(\sigma \boldsymbol{A} - \alpha \boldsymbol{B} - \beta \boldsymbol{C})|_{\sigma=\sigma_0} = 3,$$

so we get only 5 linearly independent eigenvectors:

$$\begin{aligned}
\boldsymbol{r}_1 &= (0, 0, 0, 0, 1, 0, 0, 0)^{\mathrm{T}}, \\
\boldsymbol{r}_2 &= (0, 0, 0, 0, 0, 1, 0, 0)^{\mathrm{T}}, \\
\boldsymbol{r}_3 &= (0, 0, 0, 0, 0, 0, 1, 0)^{\mathrm{T}}, \\
\boldsymbol{r}_4 &= (0, 0, 0, 0, 0, 0, 0, 1)^{\mathrm{T}}, \\
\boldsymbol{r}_5 &= (0, 1, 0, 0, 0, 0, 0, 0)^{\mathrm{T}}.
\end{aligned} \tag{8.8}$$

This means that associated with the eigenvalue $\sigma = 0$ (multiplicity 6) there exist 5, and only 5, linearly independent eigenvectors. Consequently, the Lagrangian system (8.2) does not have a complete set of 8 linearly independent eigenvectors — one eigenvector is missing—and is therefore weakly hyperbolic. We note that in [3] the free divergence constraints (8.4) were used to rewrite the system (8.2) (mainly the mass conservation equation) into canonical form, and it was then found that three (two in the special case) eigenvectors are missing, although the eigenvalues are the same as (8.6). Nevertheless, the conclusion is the same: Lagrangian gas dynamics system (8.2) is weakly hyperbolic, which appears to be an intrinsic property of the Lagrangian formulation due to shear discontinuities.

8.3   Non-Equivalency of Lagrangian and Eularian Formulation

Similar to 2-D flow, the 3-D Lagrangian gas dynamics system is also found to be weakly hyperbolic[2]. This implies that for 2-D and 3-D flow, **Lagarangian system of gas dynamics is not equivalent to the Eulerian system** (for the special case of steady flow, this finding was first reported in [4]). In particular, the Cauchy problem for weakly hyperbolic systems is not well posed in the sense of Hadamard; it is well-posed only in a weaker sense, compared with strong hyperbolic systems.

This seems surprising, in view of the fact that Eulerian system of gas dynamics equations is long known to be strongly hyperbolic. However, it has been

demonstrated[3] that Lagrangian re-formulation of any Eulerian hyperbolic system will generically lead to a weakly hyperbolic system. A simple example is given here.

Consider the inviscid hyperbolic Burgers equation

$$u_t + uu_x = 0. \tag{8.9}$$

The Lagrangian transformation

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = u\mathrm{d}\lambda + A\mathrm{d}\xi \end{cases} \tag{8.10}$$

changes (8.9) into

$$\begin{pmatrix} u \\ A \end{pmatrix}_\lambda + \begin{pmatrix} 0 & 0 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} u \\ A \end{pmatrix}_\xi = 0. \tag{8.11}$$

The eigenvalues of the system (8.11) are 0 and 0, but there exists only one linearly independent right eigenvector $(0, 1)^{\mathrm{T}}$. Hence (8.11) is weakly hyperbolic.

In this regards, the 1-D Lagrangian system of gas dynamics equations (4.3) is just an exception: it happens to be strongly hyperbolic.

References

[1] W.H. Hui, P.Y. Li and Z.W. Li. A unified coordinate system for solving the two-dimensional Euler equations. *J. Comput. Phys.*, 153: 596-637, 1999.

[2] W.H. Hui and S. Kudriakov. A unified coordinate system for solving the three-dimensional Euler equations. *J. Comput. Phys.*, 172: 235-260, 2001.

[3] B. Despres and C. Mazeran. Lagrangian gas dynamics in two dimensions and Lagrangian systems. *Arch. Rathion. Mech. Anal.*, 178: 327-372, 2005.

[4] W.H. Hui and Y.C. Zhao. A generalized Lagrangian method for solving the Euler equations. in *Nonlinear Hyperbolic Problems: Theoretical, Applied, and Computational Aspects*, (Eds.) Donato, A. and Oliveri, F., Vieweg, 336, 1993.

Chapter 9
Steady 2-D and 3-D Supersonic Flow

9.1 The Unified Coordinates for Steady Flow

As shown in Section 7.6, in the case of steady flow, for two of the unified coordinates (λ, ξ, η) to be material coordinates, the mesh velocity must be parallel to the fluid velocity, i.e., $(U, V, W) = h(u, v, w)$. Therefore, transformation (6.6) becomes

$$\begin{cases} \mathrm{d}x = hu\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = hv\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta, \\ \mathrm{d}z = hw\mathrm{d}\lambda + C\mathrm{d}\xi + N\mathrm{d}\eta, \end{cases} \tag{9.1}$$

where h is to be determined later, and u, v, w are the x-, y- and z-component of fluid velocity $\boldsymbol{q}$. It is easy to see that

$$\frac{D\xi}{Dt} = 0,$$

$$\frac{D\eta}{Dt} = 0. \tag{9.2}$$

Hence, ξ and η are stream functions in steady flow.

The geometrical state variables $\boldsymbol{T} = (A, B, C)^{\mathrm{T}}$ and $\boldsymbol{S} = (L, M, N)^{\mathrm{T}}$ satisfy the compatibility conditions,

$$\frac{\partial \boldsymbol{T}}{\partial \lambda} - \frac{\partial (h\boldsymbol{q})}{\partial \xi} = \boldsymbol{0},$$

$$\frac{\partial \boldsymbol{S}}{\partial \lambda} - \frac{\partial (h\boldsymbol{q})}{\partial \eta} = \boldsymbol{0}. \tag{9.3}$$

In the special case of steady 2-D or axisymmetric flow, transformation (9.1) reduces to

$$\begin{cases} \mathrm{d}x = hu\mathrm{d}\lambda + A\mathrm{d}\xi, \\ \mathrm{d}y = hv\mathrm{d}\lambda + B\mathrm{d}\xi. \end{cases} \tag{9.4}$$

Here ξ is a stream function and the arbitrary function h is chosen so that the mesh λ, ξ is orthogonal (see (9.8)).

9.2   Euler Equations in the Unified Coordinates

The Euler equations for 2-D and axisymmetric steady flow in Cartesian coordinates are

$$\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\\rho u^2 + p\\\rho uv\\\rho ue\end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix}\rho v\\\rho uv\\\rho v^2 + p\\\rho vH\end{pmatrix} = -\frac{\alpha}{y}\begin{pmatrix}\rho v\\\rho uv\\\rho v^2\\\rho vH\end{pmatrix}, \tag{9.5}$$

where p is the pressure, $e = \frac{1}{2}(u^2 + v^2) + \frac{1}{\gamma - 1}\frac{p}{\rho}$ and $H = e + \frac{p}{\rho}$. Here $\alpha = 0$ denotes 2-D flow and $\alpha = 1$ denotes axisymmetric flow.

The eigenvalues of (9.5), with respect to the x-direction, are

$$\sigma_0 = \frac{v}{u} \quad \text{(multiplicity 2)},$$

$$\sigma_\pm = \frac{uv \pm a^2\sqrt{M^2 - 1}}{u^2 - a^2},$$

where a is the speed of sound and $M = \sqrt{u^2 + v^2}/a$ is the flow Mach number. System (9.5) is known to be hyperbolic for supersonic flow: $M > 1$. However, the initial value problem for (9.5) is well-posed if and only if

$$\frac{u}{a} > 1, \tag{9.6}$$

which requires the flow in the marching direction, x, be supersonic. It is, therefore, a stronger condition than simply $M > 1$.

Under transformation (9.4), (9.5) becomes

$$\frac{\partial}{\partial \lambda}\begin{pmatrix}K\\H\\Ku + pB\\Kv - pA\\A\\B\end{pmatrix} + \frac{\partial}{\partial \xi}h\begin{pmatrix}0\\0\\-pv\\pu\\-u\\-v\end{pmatrix} = -\frac{\alpha hKv}{y}\begin{pmatrix}1\\0\\u\\v\\0\\0\end{pmatrix}, \tag{9.7}$$

where $K = \rho(uB - vA)$.

We note that while the first 4 equations of (9.7), which are the physical conservation laws transformed from (9.5), are in conservation form they do not form a closed system as they contain the additional unknowns A and B. It is by appending the compatibility conditions of the transformation (the last 2 equations of (9.7))

that they form a closed system of PDE in conservation form. These 2 compatibility conditions are called geometric conservation laws.

The free function h is chosen to ensure mesh orthogonality. This means that $uA + vB = 0$ for all λ, which gives[1]

$$h = \exp \int^\xi \frac{1}{\rho q^2} \frac{\partial p}{\partial \xi} \mathrm{d}\xi / q. \tag{9.8}$$

We note that the system of equations (9.7) is in conservation form, so any shock-capturing method can be employed to solve them. Similar to 1-D flow in Chapter 4, with the application of shock-adaptive Godunov scheme[2, 3] in which the shock wave is fitted exactly using the Riemann solution without additional cost, the energy equation can be replaced by conservation of entropy $S = p/\rho^\gamma$ along a streamline in smooth flow regions. We also use the mesh orthogonality condition (9.8), which is equivalent to $uA + vB = 0$ for all λ, to eliminate A. Equations (9.7) are then simplified to

$$\frac{\partial \boldsymbol{F}}{\partial \lambda} + \frac{\partial \mathbf{G}}{\partial \xi} = \boldsymbol{S}, \tag{9.9}$$

$$\boldsymbol{F} = \begin{pmatrix} K \\ H \\ S \\ Ku + pB \\ Kv - \dfrac{pvB}{u} \end{pmatrix}, \quad \mathbf{G} = h \begin{pmatrix} 0 \\ 0 \\ 0 \\ -pv \\ pu \end{pmatrix}, \quad \boldsymbol{S} = -\frac{\alpha h K v}{y} \begin{pmatrix} 1 \\ 0 \\ 0 \\ u \\ v \end{pmatrix}. \tag{9.10}$$

The eigenvalues of (9.9) are[1]

$$\sigma_0 = 0 \ \text{(multiplicity 3)},$$
$$\sigma_\pm = \pm \frac{hq \tan \mu}{T},$$

where $\mu = \arcsin \dfrac{1}{M}$ and $T = \sqrt{A^2 + B^2}$.

It is shown[4] that (9.9) is hyperbolic if and only if the flow is supersonic, i.e.,

$$M > 1. \tag{9.11}$$

It is further shown that condition (9.11) also ensures well-posedness of the initial value problem for (9.9). We should also remark that under transformation (9.4), which involves the dependent variables u and v, there is no guarantee that the

hyperbolic system (9.5) will be transformed to a system of the same type. The fact that the transformed system (9.9) turns out to be also hyperbolic must be regarded as fortuitous.

9.3   The Space-Marching Computation

(9.9) is solved using the fractional step method as follows:

   Step 1   Solve $\partial \boldsymbol{F}/\partial \lambda + \partial \boldsymbol{G}/\partial \xi = \boldsymbol{0}$ to get

$$\boldsymbol{F}^* = \boldsymbol{F}_j^n - \frac{\Delta \lambda}{\Delta \xi}[\widetilde{\boldsymbol{G}}_{j+1/2}(\boldsymbol{F}^n) - \widetilde{\boldsymbol{G}}_{j-1/2}(\boldsymbol{F}^n)]. \tag{9.12}$$

   Step 2   Solve $\partial \boldsymbol{F}/\partial \lambda = \boldsymbol{S}$ to get

$$\boldsymbol{F}^{n+1} = \boldsymbol{F}_j^n + \Delta \lambda \boldsymbol{S}(\boldsymbol{F}^*). \tag{9.13}$$

In Step 1, we use the adaptive Godunov scheme[2, 3], where $\widetilde{\boldsymbol{G}}_{j+1/2}$ is Godunov flux.

   The procedure of space-marching computation for flow past a body is as follows:

   (1) Begin with a column of uniform orthogonal cells, where the uniform flow is given and is perpendicular to the column.

   (2) Solve (9.9) by marching in λ with the Godunov-MUSCL scheme. After one step $\Delta \lambda$, the initial column of cells moves to the right by $hq_\infty \Delta \lambda$, where q_∞ is the velocity of the uniform free stream.

   (3) After several steps when the initial column of cells has moved to the right by Δx, the given cell width, add one new column of cells on the left that is identical to the initial column.

   (4) Repeat the above process and, when the cells meet the body surface, impose the boundary conditions there. Since the body surface is a streamline, it corresponds to a coordinate line and, therefore it is very easy to satisfy boundary condition there.

   (5) Continue this process until the columns of cells cover the whole body surface. This then completes the wanted computation.

   (6) (optional) The mesh at any step λ can be constructed (and plotted) by integrating (9.4).

   Remark   (1) With the use of the unified coordinates, only the two momentum equations in (9.9) need be solved, but the conservation laws of mass, energy and

entropy (in smooth flow regions) are satisfied exactly. This is to be compared with Eulerian computation, in which all four equations in (9.5) need be solved. So use of the unified coordinates not only produces more accurate results but also saves computing time.

(2) In principle, a space-marching method can also be employed using Eulerian coordinates. However, in order to do that, a body-fitted mesh must be generated prior to the marching in order to satisfy the boundary condition correctly. It is known that even after thirty years of research, mesh-generation remains a major bottleneck of CFD, as it is tedious, time-consuming and requires experiences and special training. By contrast, space-marching based on the unified coordinates requires no mesh-generation prior to flow computation, although the flow-generated mesh can be constructed if desired. This avoids the cumbersome mesh-generation stage in Eulerian computation, thus greatly saving computing time.

(3) What is more important is that even after mesh-generation, a space-marching method using Eulerian coordinates will fail when the flow in the marching direction, x say, is subsonic, even though the flow is supersonic everywhere (see Example 2 below). By contrast, in computations using the unified coordinates, the space-marching is along the local flow direction (streamlines), hence it always succeeds, providing only that the flow is supersonic everywhere.

9.4   Examples

Example 1[1]   This is a Riemann problem formed with two intersecting uniform streams making an angle of 23 degrees as shown in the Figure 9.1. This extremely difficult problem was posed by Glaz and Wardlaw[5] and attempts were made to find its solution using Eulerian computation. Unfortunately, the solution given in [5] was incorrect. In our computation, we use 20 uniform cells with $T = 0.5$ for the top part and 30 non-uniform cells with $T = 0.05$ for the bottom part. This small initial data of T has the effect of increasing the accuracy in the expansion flow region. The flexibility in defining the stream function ξ in UC allows us to choose different scales T for different regions. The computed result for Mach number distribution is shown in the Figure 9.2, where the very strong slip line is seen resolved sharply, a task that is almost impossible to achieve by Eulerian computation.
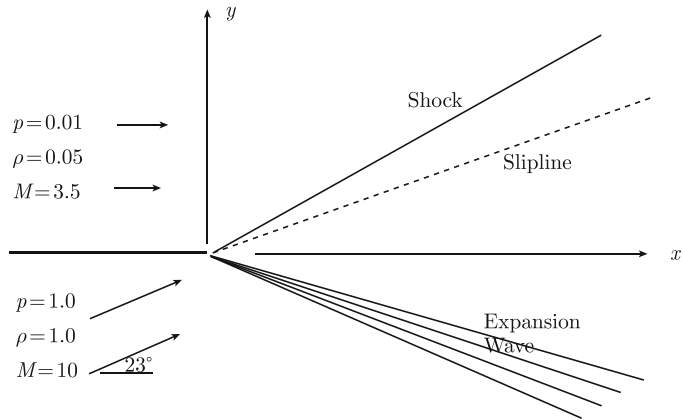
Figure 9.1   Schematic 2-D Riemann problem
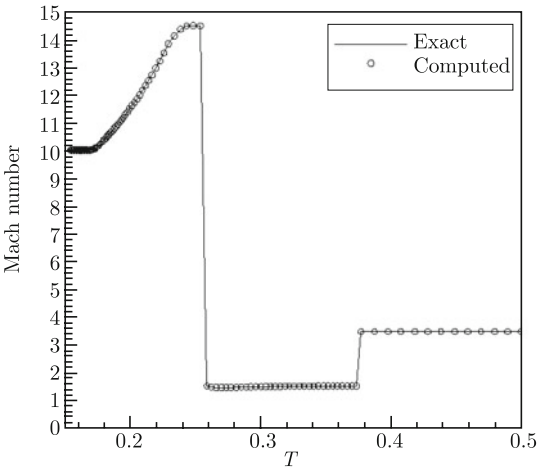
Figure 9.2   Mach number distribution

Example 2[6]   This is a supersonic flow of $M_\infty = 2.0$ past a diamond-shaped airfoil with semi-apex angle of $7.5°$ placed at an angle of attack of $10°$ (Figure 9.3).
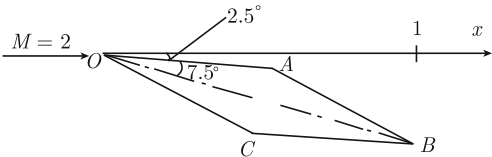
Figure 9.3   Diamond-shape airfoil

The flow-generated meshes at different λ are plotted in Figure 9.4 and Figure 9.5, and the computed pressure contours and surface Mach numbers are plotted in Figure 9.6, respectively. The computation took very little time (1.8 s on a P4, 2.8 GHz PC machine), and yet the computed results are identical to the exact solution.
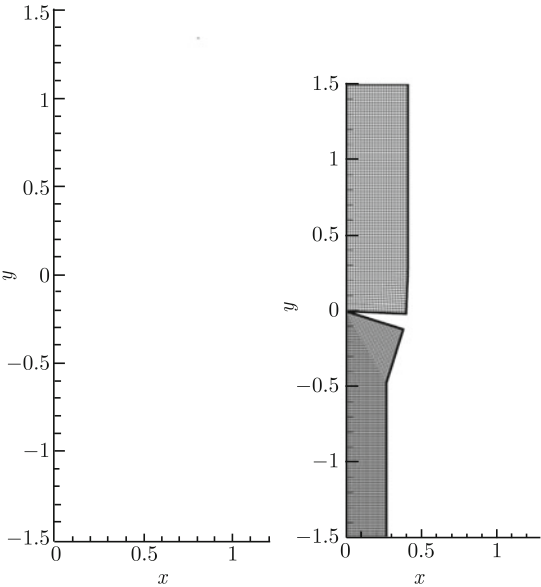
Figure 9.4   Flow-generated mesh at $\lambda = 0$ (left) and $\lambda = 0.4$ (right)
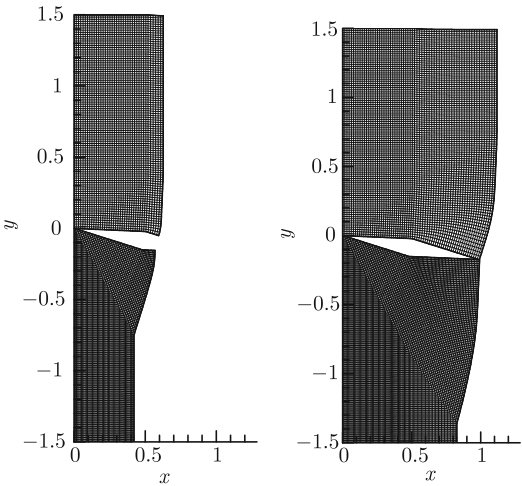
Figure 9.5   Flow-generated mesh at $\lambda = 0.6$ (left) and $\lambda = 1.0$ (right)
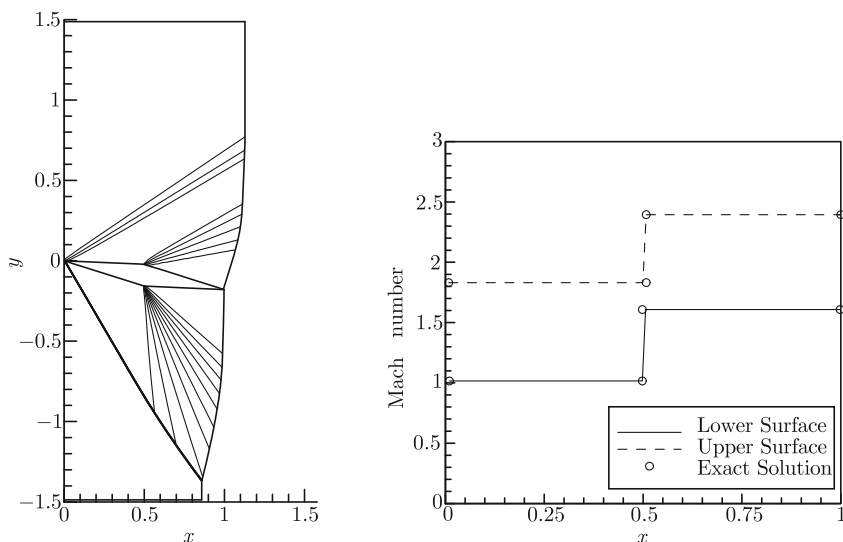
Figure 9.6   Pressure contours (left) and surface Mach number (right)

In Eulerian computation, a body-fitted mesh must be generated prior to computing the flow. We also note that in this example, a space-marching method in Eulerian coordinates fails as the x-component (in the free stream direction) of the flow behind the shock is subsonic. A time-marching method has to be used whose computing time is more than three orders of magnitude longer (2393 s, same machine), plus the time (2180 s) spent in generating a body-fitted mesh to begin the time- marching computation. Yet the computed shock and the Prandtl-Meyer expansions are smeared (Figure 9.7).

Example 3[6]   This is a hypersonic flow of $M_\infty = 5$ past a circular cone with $20°$ apex angle at zero angle of attack. This is an axisymmetric flow and its computation details are different from that for the 2-D of Example 2, where (9.13) in Step 2 is not needed. In the present axisymmetric flow, we need Step 2. In particular, (9.12) in Step 1, computes a 2-D flow of $M_\infty = 5$ past a wedge of $20°$, hence after this step the 2-D entropy of 1.056 immediately behind the shock at the apex $(x = 0)$ is higher than the corresponding value for the cone, and this higher value remains constant along the streamline, hence on the cone surface. This is obviously incorrect. However, the application of (9.13) in Step 2 gives an axisymmetric correction to the flow field, so that the entropy along streamline
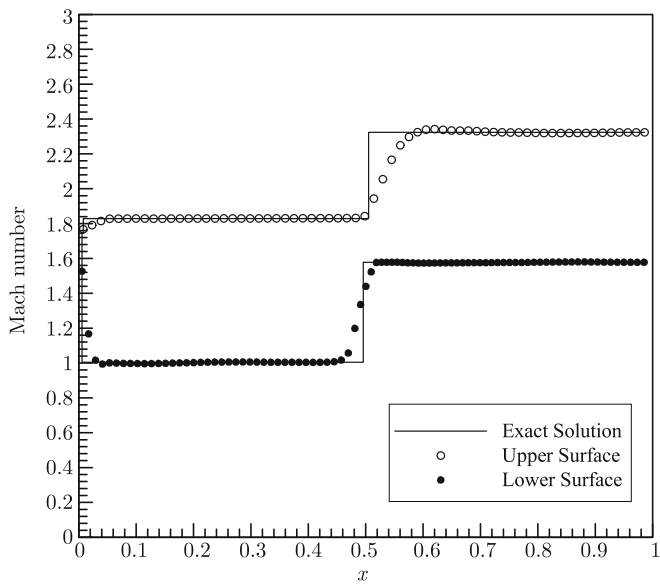
Figure 9.7   Surface Mach number distribution on diamond-shaped airfoil. Eulerian computation (5th Order WENO Scheme), 100×200 cells. Computing time to 20000 Steps: 2393 s on P4, 2.8 GHz PC machine
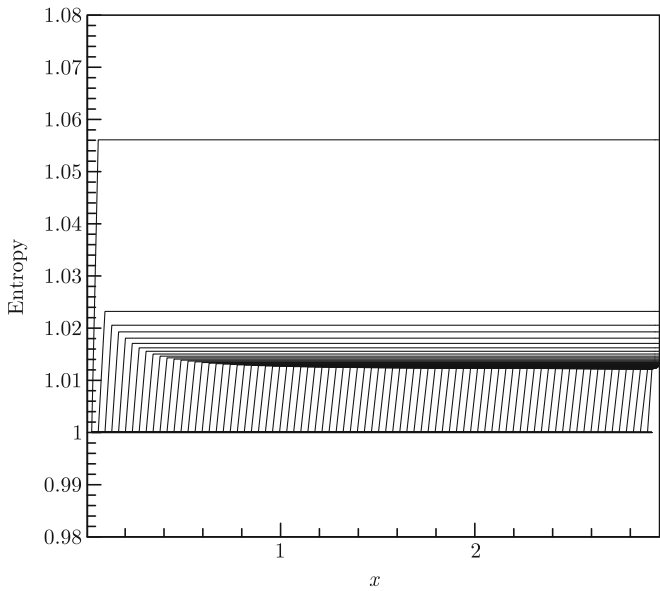
Figure 9.8   Entropy distribution along streamlines entering the shock wave at different position x

entering the shock wave at later x decreases with x and approaches the correct cone value (Figure 9.8). The initial surface entropy overshoot is similar to the famous "wall overheating" paradox of von Neumann.

The computed Mach numbers across the rays from the apex are plotted in Figure 9.9 and compared to the well-known Taylor-Maccoll solution[7]. The computing time is small (only 5% of that in Taylor-Maccoll computation), yet the agreement is excellent, especially the shock is sharply resolved.
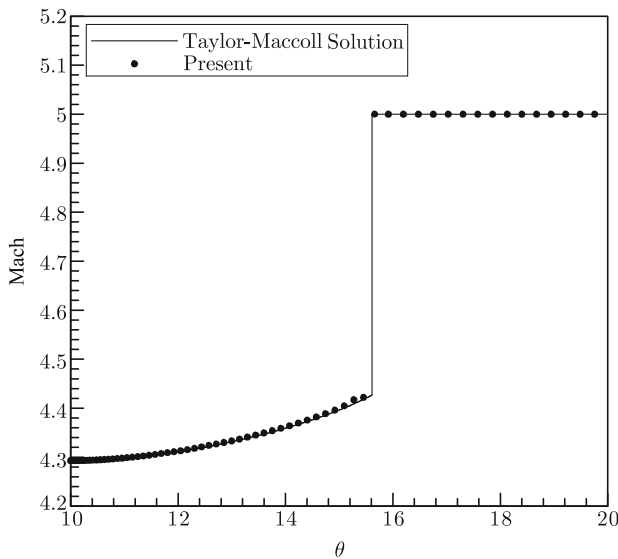
Figure 9.9   Mach number distribution, 102 cells. Computing time:

7.21 s (P4, 2.8 GHz)

As seen from these examples, for 2-D steady supersonic flow the unified coordinate computation can resolve both shock and contact sharply and, for flow past a body it does not require generating a body-fitted mesh prior to computing the flow field. Moreover, a 2-D steady supersonic flow is computed easily as if it were a 1-D unsteady flow. We conclude that the unified coordinates as defined by (9.4) plus the orthogonality condition (9.8) constitutes an optimal coordinate system for 2-D steady supersonic flow, being robust, accurate, efficient and, for flow past a body, capable of automatically generating a mesh by the flow being computed.

It is interesting to note that the UC moves with velocity $h\mathbf{q}$, it therefore follows the fluid particles only in their direction of flow, but not with their speed. We thus see that the advantage of Lagrangin coordinates is retained simply by following

the fluid particles in their direction of motion. We then adjust the speed of the coordinates so that the system is orthogonal. In this way, the disadvantage of the Lagrangian coordinates in this case, namely cell distortion, is avoided.

9.5   3-D Flow

The situation is more complicated for 3-D flow, partly due to 3-D topology. We shall first show that, unlike the 2-D case, an optimal coordinate system does not exist for 3-D flow in general.

Definition 9.1   a coordinate system (λ, ξ, η) is said to be optimal if it is orthogonal and $\nabla\lambda // \boldsymbol{q}$.

Theorem 9.1[4]   For smooth 3-D steady flow, an optimal coordinate system (λ, ξ, η) exists if and only if the flow satisfies

$$\boldsymbol{q} \cdot (\nabla \times \boldsymbol{q}) = 0. \tag{9.14}$$

Flows of this class are called complex-laminar[8].

Remark   (1) The 2-D steady gas flow is a special case of the flow for which (9.14) is satisfied; hence optimal coordinates exist for the 2-D steady supersonic flow as shown in section 9.4.

(2) Irrotational flow is another special case which satisfies the flow condition (9.14). In this case, we may simply take $\lambda = \varphi$, where φ is the velocity potential for irrotational flow, whilst ξ and η are orthogonal stream functions.

(3) When the flow condition (9.14) is not satisfied, optimal coordinates do not exist. Now we prove the above theorem.

Proof   Suppose that the optimal coordinates exist. Then

$$\nabla\lambda = k(x, y, z)\boldsymbol{q}, \quad (x, y, z) \in \Omega$$

for some function $k(x, y, z)$. That is, $k\boldsymbol{q}$ is an irrotational vector field in Ω and

$$\nabla \times (k\boldsymbol{q}) \equiv \boldsymbol{0} \quad \text{in} \quad \Omega$$

or

$$\nabla \ln k \times \boldsymbol{q} + \nabla \times \boldsymbol{q} \equiv \boldsymbol{0} \quad \text{in} \quad \Omega. \tag{9.15}$$

Multiplying $\boldsymbol{q}$ with (9.15) yields (9.14).

Conversely, suppose that (9.14) holds; then by the Pfaffian theorem, there exists an integrating factor $k = k(x, y, z)$ such that

$$k\boldsymbol{q} \cdot \mathrm{d}\boldsymbol{r} = \mathrm{d}\phi$$

is a total differential of some function $\phi = \phi(x, y, z)$. Thus

$$\nabla\phi = k\boldsymbol{q}.$$

Now we let $\lambda = \phi$ and construct the optimal coordinates (λ, ξ, η) by making the transformation

$$(x, y, z) \rightarrow (\lambda, \xi, \eta),$$

where $\xi = \xi(x, y, z), \eta = \eta(x, y, z)$ are selected so that they are two independent families of stream functions whose existence are known a priori; thus

$$\nabla\xi \perp \boldsymbol{q}, \quad \nabla\eta \perp \boldsymbol{q}.$$

The transformation so constructed is

$$
\begin{aligned}
\mathrm{d}\lambda &= \lambda_x \mathrm{d}x + \lambda_y \mathrm{d}y + \lambda_z \mathrm{d}z = \nabla\lambda \cdot \mathrm{d}\boldsymbol{r}, \\
\mathrm{d}\xi &= \xi_x \mathrm{d}x + \xi_y \mathrm{d}y + \xi_z \mathrm{d}z = \nabla\xi \cdot \mathrm{d}\boldsymbol{r}, \\
\mathrm{d}\eta &= \eta_x \mathrm{d}x + \eta_y \mathrm{d}y + \eta_z \mathrm{d}z = \nabla\eta \cdot \mathrm{d}\boldsymbol{r}.
\end{aligned}
\tag{9.16}
$$

Now we need to show that the above transformation belongs to the class (9.1). For this purpose, we invert (9.16) to get

$$
\begin{pmatrix} \mathrm{d}x \\ \mathrm{d}y \\ \mathrm{d}z \end{pmatrix} = \frac{1}{\det\left[\dfrac{\partial(\lambda, \xi, \eta)}{\partial(x, y, z)}\right]}
\begin{pmatrix}
\begin{vmatrix} \xi_y & \xi_z \\ \eta_y & \eta_z \end{vmatrix} & -\begin{vmatrix} \lambda_y & \lambda_z \\ \eta_y & \eta_z \end{vmatrix} & \begin{vmatrix} \lambda_y & \lambda_z \\ \xi_y & \xi_z \end{vmatrix} \\[2mm]
-\begin{vmatrix} \xi_x & \xi_z \\ \eta_x & \eta_z \end{vmatrix} & \begin{vmatrix} \lambda_x & \lambda_z \\ \eta_x & \eta_z \end{vmatrix} & -\begin{vmatrix} \lambda_x & \lambda_z \\ \xi_x & \xi_z \end{vmatrix} \\[2mm]
\begin{vmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{vmatrix} & -\begin{vmatrix} \lambda_x & \lambda_y \\ \eta_x & \eta_y \end{vmatrix} & \begin{vmatrix} \lambda_x & \lambda_y \\ \xi_x & \xi_y \end{vmatrix}
\end{pmatrix}
\begin{pmatrix} \mathrm{d}\lambda \\ \mathrm{d}\xi \\ \mathrm{d}\eta \end{pmatrix}.
$$

$$\tag{9.17}$$

Since $\nabla\xi \perp \boldsymbol{q}, \nabla\eta \perp \boldsymbol{q}$, it follows that

$$\nabla\xi \times \nabla\eta /\!/ \boldsymbol{q};$$

that is,

$$\frac{\begin{vmatrix} \xi_y & \xi_z \\ \eta_y & \eta_z \end{vmatrix}}{u} = \frac{\begin{vmatrix} \xi_z & \xi_x \\ \eta_z & \eta_x \end{vmatrix}}{v} = \frac{\begin{vmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{vmatrix}}{w} \equiv \tilde{h}.$$

Meanwhile,

$$\nabla\lambda = (\lambda_x, \lambda_y, \lambda_z) = (ku, kv, kw) = k\boldsymbol{q};$$

hence

$$\left|\frac{\partial(\lambda, \xi, \eta)}{\partial(x, y, z)}\right| = \begin{vmatrix} \lambda_x & \lambda_y & \lambda_z \\ \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \end{vmatrix} = \lambda_x \tilde{h}u + \lambda_y \tilde{h}v + \lambda_z \tilde{h}w = k\tilde{h}q^2.$$

Thus, (9.17) becomes

$$\begin{pmatrix} dx \\ dy \\ dz \end{pmatrix} = \frac{1}{k\tilde{h}q^2} \begin{pmatrix} \tilde{h}u & -\begin{vmatrix} \lambda_y & \lambda_z \\ \eta_y & \eta_z \end{vmatrix} & \begin{vmatrix} \lambda_y & \lambda_z \\ \xi_y & \xi_z \end{vmatrix} \\ \tilde{h}v & \begin{vmatrix} \lambda_x & \lambda_z \\ \eta_x & \eta_z \end{vmatrix} & -\begin{vmatrix} \lambda_x & \lambda_z \\ \xi_x & \xi_z \end{vmatrix} \\ \tilde{h}w & -\begin{vmatrix} \lambda_x & \lambda_y \\ \eta_x & \eta_y \end{vmatrix} & \begin{vmatrix} \lambda_x & \lambda_y \\ \xi_x & \xi_y \end{vmatrix} \end{pmatrix} \begin{pmatrix} d\lambda \\ d\xi \\ d\eta \end{pmatrix}$$

$$= \begin{pmatrix} \dfrac{u}{kq^2} & -\dfrac{1}{\tilde{h}q^2}\begin{vmatrix} v & w \\ \eta_y & \eta_z \end{vmatrix} & \dfrac{1}{\tilde{h}q^2}\begin{vmatrix} v & w \\ \xi_y & \xi_z \end{vmatrix} \\ \dfrac{1}{kq^2} & \dfrac{1}{\tilde{h}q^2}\begin{vmatrix} u & w \\ \eta_x & \eta_z \end{vmatrix} & -\dfrac{1}{\tilde{h}q^2}\begin{vmatrix} u & w \\ \xi_x & \xi_z \end{vmatrix} \\ \dfrac{w}{kq^2} & -\dfrac{1}{\tilde{h}q^2}\begin{vmatrix} u & v \\ \eta_x & \eta_y \end{vmatrix} & \dfrac{1}{\tilde{h}q^2}\begin{vmatrix} u & v \\ \xi_x & \xi_y \end{vmatrix} \end{pmatrix} \begin{pmatrix} d\lambda \\ d\xi \\ d\eta \end{pmatrix}$$

$$= \begin{pmatrix} hu & A_1 & B_1 \\ hv & A_2 & B_2 \\ hw & A_3 & B_3 \end{pmatrix} \begin{pmatrix} d\lambda \\ d\xi \\ d\eta \end{pmatrix}. \tag{9.18}$$

This completes the proof. Moreover, we have

$$h = \frac{1}{kq^2} \quad \text{or} \quad k = \frac{1}{hq^2}.$$

Conversely, when transformation (9.18) is given and the flow condition (9.14) is satisfied, it can be shown similarly that

$$h = \frac{1}{kq^2}.$$

Although the optimal coordinate system does not exist for general 3-D steady flow, the UC λ-marching method still gives good results, e.g., good shear layer resolution. In the following, a 3-D steady supersonic inviscid corner flow computation is presented[9], where two intersecting wedges, both with angles of $9.5°$ form an axial corner over which there is a Mach 3 flow. The flow field consists of two planar wedge shocks, two embedded shocks, a corner shock and a shear layer. Figure 9.10 shows the geometrical configuration (left) and the computed u-velocity contours (right) on a typical λ-plane. The corner shock, embedded shocks, and 2-D wedge shocks are clearly shown. In particular, the triangular slip-surfaces are distinct and sharp. They all agree with the experimental locations by West and Korkegi[10].

(a) sketch of the problem   (b) u-velocity contours
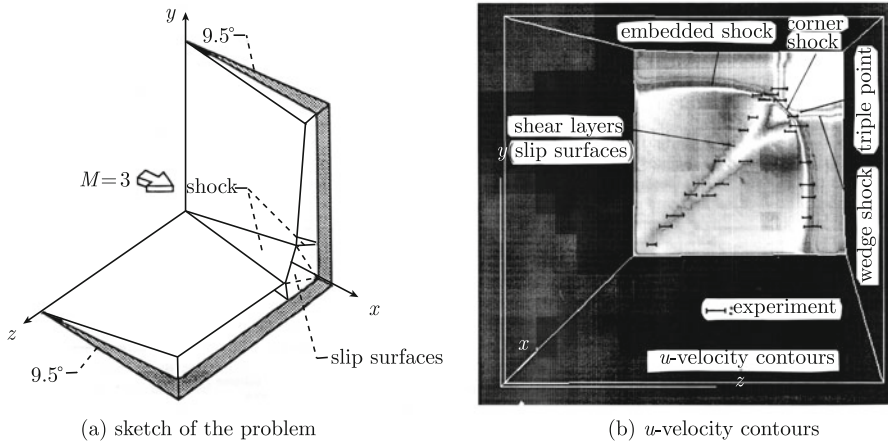
Figure 9.10   Supersonic flow past a corner[9]

Instead of using a shock capturing method, the random choice method has also been successfully applied in UC to numerically solve 2-D and 3-D supersonic steady flow problems[11, 12].

References

[1] W.H. HUI AND D.L. CHU. Optimum grid for the steady Euler equations. *Computational Fluid Dynamics Journal*, 4: 403-426, 1996.

[2] W.H. HUI AND C.Y. LOH. A new Lagrangian method for steady supersonic flow computation, Part III: Strong shocks. *J. Comput. Phys.*, 103: 465-471, 1992.

[3] C.Y. LEPAGE AND W.H. HUI. A shock-adaptive Godunov scheme based on the generalized Lagrangian formulation. *J. Comput. Phys.*, 122: 291-299, 1995.

[4] W.H. HUI AND Y. HE. Hyperbolicity and optimal coordinates of the three-dimensional steady Euler equations. *SIAM Journal on Applied Mathematics*, 57: 893-928, 1997.

[5] H.M. GLAZ AND A.B. WARDLAW. A high-order Godunov scheme for steady supersonic gas dynamics. *J. Comput. Phys.*, 58: 157-187, 1985.

[6] W.H. HUI AND J.J. HU. Space-marching gridless computation of steady supersonic/hypersonic flow. *Int. J. Comput. Fluid Dyn.*, 20: 55-59, 2006.

[7] J.W. MACCOLL. The conical shock wave formed by a cone moving at high speed. *Proc. R. Soc. Ser. A*, 159: 459-472, 1937.

[8] C. ROGERS, W.K. SCHIEF AND W.H. HUI. On complex-lamellar motion of a Prim gas. *J. Math Anal. Appl.*, 266: 55-69, 2002.

[9] C.Y. LOH AND M.S. LIOU. A new Lagrangian method for three-dimensional steady supersonic flows. *J. Comput. Phys.*, 113: 224-248, 1994.

[10] J.E. WEST AND R.H. KORKEGI. Supersonic interaction in the corner of intersecting wedges at high Reynolds numbers. *AIAA J.*, 10: 652, 1972.

[11] C.Y. LOH AND W.H. HUI. A new Lagrangian random choice method for solving the steady Euler equations. *Computational Fluid Dynamics Journal*, 2: 247-268, 1993.

[12] C.Y. LOH, M. S. LIOU AND W.H. HUI. An investigation of random choice method for 3-D steady supersonic flow. *International Journal for Numerical Methods in Fluids*, 29: 97-119, 1999.

Chapter 10

Unsteady 2-D and 3-D Flow Computation

10.1 Summary of Solution to the 2-D Euler Equations Using the Unified Coordinates

To illustrate the idea, we consider 2-D flow. Let us first summarize the results in Chapters 6 and 7. The unified coordinates (λ, ξ, η) are related to the Eulerian ones (t, x, y) via the transformation

$$
\begin{cases}
dt = d\lambda, \\
dx = U d\lambda + A d\xi + L d\eta, \\
dy = V d\lambda + B d\xi + M d\eta.
\end{cases}
\tag{10.1}
$$

From (10.1), we get

$$
\frac{D_{\boldsymbol{Q}}}{Dt} \begin{pmatrix} \xi \\ \eta \end{pmatrix} = 0,
\tag{10.2}
$$

where

$$
\frac{D_{\boldsymbol{Q}}}{Dt} \equiv \frac{\partial}{\partial t} + \boldsymbol{Q} \cdot \nabla_x.
$$

So the coordinates (ξ, η), and hence computational cells, move with velocity $\boldsymbol{Q} = (U, V)$.

Under transformation (10.1), the system of governing equations becomes

$$
\frac{\partial \boldsymbol{E}}{\partial \lambda} + \frac{\partial \boldsymbol{F}}{\partial \xi} + \frac{\partial \boldsymbol{G}}{\partial \eta} = \boldsymbol{0},
\tag{10.3}
$$

where

$$
\boldsymbol{E} = \begin{pmatrix} \rho\Delta \\ \rho\Delta u \\ \rho\Delta v \\ \rho\Delta e \\ A \\ B \\ L \\ M \end{pmatrix}, \quad
\boldsymbol{F} = \begin{pmatrix} \rho X \\ \rho X u + pM \\ \rho X v - pL \\ \rho X e + p(uM - vL) \\ -U \\ -V \\ 0 \\ 0 \end{pmatrix}, \quad
\boldsymbol{G} = \begin{pmatrix} \rho Y \\ \rho Y u - pB \\ \rho Y v + pA \\ \rho Y e + p(vA - uB) \\ 0 \\ 0 \\ -U \\ -V \end{pmatrix}
\tag{10.4}
$$

with $\Delta = AM - BL, X = (u - U)M - (v - V)L$ and $Y = (v - V)A - (u - U)B$. The first four equations in (10.3) are the physical conservation laws and the last four are the geometric conservation laws.

Since there are two arbitrary functions U and V, we prescribe two requirements:

(1) Coordinate lines $\eta =$const. shall be material lines of fluid particles, meaning

$$\frac{D_q \eta}{Dt} = 0. \tag{10.5}$$

Together with the second equation of (10.2), we get

$$(v - V)A = (u - U)B. \tag{10.6}$$

(2) Mesh angles, and hence mesh orthogonality, shall be preserved during the λ-marching computation. Thus we have

$$\frac{\partial}{\partial \lambda} \cos^{-1}\left(\frac{\nabla\xi \cdot \nabla\eta}{|\nabla\xi||\nabla\eta|}\right) = \frac{\partial}{\partial \lambda}\arccos\left(\frac{AL + BM}{\sqrt{A^2 + B^2}\sqrt{L^2 + M^2}}\right) = 0. \tag{10.7}$$

After expressing V in terms of U from (10.6) and using the geometric conservation laws, (10.7) becomes an ordinary differential equation for U,

$$\frac{\partial U}{\partial \eta} + P(\eta; \lambda, \xi)U = Q(\eta; \lambda, \xi), \tag{10.8}$$

where

$$P(\eta; \lambda, \xi) = \frac{S^2}{T^2\Delta}\left(A\frac{\partial B}{\partial \xi} - B\frac{\partial A}{\partial \xi}\right) - \frac{L}{A\Delta}\left(A\frac{\partial B}{\partial \eta} - B\frac{\partial A}{\partial \eta}\right),$$

$$Q(\eta; \lambda, \xi) = \frac{S^2 A}{T^2\Delta}\left(B\frac{\partial u}{\partial \xi} - A\frac{\partial v}{\partial \xi}\right) + \frac{L}{\Delta}\left(A\frac{\partial v}{\partial \eta} - B\frac{\partial u}{\partial \eta}\right) + uP(\eta; \lambda, \xi)$$

and

$$\Delta = AM - BL, \quad S^2 = L^2 + M^2, \quad T^2 = A^2 + B^2.$$

We can specify any initial data for U at $\eta = $ const..

We note that since (10.3) is in conservation PDE form, it can be solved as easily as Eulerian system in the λ-ξ-η space by marching in λ. The only difference is that at each time step we need compute the mesh velocity (U, V) by solving (10.8) for U and then (10.6) for V. After that we update the geometric quantities (A, B, L, M) to be used in the physical conservation laws. Detailed computation procedure follows.

10.2   Computation Procedure

We now show how to compute steady flow which may have subsonic regions. Flows of this type are computed by marching in time λ until a steady state is reached. An example of time-accurate flow computation is also given at the end of Section 10.3.

The computation procedure for uniform flow past a body is illustrated by a Mach 0.8 steady flow of air ($\gamma = 1.4$) past a NACA 0012 airfoil as follows[1]: **Initialization Stage—automatic generation of body-fitted mesh in a computational window.**

Given the mesh sizes Δx and Δy and the number of cells $M \times N$ in the window (we use $M = 200$ and $N = 100$ in the example).

Step 1   Begin with a column of N orthogonal cells, representing the given uniform flow in the x-direction (Figure 10.1(a)). This gives the initial vales of $(A, B, L, M) = (1, 0, 0, 1)$. We also take $(U, V) = (u, v)$ initially.

(a)   (b)   (c)   (d)
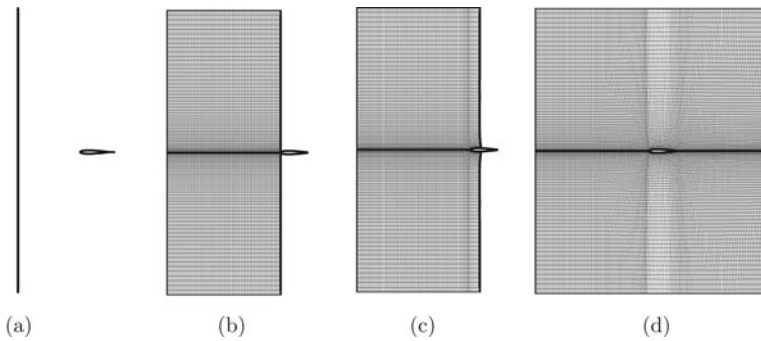
Figure 10.1   Flow generated mesh at different times

Step 2   Compute the solution to (10.3) by marching in time λ, using dimensional splitting: Splitting into two 1-D systems in λ-ξ and λ-η, each of them is solved using the standard Godunov/MUSCL scheme with the minmod limiter (Details are as follows: To update the solution from time n to time $n+1$: ① Solve the first four equations (the physical conservation laws) of (10.3) for (ρ, p, u, v) keeping A, B, L, M, U and V at time n level. ② Use this updated values together with a specified initial data to solve (10.8) for U and then (10.6) for V at time $n + 1$. ③ Use these 3 updated values of U and V to update (A, B, L, M) at time level

$n + 1$ by integrating the geometric conservation laws. At all outer boundaries of the computational region at every time step, we apply the characteristic boundary conditions[2]). After one time step $\Delta\lambda$, this column of cells moves to the right by $U\Delta\lambda$.

Step 3   After several time steps when the initial column of cells has moved to the right by a distance equal to Δx, add one new column of cells on the left that is identical to the initial column.

Step 4   Repeat this process of adding cell columns on the left of the computational region until the leading column meets the body surface (Figure 10.1(b)) we then impose the boundary condition of zero normal velocity on the body surface.

Step 5   Continue this process until after the columns of cells cover the whole body surface and further downstream, when we have M columns of cells in the window (Figure 10.1(d)). This completes the initialization stage, and we now have a body-fitted mesh (Figure 10.2) and a flow field around the airfoil in the window.
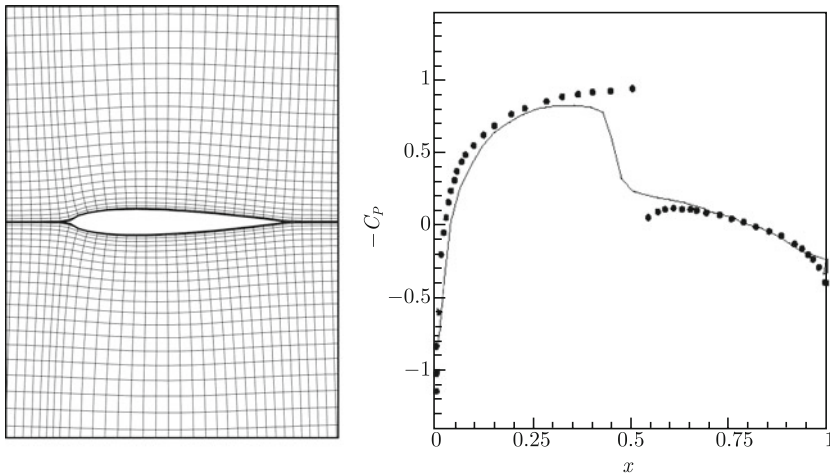
Figure 10.2   Preliminary mesh and surface pressure from UC computation (solid line) and reference results by Hafez et al.[3] (dots)

The computed mesh is seen orthogonal, as predicted. It is also fairly uniform in the x-direction, because in solving (10.8) we have specified uniform data for U at $\eta =$const.. The associated flow field computed (e.g., surface pressure in Figure 10.2) is, however, only a very rough approximation to the correct one (see, e.g., the potential solution of Hafez et al. [3]), partly because it has not reached the steady

state and partly because the downstream boundary conditions used at the transient times, e.g., in Figure 10.1 (c), are obviously incorrect as the computational regions at those times are not the full window.

To progress further, one could use the body-fitted orthogonal mesh generated so far to perform an Eulerian computation with the associated flow field as an initial solution. This can be easily done by putting $U = V = 0$ (without solving (10.8) and (10.6)) during the subsequent iterations towards a steady state. In this way, the unified coordinate approach plays the role of mesh generation for Eulerian computation.

An alternative and better way is to continue the unified coordinate computation to proceed to the **Main Stage—iteration with flow-adjusted meshes.**

Step 6   To iterate the solution towards a steady state, whenever we add a new column of cells on the left of the window we also simultaneously delete the rightmost column of cells from the computation window, thus keeping the window in the same size. At the same time, we improve the solution by using the information of the flow field at every time step, e.g., the surface pressure gradient, to adjust the initial data of U at $\eta =$const. in solving (10.8) so that the mesh is refined in regions of high pressure gradient. We note that this flow-adjusted refined mesh remains orthogonal. The computed mesh and surface pressure distribution are shown in Figure 10.3, which are much better than Figure 10.2 and also in good agreement with the potential flow computation of Hafez et al[3]. The pressure contours at the same time are shown in Figure 10.4.
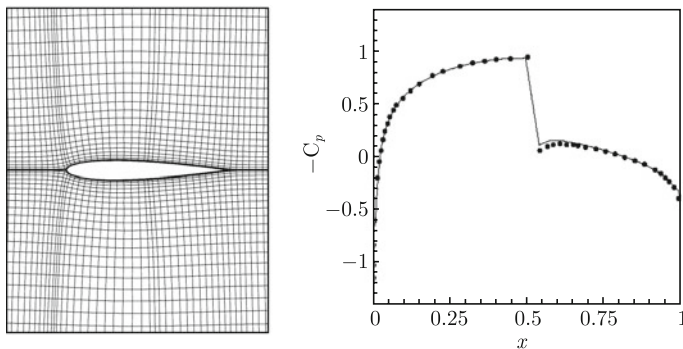
Figure 10.3   Self-justified mesh (left) and final pressure distributions (right) from UC computation (solid line) and reference results by Hafez et al.[3] (solid dots)
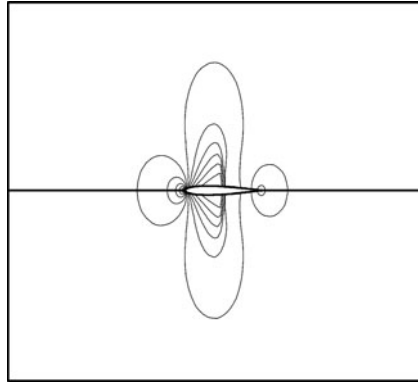
Figure 10.4   Pressure contours

The flow-adjusted mesh in Figure 10.3 looks similar to those obtained by the Mesh re-distribution method in Euelrian computation. But there are differences: Mesh re-distribution requires generating another mesh at every time step by solving an elliptic equation, and conservation properties in the interpolations of geometry and flow variables between the two meshes must be ensured. These issues do not arise in our flow-adjusted mesh approach because we need only one mesh; the only modification is to specify the initial data for U at η =const. by using the pressure gradient information known at each time step.

In terms of computing time, most of the CPU time is used in solving the Riemann problems for the physical conservation laws, which is the same as in Eulerian computation. However, additional times are needed to solve (10.8) and to update the geometric variables (A, B, L, M). These typically increase the CPU time by $5\% \sim 10\%$. (This can also be reduced if in Step 6 we put $U = V = 0$ after the flow-adjusted mesh is well established so that there is no need to solve (10.8) subsequently). On the other hand, mesh re-distribution in Eulerian computation by solving an elliptic equation increases CPU time. What is most important is that Eulerian computation always needs generating a body-fitted mesh prior to flow computation, and this can be time-consuming.

10.3   Examples

Example 1   Example 1 showing a transonic flow computation is already given above.

Example 2   Example 2[1] shows a sample computation for a Mach 2.2 steady flow over a NACA 0012 airfoil at an angle of attack of 8 degrees and a horizontal air-SF6 material. Figure 10.5 shows the flow-generated meshes at different times, whereas Figures 10.6 and 10.7 show the computed contours of density, pressure and entropy at those times. The interface between air and SF6 was initially identified with a particular value of η and it remained so, because a contact line coincides with a coordinate line η =const. in the unified coordinate system. The computation was straight forward, with no special treatment; yet the results are better than the corresponding Eulerian ones, in particular the interface and the slip line behind the airfoil are resolved sharper.

(a) t=0.128   (b) t=1.04   (c) t=4.0
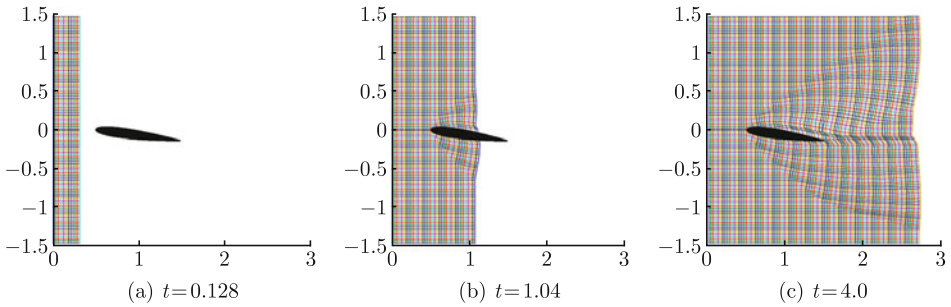
Figure 10.5   Flow-generated meshes

(a) t=0.128   (b) t=1.04   (c) t=4.0
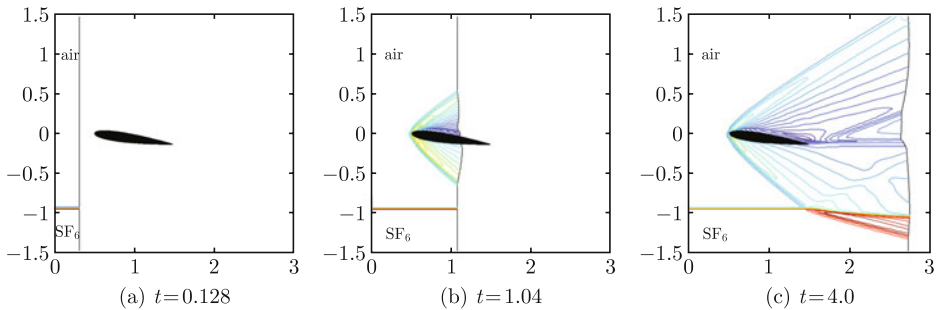
Figure 10.6   Density contours

Example 3   This is an unsteady supersonic flow, $M_\infty = 3.0$, past a diamond-shape airfoil with 10° vertex angle which is oscillating about its vertex according to $\theta = 2\sin 30\pi t$, where θ is the instantaneous pitching angle. The computed

flow-generated meshes[4] at different times are plotted in Figure 10.8. It is seen that at all times of the oscillation, the flow-generated meshes are body-fitted and are almost orthogonal, although the mesh on the expansion side of the airfoil is coarsened.

The UC approach also works well with the shallow water wave equations[5].

(a) pressure contours, $t=4.0$　　　(b) entropy contours, $t=4.0$
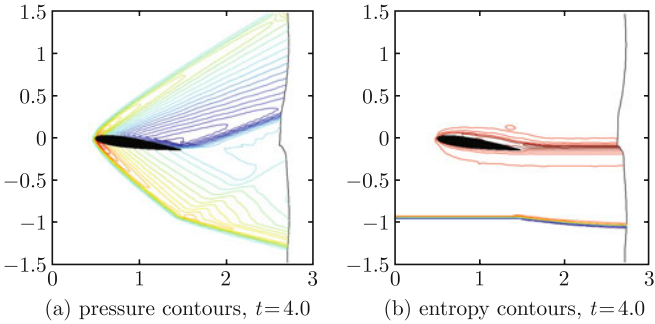
Figure 10.7　Pressure and entropy contours

(a) $t=0$　　　(b) $t=\frac{1}{10}T$　　　(c) $t=\frac{3}{10}T$

(d) $t=\frac{6}{10}T$　　　(e) $t=\frac{8}{10}T$　　　(f) $t=T$

(g) $t = 10\,T$ (h) $t = \left(10 + \frac{1}{4}\right)T$ (i) $t = \left(10 + \frac{2}{4}\right)T$

(j) $t = \left(10 + \frac{3}{4}\right)T$ (k) $t = 11\,T$
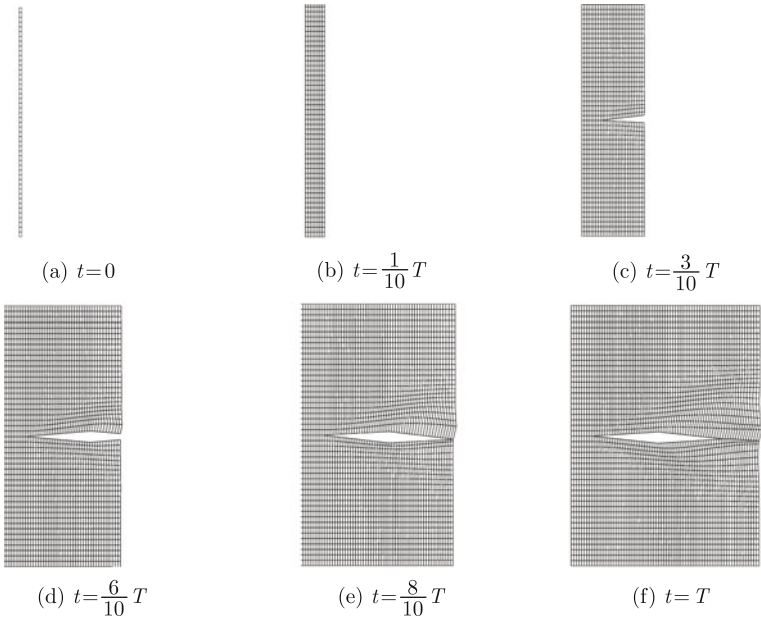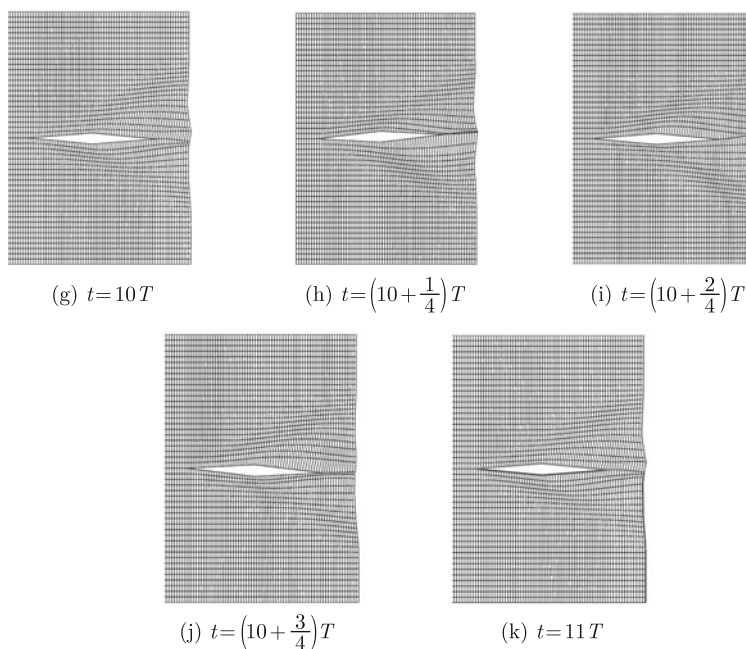
Figure 10.8   Flow-generated meshes for oscillating diamond-shape airfoil. Apex angle = 10, pitching motion about the apex: $\theta(t) = 2\sin 2\pi t/T$, the period of oscillation $T = 2/30$, free stream Mach number $M = 3.0$

References

[1] W.H. Hui, J.J. Hu and K.M. Shyue. Role of coordinates in computational fluid dynamics. *International Journal of Computational Fluid Dynamics*, 22: 3–9, 2008.

[2] C. Hirsch. *Numerical Computation of Internal and External Flows, Vol. II: Computational Methods for Inviscid and Viscous Flows.* New York: Wiley, 1990.

[3] M.M. Hafez, S. Osher and W. Whitlow. Improved finite different schemes for transonic potential calculations. *AIAA Paper 84-0092*, AIAA 22nd Aerospace Sciences Meeting, 1984.

[4] W.H. Hui, G.P. Zhao, J.J. Hu and Y. Zheng. Flow-generated-grids – gridless computation using the unified coordinates. in *Proceedings of the 9th International Conference on Numerical Grid Generation*, 111-121, 2005.

[5] W.H. Hui and S. Kudriakov. Computation of shallow water waves using the unified coordinates. *SIAM Journal on Scientific Computing*, 23: 1615-1654, 2002.

Chapter 11

Viscous Flow Computation Using Navier-Stokes Equations

11.1 Navier-Stokes Equations in the Unified Coordinates

In the precedent chapters, we have concentrated on inviscid flow. We now extend the unified coordinates method to viscous flow via the Navier-Stokes equations in this chapter, and via the BGK modeled Boltzmann equation in the next chapter.

The Navier-Stokes equations in 2-D Eulerian space can be expressed as

$$\frac{\partial \boldsymbol{E}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} + \frac{\partial \boldsymbol{G}}{\partial y} = \frac{\partial \boldsymbol{F}_v}{\partial x} + \frac{\partial \boldsymbol{G}_v}{\partial y}, \tag{11.1}$$

where

$$\boldsymbol{E} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho e \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (\rho e + p)u \end{pmatrix}, \quad \boldsymbol{F}_v = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - \kappa_x \end{pmatrix},$$

$$\boldsymbol{G} = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho e + p)v \end{pmatrix}, \quad \boldsymbol{G}_v = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} - \kappa_y \end{pmatrix}.$$

For a Newtonian fluid with Stokes hypothesis, the viscous stresses $\tau_{xx}, \tau_{xy}, \tau_{yy}$ and heat transfer rates κ_x, κ_y are

$$\tau_{xx} = 2\mu\frac{\partial u}{\partial x} - \frac{2}{3}\mu\nabla\cdot\boldsymbol{q}, \quad \tau_{yy} = 2\mu\frac{\partial v}{\partial y} - \frac{2}{3}\mu\nabla\cdot\boldsymbol{q}, \quad \tau_{xy} = \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \tag{11.2}$$

and

$$\kappa_x = -\frac{\mu c_p}{P_r}\frac{\partial T}{\partial x}, \quad \kappa_y = -\frac{\mu c_p}{P_r}\frac{\partial T}{\partial y}. \tag{11.3}$$

Here μ is viscosity coefficient, c_p is the specific heat at constant pressure, P_r is the Prandtl number and T is the temperature.

With the following transformation between the physical coordinates (t, x, y) and unified coordinates (λ, ξ, η),

$$\begin{cases} dt = d\lambda, \\ dx = U d\lambda + A d\xi + L d\eta, \\ dy = V d\lambda + B d\xi + M d\eta, \end{cases} \tag{11.4}$$

we can get the inverse transformation

$$\begin{pmatrix} d\lambda \\ d\xi \\ d\eta \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \xi_t & \xi_x & \xi_y \\ \eta_t & \eta_x & \eta_y \end{pmatrix} \begin{pmatrix} dt \\ dx \\ dy \end{pmatrix}. \tag{11.5}$$

The Navier-Stokes equations (11.1) in the unified coordinates (λ, ξ, η) become

$$\frac{\partial \bar{E}}{\partial \lambda} + \frac{\partial \bar{F}}{\partial \xi} + \frac{\partial \bar{G}}{\partial \eta} = \frac{\partial \bar{F}_v}{\partial \xi} + \frac{\partial \bar{G}_v}{\partial \eta}, \tag{11.6}$$

where

$$\begin{cases} \bar{E} = \Delta E, \\ \bar{F} = \Delta(\xi_t E + \xi_x F + \xi_y G), \\ \bar{G} = \Delta(\eta_t E + \eta_x F + \eta_y G), \\ \bar{F}_v = \Delta(\xi_x F_v + \xi_y G_v), \\ \bar{G}_v = \Delta(\eta_x F_v + \eta_y G_v) \end{cases} \tag{11.7}$$

and $\Delta = \partial(x, y)/\partial(\xi, \eta) = AM - BL$.

The coefficient matrix in (11.5) becomes

$$\begin{pmatrix} 1 & 0 & 0 \\ \xi_t & \xi_x & \xi_y \\ \eta_t & \eta_x & \eta_y \end{pmatrix} = \frac{1}{\Delta} \begin{pmatrix} \Delta & 0 & 0 \\ -X & M & -L \\ -Y & -B & A \end{pmatrix}, \tag{11.8}$$

where $X = MU - LV$ and $Y = AV - BU$. Hence, the variables in (11.6) are

$$\begin{cases} \bar{E} = \Delta E, \\ \bar{F} = (-X E + M F - L G), \\ \bar{G} = (-Y E - B F + A G), \\ \bar{F}_v = (M F_v - L G_v), \\ \bar{G}_v = (-B F_v + A G_v). \end{cases} \tag{11.9}$$

The viscous shear stresses given by (11.2) become the following in the transformed computational space:

$$
\begin{cases}
\tau_{xx} = \dfrac{\mu}{\Delta}\left[\dfrac{4}{3}(Mu_\xi - Bu_\eta) - \dfrac{2}{3}(-Lv_\xi + Av_\eta)\right], \\[3mm]
\tau_{yy} = \dfrac{\mu}{\Delta}\left[\dfrac{4}{3}(-Lv_\xi + Av_\eta) - \dfrac{2}{3}(Mu_\xi - Bu_\eta)\right], \\[3mm]
\tau_{xy} = \dfrac{\mu}{\Delta}(-Lu_\xi + Au_\eta + Mv_\xi - Bv_\eta),
\end{cases}
\tag{11.10}
$$

and the heat conduction terms in (11.3) are

$$
\begin{cases}
\kappa_x = -\dfrac{\mu c_p}{P_r \Delta}(MT_\xi - BT_\eta), \\[3mm]
\kappa_y = -\dfrac{\mu c_p}{P_r \Delta}(-LT_\xi + AT_\eta).
\end{cases}
\tag{11.11}
$$

In the computational space, the viscous flux vector $\bar{\boldsymbol{F}}_v$ and $\bar{\boldsymbol{G}}_v$ are

$$
\bar{\boldsymbol{F}}_v = \frac{\mu}{\Delta}
\begin{pmatrix}
0 \\[2mm]
\left(\dfrac{4}{3}M^2+L^2\right)u_\xi - \dfrac{1}{3}MLv_\xi - \left(\dfrac{4}{3}MB+AL\right)u_\eta + \left(-\dfrac{2}{3}AM+BL\right)v_\eta \\[3mm]
-\dfrac{1}{3}MLu_\xi + \left(M^2+\dfrac{4}{3}L^2\right)v_\xi + \left(MA-\dfrac{2}{3}BL\right)u_\eta - \left(MB+\dfrac{4}{3}AL\right)v_\eta \\[3mm]
\dfrac{1}{2}\left(\dfrac{4}{3}M^2+L^2\right)(u^2)_\xi + \dfrac{1}{2}\left(M^2+\dfrac{4}{3}L^2\right)(v^2)_\xi - \dfrac{1}{3}ML(uv)_\xi \\[3mm]
+\dfrac{c_p(M^2+L^2)}{P_r}T_\xi + \dfrac{1}{2}\left(-\dfrac{4}{3}MB-AL\right)(u^2)_\eta + \dfrac{1}{2}\left(-MB-\dfrac{4}{3}AL\right)(v^2)_\eta \\[3mm]
+\left(MA-\dfrac{2}{3}BL\right)vu_\eta + \left(BL-\dfrac{2}{3}MA\right)uv_\eta + \dfrac{c_p(-MB-AL)}{P_r}T_\eta
\end{pmatrix}
\tag{11.12}
$$

and

$$
\bar{\boldsymbol{G}}_v = \frac{\mu}{\Delta}
\begin{pmatrix}
0 \\[2mm]
\left(-\dfrac{4}{3}MB-AL\right)u_\xi + \left(MA-\dfrac{2}{3}BL\right)v_\xi + \left(\dfrac{4}{3}B^2+A^2\right)u_\eta - \dfrac{1}{3}BAv_\eta \\[3mm]
\left(BL-\dfrac{2}{3}MA\right)u_\xi - \left(MB+\dfrac{4}{3}AL\right)v_\xi - \dfrac{1}{3}ABu_\eta + \left(B^2+\dfrac{4}{3}A^2\right)v_\eta \\[3mm]
\dfrac{1}{2}\left(-\dfrac{4}{3}MB-AL\right)(u^2)_\xi - \dfrac{1}{2}\left(MB+\dfrac{4}{3}AL\right)(v^2)_\xi + \left(AM-\dfrac{2}{3}BL\right)uv_\xi \\[3mm]
+\left(BL-\dfrac{2}{3}AM\right)vu_\xi + \dfrac{c_p(-MB-AL)}{P_r}T_\xi + \dfrac{1}{2}\left(\dfrac{4}{3}B^2+A^2\right)(u^2)_\eta \\[3mm]
+\dfrac{1}{2}\left(B^2+\dfrac{4}{3}A^2\right)(v^2)_\eta - \dfrac{1}{3}AB(uv)_\eta + \dfrac{c_p(B^2+A^2)}{P_r}T_\eta
\end{pmatrix}.
\tag{11.13}
$$

In order to solve the Navier-Stokes equations under unified coordinate system, the standard operator-splitting method is usually used, where the inviscid part is solved based on the Riemann solution as presented in the previous chapter, and the viscous terms are discretized using central differences.

The advantage for the inviscid flow computation using the unified coordinate method is mainly due to the fact that there is clear material or slip line, which can be used as one of the coordinate, such as η =const.. However, for the viscous flows, there is no sharp slip line, and the shear layer can be considered as an infinite number of slip lines. The failure of the classical Lagrangian method for the viscous flow computation is due to the mesh velocity following the fluid velocity in the dissipative region, such as inside the viscous boundary layer, where boundary cells can be infinitely deformed. So for computing steady or unsteady viscous flow, it is necessary to decouple the mesh velocity from the fluid velocity. The strategy used earlier, such as mesh orthogonality, is a good choice for the viscous flows.

11.2   The Angle-preserving Equation

We now discuss how to use unified coordinates to compte 2-D steady flow via the Navier-Stokes equations. We shall use the time-marching approach until the flow field reaches steady state. It is known from Section 7.6 that at steady state the mesh velocity $\boldsymbol{Q}$ is proportional to the fluid velocity $\boldsymbol{q}$,

$$\begin{cases} U = hu, \\ V = hv. \end{cases} \tag{11.14}$$

To accelerate convergence to steady state, we shall use $\boldsymbol{Q} = h\boldsymbol{q}$ during the transition stage. Accordingly, transformation (11.4) becomes

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = hu\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = hv\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta. \end{cases} \tag{11.15}$$

(11.14) requires the mesh velocity $\boldsymbol{Q}$ to be proportional to the fluid velocity $\boldsymbol{q}$, and thus might cause difficulties for viscous flow at solid boundary. But this apparent difficulty can be overcomed as explained presently.

The angle-preserving h-equation derived by Hui et al.[1] is

$$S^2 J \frac{\partial h}{\partial \xi} + T^2 I \frac{\partial h}{\partial \eta} = \left[S^2 \left(B \frac{\partial u}{\partial \xi} - A \frac{\partial v}{\partial \xi} \right) - T^2 \left(M \frac{\partial u}{\partial \eta} - L \frac{\partial v}{\partial \eta} \right) \right] h, \tag{11.16}$$

where

$$S^2 = L^2 + M^2, \quad T^2 = A^2 + B^2.$$

In the following, we will assume that the coefficients (A, B, L, M) are smooth enough. If the solution is smooth, then with h computed by (11.16), we have that if the mesh is orthogonal at the flow entry, the entire mesh will become orthogonal after a sufficiently long time iteration, independently of the orthogonality of the initial mesh. Orthogonal mesh is known to possess many desirable properties over non-orthogonal grids, e.g., attaining higher accuracy than non-orthogonal meshes.

As for the inviscid flow case, (11.16) is to be solved at every time step after the flow variables $\boldsymbol{Q} = (\rho, p, u, v)^{\mathrm{T}}$ and the geometric variables $\boldsymbol{K} = (A, B, L, M)^{\mathrm{T}}$ are found. We recall that the h-equation is a first order linear partial differential equations for $h(\xi, \eta; \lambda)$ with λ appearing as a parameter. To find solution h in the range

$$0 \leqslant h \leqslant 1, \tag{11.17}$$

we note that (11.16) is linear and homogeneous, therefore it possesses two properties: ① positive solution $h > 0$ always exists; ② if h is a solution to (11.16) so is h/C, C being any constant.

In order to ensure a solution satisfying (11.17), Hui et al[1] defined $g = \ln(hq)$ to replace the h equation by

$$S^2(A\sin\theta - B\cos\theta)\frac{\partial g}{\partial\xi} + T^2(M\cos\theta - L\sin\theta)\frac{\partial g}{\partial\eta}$$
$$= S^2\left(B\frac{\partial\cos\theta}{\partial\xi} - A\frac{\partial\sin\theta}{\partial\xi}\right) - T^2\left(M\frac{\partial\cos\theta}{\partial\eta} - L\frac{\partial\sin\theta}{\partial\eta}\right), \tag{11.18}$$

where $q = \sqrt{u^2 + v^2}$ and θ is the flow angle: $u = q\cos\theta, v = q\sin\theta$. Now, if g_1 is any solution to (11.18) then $h = \exp(g_1)/qC$ is a solution to (11.16) satisfying condition (11.17), provided we choose C equal to the maximum of $\exp(g_1)/q$ over the whole flow field being computed.

The advantages of replacing (11.16) by (11.18) become more pronounced for the Navier-Stokes equations. These, along with some further precautions, are explained in the next section.

11.3   Advantages of the **g**-equation Over the **h**-equation

There are three advantages of the g-equation (11.18) over the h-equation (11.16):

(1) g behaves more smooth than h. Indeed, as shown in [2] for supersonic flow, g is continuous across a slip line, but q and h are not. As such, accurate solution in g can be obtained easily from (11.18) but it is difficult to get accurate solution in h from (11.16).

(2) The mesh moves with the velocity hq. When we have a stagnation point such as the center of a vortex, a finite speed of the mesh, i.e., $hq > 0$, implies that we must have $h \to \infty$ as $q \to 0$. If we force h to be finite as given by solving the h-equation, the mesh cannot move at the stagnation point so that severe mesh deformation may still occur. If we solve the g-equation, we can ensure continuity of g, and hence of hq, everywhere.

(3) The greatest advantage in the context of viscous flow computation is the simplicity of specifying boundary condition at a nonslip wall. In order to preserve the mesh angle at the wall, say a line η_0, one should not define the boundary condition in the following way:

$$\frac{\partial(hq)}{\partial\eta} = 0 \quad \text{at} \quad \eta = \eta_0. \tag{11.19}$$

The above equation can be rewritten as

$$h\frac{\partial(q)}{\partial\eta} + q\frac{\partial h}{\partial\eta} = 0 \quad \text{at} \quad \eta = \eta_0.$$

However, on the wall $q = 0$, so that we have the following singular boundary condition which is very difficult to be realized in numerical computation:

$$\frac{\partial h}{\partial\eta} \to \infty. \tag{11.20}$$

Since we solve the g-equation, then, by (11.19), the boundary condition becomes simply

$$\frac{\partial g}{\partial\eta} = 0 \quad \text{at} \quad \eta = \eta_0. \tag{11.21}$$

Summing up, while the fluid velocity vector $\boldsymbol{q} = \boldsymbol{0}$ holds on a solid boundary, solving the g-equation with boundary condition (11.21) can give a finite value of hq on the solid boundary, allowing computational cells (the pseudo particles) to glide over the solid surface. This could not be achieved if the h-equation were solved, because the requisite boundary condition (11.20) is difficult to impose.

Some further precautions should be made in order to make the computation successful:

(1) the solution hq should be kept in memory since it is with hq that the mesh moves. If we just keep h, then at a stagnation point one would have $h \to \infty$, which is very difficult to handle numerically.

(2) when solving (11.18) up to a boundary, one might readily compute the flow angle by using

$$\cos\theta = \frac{u}{\max(\sqrt{u^2+v^2},\epsilon)}, \quad \sin\theta = \frac{v}{\max(\sqrt{u^2+v^2},\epsilon)},$$

where ϵ is a small parameter used to avoiding division by zero near a stagnation point. However, this is incorrect and would yield a wrong solution since in such a way one would have $\cos^2\theta + \sin^2\theta = 0$ at a stagnation point. At a stagnation point, one could extrapolate the flow angle from nearby points. Here we just solve (11.18) in the interior domain with g obtained through boundary condition (11.21).

11.4   Boundary Condition and Movement of Boundary Cells

The following details are programming techniques, which are nevertheless important for successful computation of viscous flows.

In the computations performed for inviscid flow past a solid body, a cell interface is taken to coincide with the solid wall and a ghost cell to lie inside the solid wall so that the solid wall lies at the middle between the ghost cell and the boundary cell. For a nonslip wall here, it is more convenient to place the center of the boundary cell on the wall so that the fluid velocity vanishes there exactly. As explained in the last section, solution to the g-equation allows the boundary cells to glide over the wall with velocity $h\boldsymbol{q}$. In some cases, the wall may involve discontinuities of slop. These must be taken into account when the boundary cells glide along the wall. A convenient way is to save the initial boundary points for all the time. Then the new boundary points are interpolated, by cubic interpolation, from the fixed initial boundary points under the constraint of orthogonality. If the initial boundary points were not saved, and one simply interpolates the boundary points of the new time step from those of the old time step, then slop discontinuities would propagate irregularly so that the shape of the wall would deform, sometimes severely. These remarks also apply to inviscid flow computations as in Example 2 of Chapter 9.

11.5   Solution Strategies

As the system of Navier-Stokes equations (11.6) is in conservation form, any well-established shock-capturing method can be used to solve the inviscid part. Similar to the approaches in the earlier chapters, we shall use the Godunov method with the MUSCL update to higher resolution to solve system (11.6). Details are shown in the last chapter. The viscous part is discretized by using central differences as follows.

Applying the divergence theorem to (11.6) over the cuboid cell (i, j, k) results in

$$\boldsymbol{E}_{i,j}^{k+1} = \boldsymbol{E}_{i,j}^{k} - \frac{\Delta\lambda^k}{\Delta\xi_i}\left(\overline{\boldsymbol{F}}_{i+1/2,j}^{k+1/2} - \overline{\boldsymbol{F}}_{i-1/2,j}^{k+1/2}\right) - \frac{\Delta\lambda^k}{\Delta\eta_j}\left(\overline{\boldsymbol{G}}_{i,j+1/2}^{k+1/2} - \overline{\boldsymbol{G}}_{i,j-1/2}^{k+1/2}\right),$$
$$i = 1, 2, \cdots, m; \ j = 1, 2, \cdots, n, \ (11.22)$$

where

$$\overline{\boldsymbol{F}}_{i+1/2,j}^{k+1/2} = \boldsymbol{F}_{i+1/2,j}^{k+1/2} + \underline{\boldsymbol{F}}_{i+1/2,j}^{k},$$
$$\overline{\boldsymbol{G}}_{i,j+1/2}^{k+1/2} = \boldsymbol{G}_{i,j+1/2}^{k+1/2} + \underline{\boldsymbol{G}}_{i,j+1/2}^{k},$$

$\boldsymbol{F}_{i+1/2,j}^{k+1/2}$ and $\boldsymbol{G}_{i,j+1/2}^{k+1/2}$ are the inviscid fluxes, $\underline{\boldsymbol{F}}_{i+1/2,j}^{k}, \underline{\boldsymbol{G}}_{i,j+1/2}^{k}$ are viscous contributions defined by second order central differences.

The numerical procedure can now be summarized as follows:

Step 1   Assume the initial conditions of a flow problem are given at $t = 0(\lambda = 0)$ in the xy plane. Then an appropriate ξ-η coordinate mesh is laid on the xy plane. A nonuniform mesh on the xy plane corresponds to a uniform mesh on the $\xi\eta$ plane, with constant $\Delta\xi_i$ and $\Delta\eta_j$, i.e., $\Delta\xi_i = \Delta\xi$ and $\Delta\eta_j = \Delta\eta$ are independent of i and j. One would wonder whether the results depend on the choice of the intervals $\Delta\xi$ and $\Delta\eta$, especially on the relative value $\Delta\xi/\Delta\eta$. The solution is independent of the choice of $\Delta\xi$ and $\Delta\eta$. However, from a numerical point of view, rounding errors may make the solution slightly different for different choices of $\Delta\xi$ and $\Delta\eta$. Most importantly, when using an iterative procedure to solve the equation for g, the convergence criterion (residue) is dependent on $\Delta\xi$ and $\Delta\eta$. Since the g-equation (11.18), when literally discretized, contains the coefficient $\dfrac{1}{\Delta\xi^2\Delta\eta^2}$, it is useful to let the convergence criterion be defined as

$$\varepsilon = \frac{10^{-14}}{\Delta\xi^2\Delta\eta^2}.$$

The initial and boundary conditions are defined according to the problem specified. The initial conditions for A, B, L and M can be computed simply as

$$A_{i,j} = \frac{x_{i+1,j} - x_{i-1,j}}{2\Delta\xi}, \quad B_{i,j} = \frac{y_{i+1,j} - y_{i-1,j}}{2\Delta\xi},$$

$$L_{i,j} = \frac{x_{i,j+1} - x_{i,j-1}}{2\Delta\eta}, \quad M_{i,j} = \frac{y_{i,j+1} - y_{i,j-1}}{2\Delta\eta}.$$

Step 2   We first describe the general strategy and then the specific steps.

General strategy: Strang splitting. Strang splitting is used to render the 2-D problem a sequential solutions of two 1-D problems. Let $\mathcal{L}_{\Delta\lambda}^{\xi}$ be the difference operator for the 1-D equation (simply obtained from (11.6) by dropping the $\frac{\partial}{\partial\eta}$ terms) whose solution after one time step $\Delta\lambda$

$$\widehat{\boldsymbol{E}}_{i,j}^{k+1} = \boldsymbol{E}_{i,j}^{k} - \frac{\Delta\lambda^{k}}{\Delta\xi_{i}}\left(\overline{\boldsymbol{F}}_{i+1/2,j}^{k+1/2} - \overline{\boldsymbol{F}}_{i-1/2,j}^{k+1/2}\right)$$

is denoted

$$\widehat{\boldsymbol{E}}^{k+1} = \mathcal{L}_{\Delta\lambda}^{\xi}\boldsymbol{E}^{k}.$$

Using $\widehat{\boldsymbol{E}}^{k+1}$ as initial solution and compute the final solution by

$$\boldsymbol{E}^{k+1} = \mathcal{L}_{\Delta\lambda}^{\eta}\widehat{\boldsymbol{E}}^{k+1} = \mathcal{L}_{\Delta\lambda}^{\eta}\mathcal{L}_{\Delta\lambda}^{\xi}\boldsymbol{E}^{k}, \tag{11.23}$$

where $\mathcal{L}_{\Delta\lambda}^{\eta}$ is the operator for the 1-D equation (simply obtained from (11.6) by dropping the $\frac{\partial}{\partial\xi}$ terms)

$$\boldsymbol{E}_{i,j}^{k+1} = \widehat{\boldsymbol{E}}^{k+1} - \frac{\Delta\lambda^{k}}{\Delta\eta_{j}}\left(\overline{\boldsymbol{G}}_{i,j+1/2}^{k+1/2}\left(\widehat{\boldsymbol{E}}^{k+1}\right) - \overline{\boldsymbol{G}}_{i,j-1/2}^{k+1/2}\left(\widehat{\boldsymbol{E}}^{k+1}\right)\right).$$

The simple splitting method (11.21) is not very accurate. The Strang splitting[3], which is more accurate than (11.23), is defined as

$$\boldsymbol{E}^{k+1} = \mathcal{L}_{\frac{\Delta\lambda}{2}}^{\xi}\mathcal{L}_{\Delta\lambda}^{\eta}\mathcal{L}_{\frac{\Delta\lambda}{2}}^{\xi}\boldsymbol{E}^{k} \tag{11.24}$$

for marching from λ^{k} to $\lambda^{k+1} = \lambda^{k} + \Delta\lambda, k = 0, 1, 2, \cdots$.

General strategy: time step-wise approach. The geometric conservation laws are decoupled from the physical conservation laws. Roughly speaking, at each time step, all the geometrical variables A, B, L, M, h are regarded as constants and we solve the equations in (11.6). After obtaining the physical variables at the new time step, we obtain A, B, L, M. Then we solve the equation for g.

Now let us state the specific steps. Note that we have used an explicit time integration method.

We first take A, B, L, M, h to be constant over the interval $\lambda^k < \lambda < \lambda^{k+1}$. Then for every pair of adjacent cells (i, j) and $(i + 1, j)$,

(0) Compute the time step $\Delta\lambda$. This is defined by using the stability condition

$$\frac{\Delta\lambda}{2}\left(\frac{\varrho_\xi}{\Delta\xi} + \frac{\varrho_\eta}{\Delta\eta}\right) < 1.$$

Here ϱ_ξ is the spectral radius of $\dfrac{\mathrm{d}\boldsymbol{F}}{\mathrm{d}\boldsymbol{E}}$ and ϱ_η that of $\dfrac{\mathrm{d}\boldsymbol{G}}{\mathrm{d}\boldsymbol{E}}$. We have

$$\varrho_\xi = \max_{\forall(i,j)}\left\{(1-h)\frac{|uM - vL|}{\Delta} + a\frac{\sqrt{|uM|^2 + |vL|^2}}{\Delta}\right\}$$

$$= \max_{\forall(i,j)}\left\{(1-h)|u\xi_x + v\xi_y| + a\sqrt{\xi_x^2 + \xi_y^2}\right\},$$

$$\varrho_\eta = \max_{\forall(i,j)}\left\{(1-h)\frac{|uB - vA|}{\Delta} + a\frac{\sqrt{|uB|^2 + |vA|^2}}{\Delta}\right\}$$

$$= \max_{\forall(i,j)}\left\{(1-h)|u\eta_x + v\eta_y| + a\sqrt{\eta_x^2 + \eta_y^2}\right\}.$$

where $\Delta = AM - BL$.

We also need a viscous update, which requires the time step to satisfy

$$\frac{\Delta\lambda}{Re}\frac{\Delta}{2}\left(\frac{\xi_x}{\Delta\xi^2} + \frac{\eta_y}{\Delta\eta^2}\right) < 1.$$

(1) Do a MUSCL type data reconstruction in a component by component manner. For example, in the ξ-direction, let f be any of the physical variables ρ, p, u, v, then, instead of assuming a uniform state in the cells (i, j) and $(i + 1, j)$, we assume linearly distributed states and use linear extrapolation to determine cell interface flow variables: $f_R = f_{i+1,j} - 0.5(f_{i+2,j} - f_{i+1,j})\phi(r^+)$ with $r^+ = (f_{i+1,j} - f_{i,j})/(f_{i+2,j} - f_{i+1,j})$ and $f_L = f_{i,j} + 0.5(f_{i,j} - f_{i-1,j})\phi(r^-)$ with $r^- = (f_{i+1,j} - f_{i,j})/(f_{i,j} - f_{i-1,j})$, where $\phi(r) = \max\{0, \min(1, r)\}$ is the minmod flux limiter and subscripts R and L of f correspond to right and left states, respectively.

(2) Define the normal direction of the cell interface $\xi_{i+\frac{1}{2},j}$ between the two adjacent cells (i, j) and $(i + 1, j)$ as

$$\boldsymbol{n} = \frac{(\nabla\xi)_{i,j} + (\nabla\xi)_{i+1,j}}{|(\nabla\xi)_{i,j} + (\nabla\xi)_{i+1,j}|}.$$

Project the velocity vector $\boldsymbol{q} = (u, v)$ into the normal and the tangential components (ω and τ).

(3) Solve the Riemann problem to get the interfacial flow variables $(\rho, p, \omega, \tau)^{\mathrm{T}}$ and hence $(\rho, p, u, v)^{\mathrm{T}}$ at $\xi = \xi_{i+\frac{1}{2},j}$. These are constants and will be denoted by $(\cdot)_{i+\frac{1}{2},j}$.See [1] for details.

(4) Update the geometrical variables as follows:

$$
\begin{pmatrix} A_{i,j}^{k+1} \\ B_{i,j}^{k+1} \end{pmatrix} = \begin{pmatrix} A_{i,j}^{k} \\ B_{i,j}^{k} \end{pmatrix} + \frac{\Delta\lambda^k}{\Delta\xi_i} h_{i,j}^k \begin{pmatrix} u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j} \\ v_{i+\frac{1}{2},j} - v_{i-\frac{1}{2},j} \end{pmatrix},
$$

$$
\begin{pmatrix} L_{i,j}^{k+1} \\ M_{i,j}^{k+1} \end{pmatrix} = \begin{pmatrix} L_{i,j}^{k} \\ M_{i,j}^{k} \end{pmatrix}
$$

or compute them directly by using

$$
A_{i,j}^{k+1} = (x_\xi)_{i,j}^{k+1} = \frac{x_{i+1,j}^{k+1} - x_{i-1,j}^{k+1}}{2\Delta\xi_i}, \quad B_{i,j}^{k+1} = (y_\xi)_{i,j}^{k+1} = \frac{y_{i+1,j}^{k+1} - y_{i-1,j}^{k+1}}{2\Delta\xi_i},
$$

$$
L_{i,j}^{k+1} = (x_\eta)_{i,j}^{k+1} = \frac{x_{i,j+1}^{k+1} - x_{i,j-1}^{k+1}}{2\Delta\eta_j}, \quad M_{i,j}^{k+1} = (y_\eta)_{i,j}^{k+1} = \frac{y_{i,j+1}^{k+1} - y_{i,j-1}^{k+1}}{2\Delta\eta_j}.
$$

(5) Calculate the cell interface fluxes. For instance, the 2nd component of the interface flux $\boldsymbol{F}_{i+\frac{1}{2},j}^{k+\frac{1}{2}}$ is evaluated as

$$
\rho_{i+\frac{1}{2},j}(1 - h_{i,j}^k)(u_{i+\frac{1}{2},j} M_{i,j}^{k+1} - v_{i+\frac{1}{2},j} L_{i,j}^{k+1}) + p_{i+\frac{1}{2},j} M_{i,j}^{k+1}
$$

plus the viscous flux.

(6) Update the conserved variables $\boldsymbol{E}_p$ in the physical conservation laws (11.6) using

$$
\boldsymbol{E}_{p_{i,j}}^{k+1} = \boldsymbol{E}_{p_{i,j}}^{k} - \frac{\Delta\lambda^k}{\Delta\xi_i}\left(\boldsymbol{F}_{i+\frac{1}{2},j}^{k+\frac{1}{2}} - \boldsymbol{F}_{i-\frac{1}{2},j}^{k+\frac{1}{2}}\right).
$$

(7) Decode $\boldsymbol{E}_{p_{i,j}}^{k+1}$ to get $\boldsymbol{Q}_{i,j}^{k+1}$, (physical variables), using $\Delta = A_{i,j}^{k+1} M_{i,j}^{k+1} - B_{i,j}^{k+1} L_{i,j}^{k+1}$.

(8) Apply Strang splitting, (11.24), to advance $\boldsymbol{E}_{i,j}^k$ to $\boldsymbol{E}_{i,j}^{k+1}$ (or $\boldsymbol{Q}_{i,j}^k$ to $\boldsymbol{Q}_{i,j}^{k+1}$).

(9) Update $h_{i,j}^k$ to $h_{i,j}^{k+1}$ by solving (11.18), using the updated values $\boldsymbol{Q}_{i,j}^{k+1}$ and $\boldsymbol{K}_{i,j}^{k+1}$ (geometrical variables) in its coefficients (Note: this step (9) is, of course, to be by-passed if $h = $ const. is assumed in the computation).

(10) (optional) Calculate the mesh in the x-y plane at λ^{k+1}:

$$
\begin{cases}
x_{i,j}^{k+1} = x_{i,j}^k + \dfrac{1}{2}\left((hq\cos\theta)_{i,j}^{\,k} + (hq\cos\theta)_{i,j}^{\,k+1}\right)\Delta\lambda, \\[2mm]
y_{i,j}^{k+1} = y_{i,j}^k + \dfrac{1}{2}\left((hq\sin\theta)_{i,j}^{\,k} + (hq\sin\theta)_{i,j}^{\,k+1}\right)\Delta\lambda.
\end{cases}
$$

By a mesh we mean the lines joining the cell centers, not the cell interface lines.

After this, we repeat Step 2 to advance the solution further to λ^{k+2}, and so on.

Step 3   The motionless viewing window technique is applied once the mesh has moved to the right. Accordingly, the column of cells which have moved out of the original physical domain to the right are deleted, while a new column of cells is added at the input flow boundary on the left.

In summary, our numerical scheme is second order accurate in space but only first order accurate in time.

11.6   Test Examples: Shock/Boundary Flow Interaction and Shock/Shock Interaction

In this section, the unified coordinates approach is tested numerically on two examples with comparison to the Eulerian approach. One is the shock/boundary flow interaction with separation. The second one is a shock/shock interaction problem.

The objective of these computations is to show the UC method can handle complex flows while keeping its advantages in resolving discontinuities in the inviscid flow regime.

As the mesh is adapted at every time step, the UC approach takes more time than the classical Eulerian approach mainly through the solution of the g-equation, for which we have simply used an explicit iterative method. It makes the UC approach to have 50% more CPU time than the Eulerian approach. We believe a more efficient algorithm to solve the g-equation can substantially reduce the CPU time.

Example 1   Shock/boundary flow interaction[4]. Shock/boundary layer interaction is an important and complex problem in supersonic and transonic flow applications. Because of its importance, it has been actively investigated since the beginning of high-speed aerodynamics in the 1940s. A comprehensive account on this subject is given in [5], where most of the historical papers and recent publications are cited. There are essentially three types of shock/boundary layer interaction. Here we only consider the interaction between an oblique incident shock wave

and a boundary layer. The oblique shock wave is supposed to be strong enough to induce separation. Our purpose here is to first demonstrate that even with such a flow (with strong recirculation) the unified coordinate system works. Then we perform some comparison with known experimental study.

The problem is defined as follows. The freestream condition is $M_\infty = 3$, $Re = 10^4$. The incident shock angle is $\beta = 30°$ which corresponds to a deflection angle $\theta = 12.8°$. The flow conditions (ρ, u, v, p, M) after the oblique shock are given by the oblique shock relations

$$\frac{\rho}{\rho_\infty} = \frac{(\gamma+1)M_\infty^2 \sin^2\beta}{2+(\gamma-1)M_\infty^2 \sin^2\beta},$$

$$\frac{p}{p_\infty} = 1 + \frac{2\gamma}{\gamma+1}(M_\infty^2 \sin^2\beta - 1),$$

$$M^2 = \frac{M_\infty^2 + \dfrac{2}{\gamma-1}}{\dfrac{2\gamma}{\gamma-1}M_\infty^2 \sin^2\beta - 1} + \frac{M_\infty^2 \cos^2\beta}{\dfrac{\gamma-1}{2}M_\infty^2 \sin^2\beta + 1},$$

$$u = M\sqrt{\frac{\gamma p}{\rho}}\cos\theta,$$

$$v = M\sqrt{\frac{\gamma p}{\rho}}\sin\theta,$$

where the deflection angle θ is related to the shock angle β by

$$\tan\theta = 2\arctan\frac{M_\infty^2 \sin^2\beta - 1}{M_\infty^2(\gamma+\cos 2\beta)+2}.$$
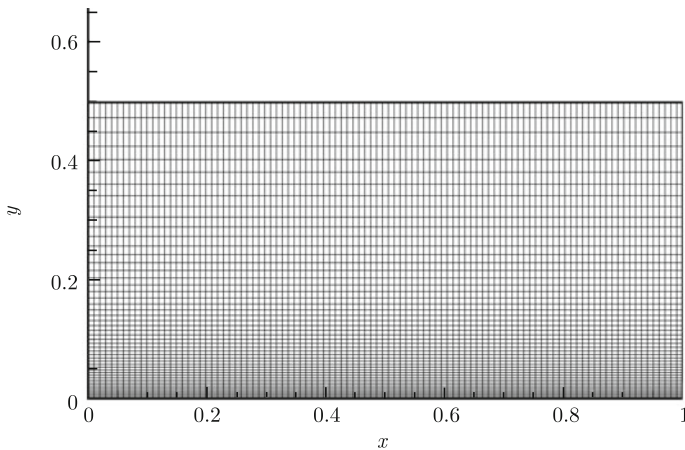
Figure 11.1   Shock/boundary layer interaction. Mesh used for the Eulerian approach.

$$\Delta y_{\min} = 0.002$$

In the computation using the Eulerian system, a mesh of 100×50 is employed. Initially, a mesh with $\Delta x = 1/100, \Delta y_{i,j=1} = 0.002$ and $\Delta y_{i,j+1}/\Delta y_{i,j} = 1.05$ ($\forall$ $j \in (1, 49)$) in the physical plane is laid over a domain of $\{0 \leqslant x \leqslant 1, 0 \leqslant y \leqslant 0.5\}$. The mesh is shown in Figure 11.1. It is also used as the initial mesh for the unified coordinates computation.

The Mach contours and the streamlines are shown in Figures 11.2 and 11.3. It is clear that the interaction induces separation and a vortex is formed.
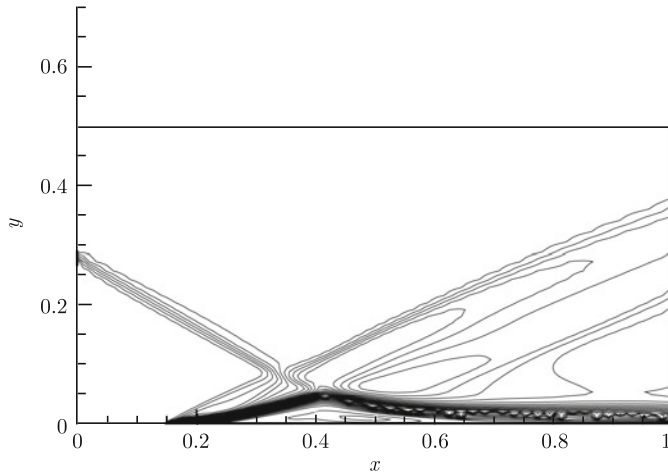
Figure 11.2   Mach contours obtained using the Eulerian approach
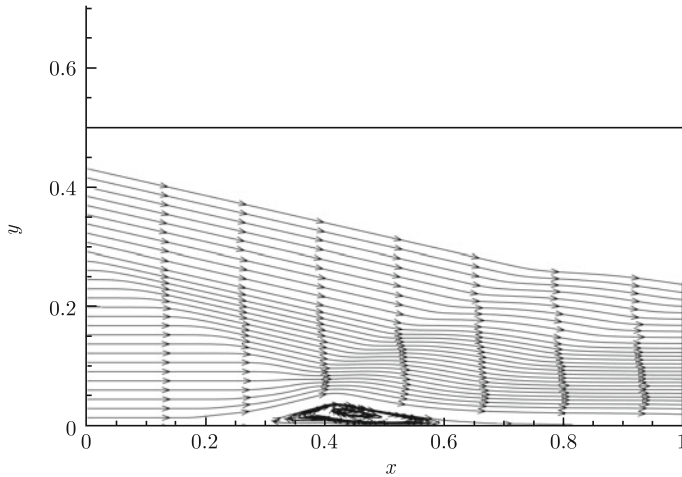
Figure 11.3   Streamlines by the Eulerian approach

Now consider the unified coordinates approach. The computation uses the Eulerian mesh given above as the initial mesh. The final mesh produced by the unified coordinates approach is displayed in Figure 11.4.
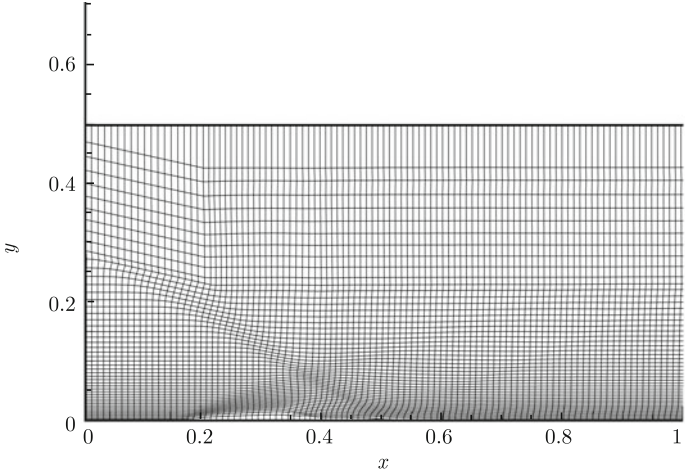
Figure 11.4   Shock/boundary layer interaction. Mesh produced by the UC approach

The Mach contours and the streamlines are shown in Figures 11.5 and 11.6, respectively. It is encouraging that the unified coordinates approach can handle the case of shock/boundary layer interaction even though a vortex exists. Such a case had been impossible using the classical Lagrangian approach.
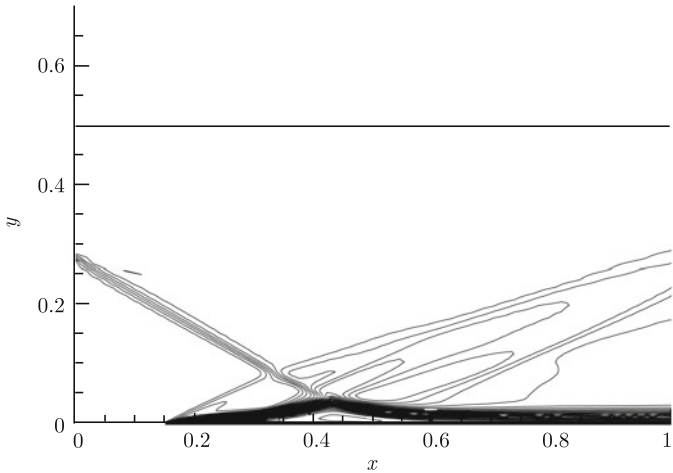
Figure 11.5   Mach contours obtained using the UC approach
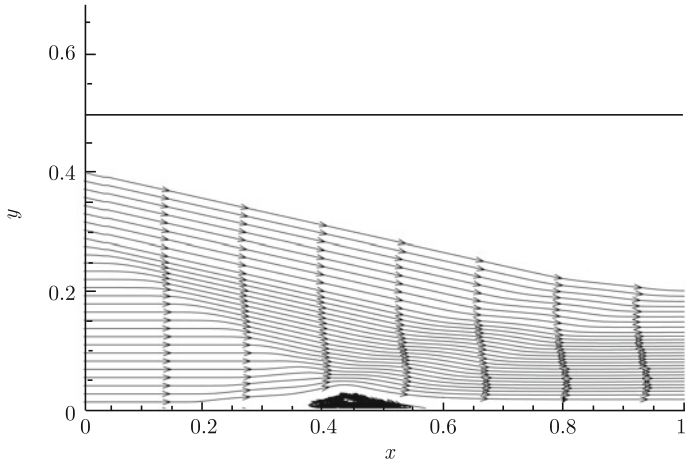
Figure 11.6   Streamlines by the UC approach

Example 2   Shock/Shock interaction. Now consider the problem of shock/shock interaction. Two oblique shocks are generated by two wedges which cause a deflection of the flow by an angle $\theta = 5°$ (lower wedge) and $\theta = 15°$ (upper). The upstream Mach number is $M_\infty = 4$. Figures 11.7 and 11.8 show Mach contours of inviscid computation using Eulerian and UC approach. After the shock/shock interaction, the emerged slipline is better resolved in the UC computation in comparison with the Eulerian one.
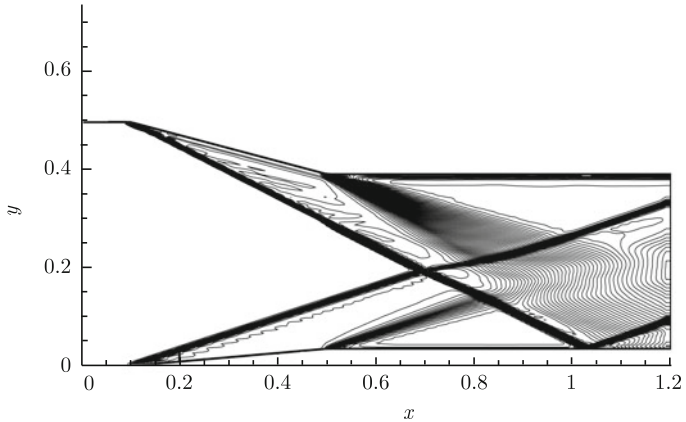
Figure 11.7   Eulerian computation for shock/shock interaction (inviscid flow). Mach contours for $M_\infty = 4$

For the viscous flow case, the Reynolds number is $Re = 10^4$ An initial mesh of 120×120 will be used. The Mach contours obtained by the Eulerian approach and unified coordinate approach are displayed in Figures 11.9 and 11.10, respectively. We note that the leading shocks produced by the UC approach are curves, and this is due to the effect of boundary layer displacement, which makes the effective solid surface from a flat one to a parabolic arc. This is less visible in Eulerian computation.
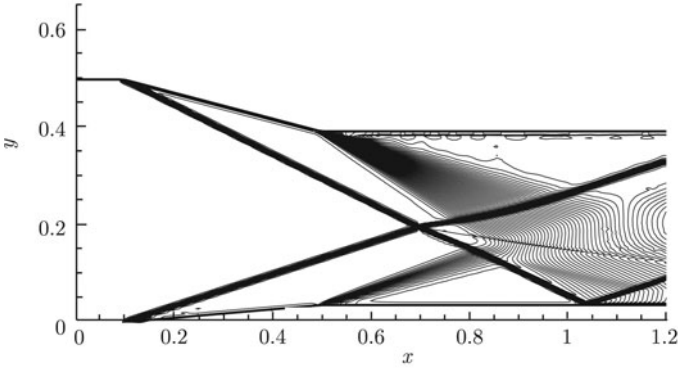
Figure 11.8   UC computation for shock/shock interaction (inviscid flow). Mach contours for $M_\infty = 4$
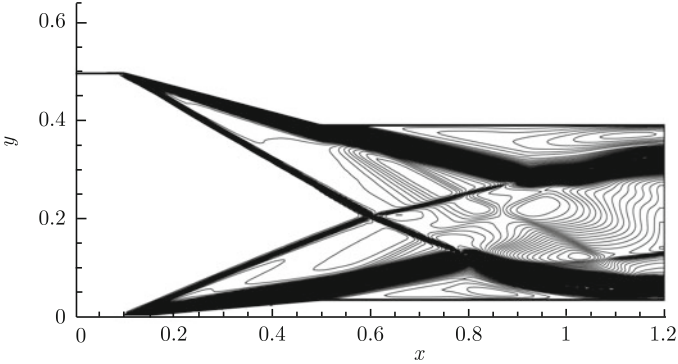
Figure 11.9   Eulerian computation for shock/shock interaction. Mach contours for $M_\infty = 4$ and $Re = 10^4$

The success of this computation clearly shows that the unified coordinate system works even for a very complex problem for which the classical Lagrangian approach

should fail to work. Notably, it captures discontinuities more sharply in the inviscid flow regimes, while it does not break down in the viscous flow regime. This is due to the use of an h-equation (or g-equation) which avoids severe mesh deformation.
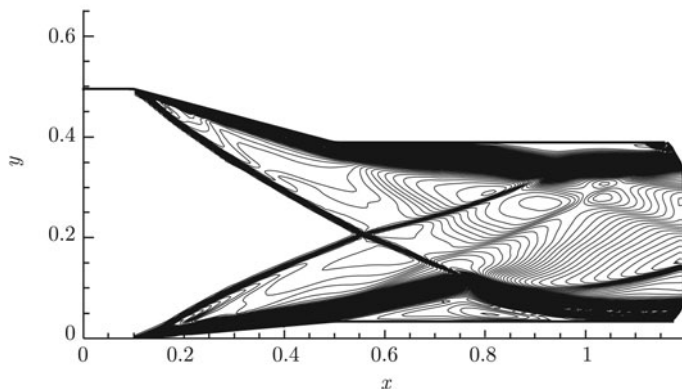
Figure 11.10   UC computation for shock/shock interaction. Mach contours for $M_\infty = 4$ and $Re = 10^4$

An important discrepancy is the curvature of the oblique shocks produced by the upper and lower wedges. The Eulerian approach produces two straight shock waves (Figure 11.9). This would be qualitatively incorrect, since the increasing (displacement) thickness of the boundary layer on the wedges should make the shocks curved. This property is nevertheless well captured by the unified coordinate system, see Figure 11.10 from which it is clear that the front two oblique shock waves are curved.

We have also performed a computation using a refined mesh (240×240) for the Eulerian approach. The Mach contours is shown in Figure 11.11, from which we see that the oblique shock waves are curved.

To conclude, this chapter extends the unified coordinate system initially proposed for inviscid flow computation to the case of viscous flow computation. In contrast with the classical Lagrangian approach which fails to work for computing boundary layer flow due to infinite mesh deformation, the unified coordinate system—a generalized Lagrangian system-works for boundary layer flow computations due to its freedom in constructing the mesh velocity.

The unified coordinate system is expected to work better than the Eulerian

approach since boundary layers can be considered as the superposition of an infinite number of slip lines and the unified coordinate system works well with sliplines.
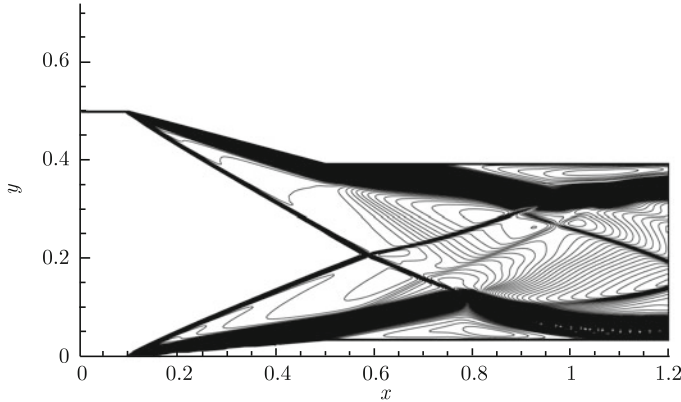
Figure 11.11   Eulerian approach for shock/shock interaction with a mesh 240×240 points. Mach contours for $M_\infty = 4$ and $Re = 10^4$

References

[1] W.H. HUI, P.Y. LI AND Z.W. LI. A unified coordinate system for solving the two-dimensional Euler equations. *J. Comput. Phys.*, 153: 596-637, 1999.

[2] W.H. HUI AND D.L. CHU. Optimum grid for the steady Euler equations. *Computational Fluid Dynamics Journal*, 4: 403-426, 1996.

[3] G. STRANG. On the construction and comparison of difference schemes. *SIAM J. Num. Anal.*, 5: 506-517, 1968.

[4] W.H. HUI, Z.N. WU AND B. GAO. Preliminary Extension of the Unified Coordinate Approach to Computation of Viscous Flows. *J. Sci. Comput.*, 30: 301-344, 2007.

[5] J.M. DELERY. Shock wave/boundary layer interactions. in *Handbook of Shock Waves*, Ben-Dor G , Igra O , Elperin T (eds), Academic Press, San Diego. Vol. II, 205-261, 2001.

Chapter 12

Applications of the Unified Coordinates to Kinetic Theory

12.1 Brief Introduction of Gas-Kinetic Theory

Besides the macroscopic governing equations—the Navier-Stokes equations, a fundamentally different approach to describing viscous flow is based on the microscopic particle (molecule) motion—the so-called Boltzmann equation. The fluid density is defined as a collection of individual molecules

$$\rho = \sum_i m n_i, \tag{12.1}$$

where m is the molecular mass and n_i is the particle number density at a certain velocity. However, due to the huge number of particles in a small volume in common situations, such as $\sum_i n_i = 2.7 \times 10^{19}$ moleculars in 1 cubic centimeter at 1 atmosphere and $T = 0°\mathrm{C}$, to follow each individual particle is impossible. Instead, a continuous distribution function is used to describe the probability of particles to be located in a certain velocity interval. For hydrodynamics purpose, n_i is approximated by a gas distribution function

$$f(x_i, t, c_i),$$

where (x_i, t) is the location of any point in Eulerian space and time, $c_i = (c_x, c_y, c_z)$ is particle velocity with three components in the x-, y-and z-direction, and the relation between n_i and f is

$$m n_i = f(x_i, t, c_i).$$

As a result, the sum in (12.1) can be replaced by the integral

$$\rho = \iiint f \mathrm{d}c_x \mathrm{d}c_y \mathrm{d}c_z$$

in the particle velocity space. For molecules with internal motion, such as rotation and vibration, the distribution function f can take these internal motion into account as well through additional variables ς_i. The dimension and formulation for ς_i are defined below.

For monatomic gas, the internal degree of freedom N is equal to 0. For diatomic gases, under the normal pressure and temperature, N is equal to 2 which accounts for two independent rotational degrees of freedom. The equipartition principle in statistical mechanics states that each degree of freedom shares an equal amount of energy $\frac{1}{2}kT$, where k is the Boltzmann constant and T is the temperature. Then the specific heats C_v and C_p for the gases in equilibrium state have the forms

$$C_v = \frac{N+3}{2}R, \quad C_p = \frac{(N+3)+2}{2}R, \tag{12.2}$$

where $R = k/m$ is the gas constant, m is the mass of each molecule, and the three accounts for the molecular translation motion in x-, y-and z-direction. From the above equations, we can obtain the ratio of the principal specific heats, which is commonly denoted by γ,

$$\gamma = \frac{C_p}{C_v} = \frac{(N+3)+2}{N+3}. \tag{12.3}$$

So γ is 5/3 for monatomic gas ($N = 0$) and 7/5 for diatomic gas ($N = 2$).

The thermodynamic aspect of the Navier-Stokes equations is based on the assumption that the departure of the gas from local equilibrium state is sufficiently small. Although we do not know the real gas distribution function f exactly in the real flow situation, in classical physics we do know the corresponding equilibrium state g locally once we know the mass, momentum and energy densities. In the following, we are going to define the equilibrium distribution and present all its physical properties. In order to understand the internal variable ς_i inside the gas distribution function, let's first write down the Maxwell-Boltzmann distribution g for the equilibrium state,

$$g = \rho\left(\frac{m}{2\pi kT}\right)^{\frac{N+3}{2}} \exp\{-(m/2kT)[(c_i - u_i)^2 + \varsigma_i^2]\}$$

$$= \rho\left(\frac{m}{2\pi kT}\right)^{\frac{N+3}{2}} \exp\{-(m/2kT)[(c_x-u)^2+(c_y-v)^2+(c_z-w)^2+\varsigma_1^2+\cdots+\varsigma_N^2]\},$$

$$\tag{12.4}$$

where $\varsigma_i = (\varsigma_1, \varsigma_2, \cdots, \varsigma_N)$ are the components of the internal particle velocity in N-D, $u_i = (u, v, w)$ is the corresponding macroscopic flow velocity with three components in the x-, y-and z-direction. In the above equation, the parameters T, u_i and ρ, which determine g uniquely, are functions of space and time. Taking moments of the equilibrium state g, the mass, momentum and energy densities at any point in space and time can be obtained. In particular, the macroscopic and microscopic descriptions are related by

$$\begin{pmatrix} \rho \\ \rho u_i \\ \rho e \end{pmatrix} = \int g \begin{pmatrix} 1 \\ c_i \\ \frac{1}{2}(c_i^2 + \varsigma^2) \end{pmatrix} dc_x dc_y dc_z d\varsigma_1 d\varsigma_2 \cdots d\varsigma_N. \tag{12.5}$$

More specifically,

$$\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{pmatrix} = \int \cdots \int_{-\infty}^{\infty} g \begin{pmatrix} 1 \\ c_x \\ c_y \\ c_z \\ \frac{1}{2}(c_x^2 + c_y^2 + c_z^2 + \varsigma_1^2 + \cdots + \varsigma_N^2) \end{pmatrix} dc_x dc_y dc_z d\varsigma_1 \cdots d\varsigma_N, \tag{12.6}$$

from which the total energy density ρe can be expressed as

$$\rho e = \frac{1}{2}\rho \left(u^2 + v^2 + w^2 + \frac{N+3}{m}kT \right),$$

which includes both kinetic and thermal energy densities. Note that (12.4) describes the gas distribution function g in 3-D and the value of N can be obtained in terms of γ from (12.3). If we re-define the internal variable ς_i as a vector in K-D, in the 3-D case we have

$$K = N = \frac{-3\gamma + 5}{\gamma - 1}.$$

For 1-D gas flow, the macroscopic average velocities in y-and z-direction are equal to zero with $(v, w) = (0, 0)$. So, the random motion of particles in y-and z-direction can be included in the internal variable ς of the molecules. As a result, the internal degree of freedom becomes $N + 2$, which is denoted again by K with the relation $K = N + 2$. The distribution function g in the 1-D case goes to

$$g = \rho \left(\frac{m}{2\pi kT} \right)^{\frac{N+3}{2}} \exp\{-(m/2kT)[(c_x - u)^2 + c_y^2 + c_z^2 + \varsigma_1^2 + \cdots + \varsigma_N^2]\} \tag{12.7}$$

$$= \rho\left(\frac{m}{2\pi kT}\right)^{\frac{K+1}{2}} \exp\{-(m/2kT)[(c_x - u)^2 + \varsigma^2]\},$$

where the dimension of ς is K. Substitue $N = K - 2$ into (12.3), we get the relation between K and γ in the 1-D case,

$$K = \frac{3 - \gamma}{\gamma - 1}.$$

For example, for diatomic gas with $N = 2$ and $\gamma = 1.4$, K is equal to 4, and the total energy density goes to

$$\rho e = \frac{1}{2}\rho\left[u^2 + \frac{(K+1)}{m}kT\right].$$

In 2-D flow calculations, K is equal to $N + 1$, and the equilibrium distribution function is

$$g = \rho\left(\frac{m}{2\pi kT}\right)^{\frac{N+3}{2}} \exp\{-(m/2kT)[(c_x - u)^2 + (c_y - v)^2 + c_z^2 + \varsigma_1^2 + \cdots + \varsigma_N^2]\}$$

$$= \rho\left(\frac{m}{2\pi kT}\right)^{\frac{K+2}{2}} \exp\{-(m/2kT)[(c_x - u)^2 + (c_y - v)^2 + \varsigma^2]\}. \tag{12.8}$$

Then the relation between γ and K becomes

$$K = \frac{4 - 2\gamma}{\gamma - 1}.$$

For diatomic gas, K is equal to 3 in the 2-D case and the total energy density becomes

$$\rho e = \frac{1}{2}\rho\left[u^2 + v^2 + \frac{(K+2)}{m}kT\right].$$

In all cases from the 1-D to 3-D, the pressure p is related to ρ and T through the following relation:

$$p = nkT = \frac{\rho}{m}kT = \rho RT,$$

where n is the particle number density, R is the gas constant and m is the molecule mass. Note that the pressure is independent of the internal degree of freedom N.

Due to the unique format of the equilibrium distribution function g in classical statistical physics, at each point in space and time, there is a one to one correspondence between g and the macroscopic densities, e.g., mass, momentum and energy. So, from macroscopic flow variables at any point in space and time, we

can construct a unique equilibrium state. However, in real physical situation, gas does not necessarily stay in the Local Thermodynamic Equilibrium (LTE) state, such as gas inside a shock or boundary layer, even though we can still construct a local equilibrium state there from the corresponding macroscopic flow variables. Usually, we do not know the explicit form of the gas distribution function f in extremely dissipative flow regions, such as that inside a strong shock wave. What we know is the time evolution of f, the so-called Boltzmann Equation

$$f_t + c_i f_{x_i} + a_i f_{c_i} = Q(f, f) = \iiint (f' f'_* - f f_*) B(\theta, V) \mathrm{d}\boldsymbol{c}_* \mathrm{d}\theta \mathrm{d}\epsilon, \qquad (12.9)$$

where $V = |\boldsymbol{c} - \boldsymbol{c}_*|$ is the particle collision velocity differences between impact particles which will be changed to $V' = |\boldsymbol{c}' - \boldsymbol{c}'_*|$ after collision. Here f is the gas distribution function before collision with $f = f(\boldsymbol{c})$ and $f_* = f(\boldsymbol{c}_*)$, and f' is the after impact gas distribution function with $f'_* = f(\boldsymbol{c}'_*)$ and $f' = f(\boldsymbol{c}'_*)$. The function B depends on the interaction potential of particles. In the above equation, f is the real gas distribution function, a_i is the external force term acting on the particle in i-th direction, and $Q(f, f)$ is the collision operator. From the physical constraints of the conservation of mass, momentum and energy during particle collisions, the following compatibility condition has to be satisfied:

$$\int \boldsymbol{\psi}_\alpha Q(f, f) \mathrm{d}\varXi = 0, \qquad (12.10)$$

where $\mathrm{d}\varXi = \mathrm{d}c_x \mathrm{d}c_y \mathrm{d}c_z \mathrm{d}\varsigma_1 \mathrm{d}\varsigma_2 \cdots \mathrm{d}\varsigma_K$ and $\boldsymbol{\psi}_\alpha = (1, c_x, c_y, c_z, \frac{1}{2}(c_x^2 + c_y^2 + c_z^2 + \varsigma^2))^{\mathrm{T}}$. For convenience, the following notations will be used:

$$\varsigma^2 = \varsigma_1^2 + \varsigma_2^2 + \cdots + \varsigma_K^2, \quad \mathrm{d}\varsigma = \mathrm{d}\varsigma_1 \mathrm{d}\varsigma_2 \cdots \mathrm{d}\varsigma_K.$$

Assuming further that the spatial and temporal variations of the distribution function f are small on the scale of the mean free path and the mean time interval between collisions, it is possible to find the first order approximations to the viscous stress tensor and the heat flux from the Boltzmann equation, which are in agreement with the Navier-Stokes equations. Thus the Navier-Stokes equations may be regarded as the leading term in an asymptotic expansion of the full Boltzmann equation in the limit of $K_n \simeq 0$, where Kn is the Knudsen number. From the Boltzmann equation, the viscosity and heat conduction coefficients can be derived

as functions of the basic quantities describing the molecules[1]. For example, the viscous stress σ_{ij} and the heat flux κ_i can be obtained from the gas distribution function f, such that

$$\sigma_{ij} = \int (c_i - u_i)(c_j - u_j) f \mathrm{d}c_x \mathrm{d}c_y \mathrm{d}c_z \mathrm{d}\varsigma - p\delta_{ij}$$

and

$$\kappa_i = \frac{1}{2} \int (c_i - u_i)\left((c_x - u)^2 + (c_y - v)^2 + (c_z - w)^2 + \varsigma^2\right) f \mathrm{d}c_x \mathrm{d}c_y \mathrm{d}c_z \mathrm{d}\varsigma, \quad (12.11)$$

where p is the local pressure. The viscous stress σ_{ij} and heat conducting κ_i terms go to zero if and only if $f = g$ for the flow in equilibrium state.

12.2   Gas-Kinetic BGK Model Under the Unified Coordinate Transformation

The BGK model of the approximate Boltzmann equation in 2-D space can be written as[2]

$$f_t + c_x f_x + c_y f_y = \frac{g - f}{\tau}, \quad (12.12)$$

where f is the gas distribution function and g is the equilibrium state approached by f. The particle collision time τ determines the closeness between f and g. Both f and g are functions of space (x, y), time t, particle velocity (c_x, c_y), and internal variable ς. In Chapter 6, Hui's transformation can be used from the physical domain (t, x, y) to the computational domain (λ, ξ, η),

$$\begin{cases} \mathrm{d}t = \mathrm{d}\lambda, \\ \mathrm{d}x = U\mathrm{d}\lambda + A\mathrm{d}\xi + L\mathrm{d}\eta, \\ \mathrm{d}y = V\mathrm{d}\lambda + B\mathrm{d}\xi + M\mathrm{d}\eta, \end{cases} \quad (12.13)$$

where (U, V) are the mesh velocity, (A, B, L, M) are determined by the compatibility conditions or the geometric conservative laws,

$$\begin{cases} \dfrac{\partial A}{\partial \lambda} = \dfrac{\partial U}{\partial \xi}, \\[2mm] \dfrac{\partial B}{\partial \lambda} = \dfrac{\partial V}{\partial \xi}, \\[2mm] \dfrac{\partial L}{\partial \lambda} = \dfrac{\partial U}{\partial \eta}, \\[2mm] \dfrac{\partial M}{\partial \lambda} = \dfrac{\partial V}{\partial \eta}. \end{cases} \quad (12.14)$$

With the above transformation (12.13), the gas-kinetic BGK equation becomes

$$\frac{\partial}{\partial\lambda}(\Delta f)+\frac{\partial}{\partial\xi}\{[(c_x-U)M-(c_y-V)L]f\}+\frac{\partial}{\partial\eta}\{[-(c_x-U)B+(c_y-V)A]f\}=\frac{g-f}{\tau}\Delta, \tag{12.15}$$

where $\Delta=AM-BL$ is the Jacobian of the transformation.

For an equilibrium flow with distribution $f=g$, by taking the conservative moments $\boldsymbol{\phi}=\left(1,c_x,c_y,\frac{1}{2}(c_x^2+c_y^2+\varsigma^2)\right)^{\mathrm{T}}$ to (12.15), the same Euler equations (6.21) under the moving mesh in the Eulerian space can be obtained,

$$\frac{\partial\boldsymbol{E}}{\partial\lambda}+\frac{\partial\boldsymbol{F}}{\partial\xi}+\frac{\partial\boldsymbol{G}}{\partial\eta}=\boldsymbol{0}, \tag{12.16}$$

where

$$\boldsymbol{E}=\begin{pmatrix}\rho J\\\rho Ju\\\rho Jv\\\rho Je\\A\\B\\L\\M\end{pmatrix},\quad\boldsymbol{F}=\begin{pmatrix}\rho X\\\rho Xu+pM\\\rho Xv-pL\\\rho Xe+p(uM-vL)\\-U\\-V\\0\\0\end{pmatrix},\quad\boldsymbol{G}=\begin{pmatrix}\rho Y\\\rho Yu-pB\\\rho Yv+pA\\\rho Ye+p(vA-uB)\\0\\0\\-U\\-V\end{pmatrix}. \tag{12.17}$$

For the viscous and heat conducting flow, the Chapman-Enskog expansion of (12.15) up to the 1st order of τ gives

$$f=g-\frac{\tau}{\Delta}\left(\frac{\partial}{\partial\lambda}(\Delta g)+\frac{\partial}{\partial\xi}\{[(c_x-U)M-(c_y-V)L]g\}+\frac{\partial}{\partial\eta}\{[-(c_x-U)B+(c_y-V)A]g\}\right).$$

Taking moments of $\boldsymbol{\phi}$ again to (12.15) with the above Navier-Stokes distribution function, we can get the Navier-Stokes equations in moving space, which are presented in Section 11.1. In this chapter, instead of solving the viscous governing equations, we are going to solve the gas-kinetic equation (12.15) to compute viscous flow.

12.3   Numerical BGK-NS Scheme in a Moving Mesh System

In this section, we are going to present the gas-kinetic method to solve (12.15) by a directional splitting method. For example, the BGK model (12.15) in the

ξ-direction is

$$\frac{\partial}{\partial \lambda}(\Delta f) + \frac{\partial}{\partial \xi}\{[(c_x - U)M - (c_y - V)L]f\} = \frac{g - f}{\tau}\Delta. \tag{12.18}$$

In order to evaluate the fluxes across a moving interface $\xi = $ const., let's first define its normal direction $\boldsymbol{n}$ and tangent direction $\boldsymbol{t}$,

$$\boldsymbol{n} = \nabla\xi/|\nabla\xi| = \frac{(M, -L)}{S}, \quad \boldsymbol{t} = \frac{(L, M)}{S},$$

where $S = \sqrt{M^2 + L^2}$ is the physical length of the cell interface. Then, the particle velocity $(c_x - U, c_y - V)$ relative to a moving cell interface can be decomposed into the normal $\tilde{c}_x$ and tangential $\tilde{c}_y$ velocities as well, namely

$$\begin{cases} \tilde{c}_x = \dfrac{(c_x - U)M}{S} - \dfrac{(c_y - V)L}{S}, \\[2mm] \tilde{c}_y = \dfrac{(c_x - U)L}{S} + \dfrac{(c_y - V)M}{S}. \end{cases} \tag{12.19}$$

Hence, with the above transformation, (12.18) in the ξ-direction becomes

$$\frac{\partial}{\partial \lambda}(\Delta f) + \frac{\partial}{\partial \xi}(S\tilde{c}_x f) = \frac{g - f}{\tau}\Delta. \tag{12.20}$$

This is the basic equation to be solved to construct the gas distribution function f at the moving cell interface $\xi = $ const., then calculate the numerical fluxes. In the above equation, Δ is the cell area and S is the cell interface length. At the center of a moving cell interface the above equation can be re-written as

$$\frac{\partial}{\partial \lambda}(f) + \frac{\partial}{\partial \tilde{x}}(\tilde{c}_x f) = \frac{g - f}{\tau}, \tag{12.21}$$

where $\tilde{x}$ is the length scale in the normal direction of the moving cell interface in the physical space. Since $\mathrm{d}\lambda = \mathrm{d}t$, the integral solution of the above equation becomes

$$\begin{aligned} f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma) = &\frac{1}{\tau}\int_0^t g(\tilde{x}', t', \tilde{c}_x, \tilde{c}_y, \zeta)\mathrm{e}^{-(t-t')/\tau}\mathrm{d}t' \\ &+ \mathrm{e}^{-t/\tau}f_0(\tilde{x}_{i+1/2} - \tilde{c}_x t, \tilde{c}_y), \end{aligned} \tag{12.22}$$

where $\tilde{x}' = \tilde{x}_{i+1/2} - \tilde{c}_x(t - t')$ is the trajectory of a particle motion relative to the moving cell interface and f_0 is the initial gas distribution function f at the beginning of each time step $(t = 0)$. The scheme based on the above solution

will be identical to the BGK-NS method[3], even though the system here $(\lambda, \tilde{x})$ is moving relative to the stationary system (t, x). The difference only appears in the construction of the equilibrium states in both g and f_0, which are presented in the following.

In the local moving frame of reference at interface $\xi = \text{const.}$, the Maxwellian distribution should have the form

$$g = \rho \left(\frac{m}{2\pi kT} \right)^{(K+2)/2} \exp\{-(m/2kT)[(\tilde{c}_x - \tilde{u})^2 + (\tilde{c}_y - \tilde{v})^2 + \varsigma^2]\},$$

where the averaged macroscopic fluid velocity $(\tilde{u}, \tilde{v})$ is related to the fluid velocity (u, v) in the inertia frame of reference by

$$\begin{cases} \tilde{u} = \dfrac{(u - U)M}{S} - \dfrac{(v - V)L}{S}, \\[2mm] \tilde{v} = \dfrac{(u - U)L}{S} + \dfrac{(v - V)M}{S}. \end{cases} \tag{12.23}$$

Numerically, (12.21) is the same equation as the one we have solved in the Eulerian space, where $\tilde{c}_x, \tilde{c}_y$ are the particle velocity and $\tilde{u}, \tilde{v}$ are the macroscopic velocity in the $\boldsymbol{n}$ and $\boldsymbol{t}$-direction. Then, the standard BGK-NS method[3] can be used to solve (12.21) to evaluate the time-dependent gas distribution function $f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma)$ at the cell interface $\xi = \xi_{i+1/2}$. The detailed formulation of the gas-kinetic BGK-NS scheme for the Navier-Stokes solutions is given in [3]. Therefore, standing on the moving cell interface the fluxes can be explicitly obtained,

$$\begin{pmatrix} \mathcal{F}_\rho \\ \mathcal{F}_{\rho\tilde{u}} \\ \mathcal{F}_{\rho\tilde{v}} \\ \mathcal{F}_{\rho\tilde{e}} \end{pmatrix}_{i+1/2,j} = \int \tilde{c}_x \begin{pmatrix} 1 \\ \tilde{c}_x \\ \tilde{c}_y \\ \frac{1}{2}(\tilde{c}_x^2 + \tilde{c}_y^2 + \varsigma^2) \end{pmatrix} f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma) d\Xi. \tag{12.24}$$

The above fluxes are evaluated standing on the moving reference of frame with mesh velocity (U, V), in order to update the conservative variables in the (ξ, η) space, we need to transfer the fluxes in (12.24) into the fluxes for the mass, momentum and energy transport of the inertia frame of reference. In other words, the above obtained gas distribution function $f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma)$ and its mass flux across the moving cell interface $\tilde{c}_x f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma)$ will carry the mass, momentum

and energy densities $\left(1, c_x, c_y, \frac{1}{2}(c_x^2 + c_y^2 + \varsigma^2)\right)$ in the inertia frame of reference. So, the time-dependent numerical flux in the Eulerian space in the $\boldsymbol{n}$-direction across the cell interface $\xi = C$ should be calculated as

$$\begin{pmatrix} \mathcal{F}_\rho \\ \mathcal{F}_{\rho u} \\ \mathcal{F}_{\rho v} \\ \mathcal{F}_{\rho e} \end{pmatrix}_{i+1/2,j} = \int S\tilde{c}_x \begin{pmatrix} 1 \\ c_x \\ c_y \\ \frac{1}{2}(c_x^2 + c_y^2 + \varsigma^2) \end{pmatrix} f(\xi_{i+1/2}, \eta_j, t, \tilde{c}_x, \tilde{c}_y, \varsigma) \mathrm{d}\Xi. \quad (12.25)$$

In the above equation, the distribution function f is a scalar function, which is invariant under coordinate transformation, but the particle velocities $(\tilde{c}_x, \tilde{c}_y)$ and (c_x, c_y) are defined differently in the different frame of reference. In order to evaluate the above flux integration, the easiest way is to write the (c_x, c_y) velocities in terms of $(\tilde{c}_x, \tilde{c}_y)$. Based on the transformation (12.19), we have

$$c_x = U + \frac{M\tilde{c}_x + L\tilde{c}_y}{S}, \quad c_y = V + \frac{-L\tilde{c}_x + M\tilde{c}_y}{S}.$$

Therefore, (12.25) becomes

$$\begin{pmatrix} \mathcal{F}_\rho \\ \mathcal{F}_{\rho u} \\ \mathcal{F}_{\rho v} \\ \mathcal{F}_{\rho e} \end{pmatrix}_{i+1/2,j} = \begin{pmatrix} S\mathcal{F}_\rho \\ M\mathcal{F}_{\rho\tilde{u}} + L\mathcal{F}_{\rho\tilde{v}} + SU\mathcal{F}_\rho \\ -L\mathcal{F}_{\rho\tilde{u}} + M\mathcal{F}_{\rho\tilde{v}} + SV\mathcal{F}_\rho \\ (MU - LV)\mathcal{F}_{\rho\tilde{u}} + (LU + MV)\mathcal{F}_{\rho\tilde{v}} + S\mathcal{F}_{\rho\tilde{e}} + \frac{S}{2}(U^2 + V^2)\mathcal{F}_\rho \end{pmatrix},$$
$$(12.26)$$

where $(\mathcal{F}_\rho, \mathcal{F}_{\rho\tilde{u}}, \mathcal{F}_{\rho\tilde{v}}, \mathcal{F}_{\rho\tilde{e}})$ are given in (12.24). So the fluxes relative to the moving cell interface in the Eulerian space is just a linear combination of the fluxes in the moving frame of reference due to the linear transformation between the inertia and moving space with relative velocity (U, V). Similarly, the fluxes at the cell interface $\eta = \text{const.}$, i.e., $\boldsymbol{G}$, can be constructed as well.

With the above fluxes, the flow variables can be updated in each moving computational cell by

$$\boldsymbol{E}_{i,j}^{n+1} = \boldsymbol{E}_{i,j}^n + \frac{1}{\Delta\xi}\int_{t^n}^{t^{n+1}} (\boldsymbol{F}_{i-1/2,j} - \boldsymbol{F}_{i+1/2,j})\mathrm{d}t + \frac{1}{\Delta\eta}\int_{t^n}^{t^{n+1}} (\boldsymbol{G}_{i,j-1/2} - \boldsymbol{G}_{i,j+1/2})\mathrm{d}t,$$
$$(12.27)$$

where $\boldsymbol{E} = (\rho\Delta, \rho\Delta u, \rho\Delta v, \rho\Delta e)^{\mathrm{T}}$ and $\boldsymbol{F} = (\mathcal{F}_\rho, \mathcal{F}_{\rho u}, \mathcal{F}_{\rho v}, \mathcal{F}_{\rho e})^{\mathrm{T}}$ are given in (12.26), $\boldsymbol{G}$ fluxes in the η-direction.

12.4   Numerical Procedure

The numerical procedure for the gas-kinetic scheme on a moving and adaptive mesh can be summarized in the following. In some numerical examples, the adaptive mesh method is not used, and the corresponding step can be ignored. Also, all fluid quantities $(\rho, \rho u, \rho v, \rho e)$ and the geometric variables (A, B, M, L) are defined at each cell center. Any cell interface values are obtained as the average of the values in the adjacent cells.

Step 1   Initialize the conservative variables $(\rho, \rho u, \rho v, \rho e)$ and (A, B, L, M) at $t = \lambda = 0$ in the xy plane. Usually, ξ and η are the initial arclength of their corresponding x-and y-coordinate lines. For example, for the rectangular domain, we take $\Delta \xi = \Delta x$, and $\Delta \eta = \Delta y$ when Δx and Δy are constants on the physical domain initially. Or, we can simply choose two constants to define $\Delta \xi$ and $\Delta \eta$. Then, (A, B, L, M) are determined according to the definition,

$$
\begin{cases}
A = \dfrac{\partial x}{\partial \xi}, \\[2mm]
B = \dfrac{\partial y}{\partial \xi}, \\[2mm]
L = \dfrac{\partial x}{\partial \eta}, \\[2mm]
M = \dfrac{\partial y}{\partial \eta}.
\end{cases}
\tag{12.28}
$$

Then, we calculate $(\Delta \rho, \Delta \rho u, \Delta \rho v, \Delta \rho E)$ inside each cell with $\Delta = AM - BL$.

Step 2   Construct or modify the mesh using the adaptive grid method[4], and update the conservative variables $(\rho, \rho u, \rho v, \rho e)$ inside each cell in the physical domain. Subsequently, calculate (A, B, L, M) by (12.28) again and update $(\Delta \rho, \Delta \rho u, \Delta \rho v, \Delta \rho e)$ with the new $\Delta = AM - BL$. The re-distribution of conservative variables in this step is fully conservative.

Step 3   Given a mesh velocity (U, V) at the center of each cell, such as the fluid velocity $\boldsymbol{q}$ in the gas-kinetic Lagrangian method, at the cell interface $\xi = \xi_{i+1/2}$ the mesh velocity is calculated as

$$
(U)_{i+1/2,j} = \frac{(U)_{i,j} + (U)_{i+1,j}}{2},
$$

$$
(V)_{i+1/2,j} = \frac{(V)_{i,j} + (V)_{i+1,j}}{2},
$$

which are the average of the velocities from the neighboring cells. Then, based on (12.26) the numerical fluxes are calculated across the moving interface. At the same time, the center of the cell, $(x_{i,j}^n, y_{i,j}^n)$, moves to a new location through

$$
\begin{cases}
x_{i,j}^{n+1} = x_{i,j}^n + U_{i,j} \times (t^{n+1} - t^n), \\
y_{i,j}^{n+1} = y_{i,j}^n + V_{i,j} \times (t^{n+1} - t^n).
\end{cases} \tag{12.29}
$$

The location of cell vertex is updated by averaging the centers of 4 neighboring cells.

Step 4   With the new mesh location x^{n+1}, y^{n+1}, calculate (A, B, L, M) and Δ using (12.28). Then, update the conservative variables by the finite volume scheme (12.27) in the newly moved cells. It can be shown that the use of (12.28) to calculate (A, B, L, M) is the same as the geometrical conservation law (12.14), such as the update of A,

$$
\begin{aligned}
A_{i,j}^{n+1} &= \left(\frac{\partial x}{\partial \xi}\right)_{i,j}^{n+1} \\
&= \frac{x_{i+1/2,j}^{n+1} - x_{i-1/2,j}^{n+1}}{\Delta \xi} \\
&= \frac{x_{i+1/2,j}^n + (U)_{i+1/2,j}^n \Delta \lambda^n - x_{i-1/2,j}^n - (U)_{i-1/2,j}^n \Delta \lambda^n}{\Delta \xi} \\
&= \frac{x_{i+1/2,j}^n - x_{i-1/2,j}^n}{\Delta \xi} + \frac{\Delta \lambda^n}{\Delta \xi}((U)_{i+1/2,j}^n - (U)_{i-1/2,j}^n) \\
&= A_{i,j}^n + \frac{\Delta \lambda^n}{\Delta \xi}((U)_{i+1/2,j}^n - (U)_{i-1/2,j}^n).
\end{aligned}
$$

Similarly, it is true for B, L and M.

Then go to Step 2 to repeat the above process until the output time.

12.5   Numerical Examples

As mentioned earlier, the main advantage of the current method is that the physical conservation laws under unified coordinate are coupled with the geometrical conservation laws. Mathematically, the unified system provides an enlarged and complete system, which avoids the difficulties in other moving mesh methods to implement the geometrical conservation separately based on physical intuition. Also, different from the Eulerian and Lagrangian methods, the choice of mesh velocity (U, V) becomes a new degree of freedom in the current model. The proper choice of

the mesh velocity depends on the specific applications to get the optimum accuracy in the numerical solution. For example, the mesh velocity can be the fluid velocity in the free surface flow, or follow the oscillating angular velocity for a pitching airfoil. In the approach by Hui et al[5], preserving the mesh angle has been used to get the local value h in the determination of mesh velocity (hu, hv). In this section, the BGK-NS scheme on a moving mesh will be tested on several examples. Different choices of the mesh velocity will be used. The numerical results are compared with the exact solutions, experimental data, and the available solutions obtained by others.

Example 1: free surface flow. The current approach with the choice of $(U, V) = (u, v)$ becomes a purely gas-kinetic Lagrangian method even though it is solving the viscous governing equations. The use of the fluid velocity as the mesh velocity can naturally capture the free surface sharply. The case we are going to study is the dam break problem, where a column of water is released by removing a vertical diaphragm. This becomes a standard benchmark problem due to its simple geometry and the available experimental measurement[6]. The initial configuration is shown in the upper left picture in Figure 12.1. In this example, a rectangular column of water in hydrostatic equilibrium is confined initially between two vertical walls. The water column is 3.5 units wide and 7.0 units high. The gravity is acting downward with the dimensionless 0.05 unit in magnitude.

After the diaphragm eruption at time $t = 0$, the water is pushing out and moves freely along a dry horizontal floor. The measured quantities include the water wave

time = 0.0   time = 9.46
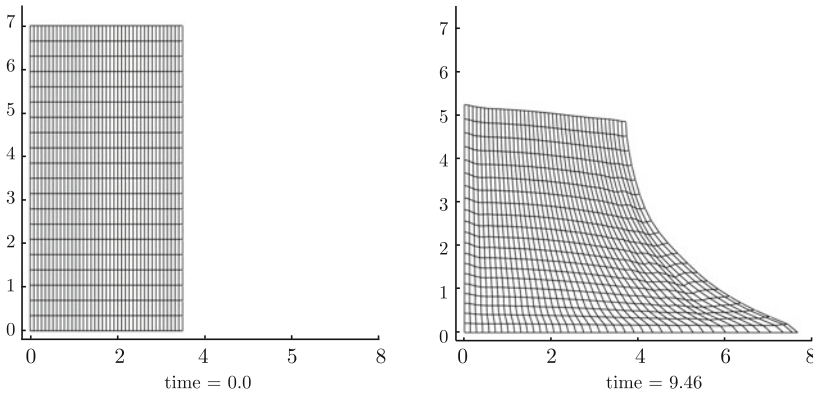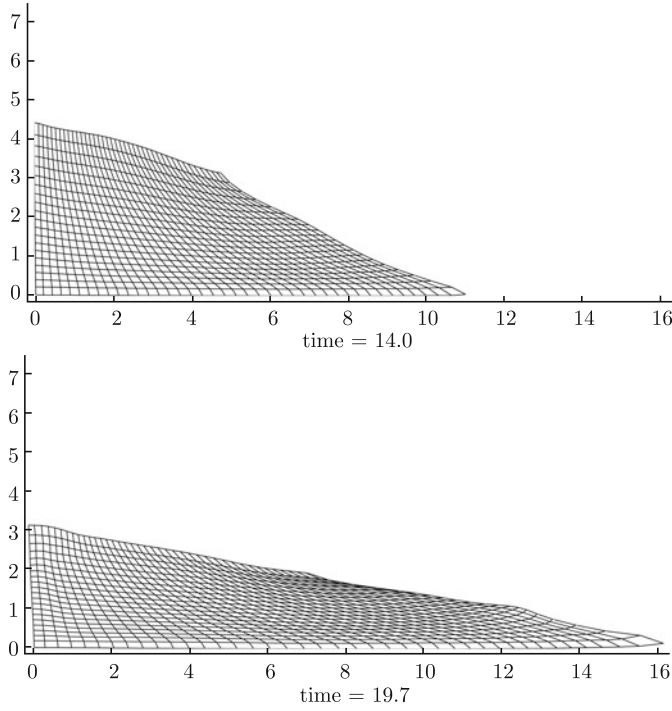
Figure 12.1   Moving meshes at time $t = 0, 9.46, 14.0, 19.7$

front location L on the floor. In our calculation, 40×20 rectangular mesh points are initially employed in a domain $0 \leqslant x \leqslant 3.5, 0 \leqslant y \leqslant 7.5$. Figure 12.1 presents the moving meshes at three subsequent times. Since the mesh velocity is equal to the fluid velocity, the mesh distribution is the same as the water distribution. In a purely Lagrangian simulation, the mesh is easily tangled at a later time. Since we have used a mesh smoothing technique through the mesh adaptive method with a constant monitor function to equally distribute the mesh[4], the moving mesh becomes generally smooth all over the domain even though the mesh adaptation steps are applied only a few times.

Figure 12.2 shows the water tip location versus time for both simulation and experimental measurement. The non-dimensional time in the horizontal coordinate is normalized by $t\sqrt{2g/W}$, where W is the width of the initial water column. Good agreements between computation and experiment are evident.

Example 2: shock reflection inside a channel with a ramp. In order to
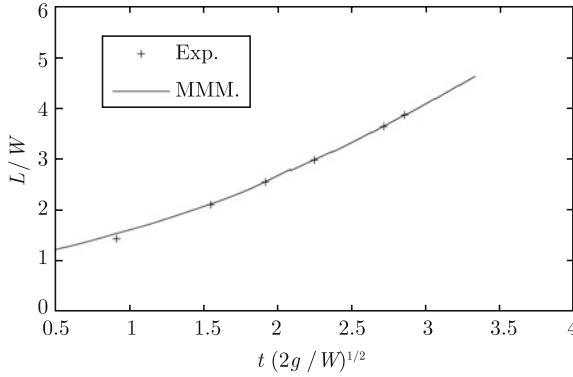
Figure 12.2   The time dependent location of water front. Experimental measurement[6] vs the calculation from UC method

preserve the mesh angles[5], the mesh velocity $(U, V) = (hu, hv)$ can be obtained by solving the following equation for h:

$$\frac{\partial}{\partial \lambda} \left(\frac{\nabla \xi}{|\nabla \xi|} \cdot \frac{\nabla \eta}{|\nabla \eta|} \right) = 0,$$

where $\nabla \xi = (M, -L)/\Delta, \nabla \eta = (-B, -A)/\Delta$ and θ is the flow angle, $u = q \cos \theta$, $v = q \sin \theta$, $q = \sqrt{u^2 + v^2}$. Let $\tilde{g} = \ln(hq)$, then the above equation becomes

$$S^2(A \sin \theta - B \cos \theta)\frac{\partial \tilde{g}}{\partial \xi} + T^2(M \cos \theta - L \sin \theta)\frac{\partial \tilde{g}}{\partial \eta}$$
$$= S^2 \left(B\frac{\partial \cos \theta}{\partial \xi} - A\frac{\partial \sin \theta}{\partial \xi} \right) - T^2 \left(M\frac{\partial \cos \theta}{\partial \eta} - L\frac{\partial \sin \theta}{\partial \eta} \right), \qquad (12.30)$$

where $S^2 = L^2 + M^2$ and $T^2 = A^2 + B^2$. Numerically, (12.30) can be solved by an iterative method to get h at different grid point. The detail description was given in [5].

In the current supersonic flow of $M = 1.8$ passing through a ramp in a channel, (12.30) is used to calculate $\tilde{g}$ and hence h, then determine the mesh velocity. Here the ramp with 15° is placed at the lower wall between $x = 0.5$ and $x = 1.0$. A computational mesh with 180×50 mesh points in the physical domain $\{0 \leqslant x \leqslant 3.6, 0 \leqslant y \leqslant 1.0\}$ is initially generated. An initial flow data with $(p, \rho, M, \theta) = (1, 1, 1.8, 0)$ is imposed inside each cell, as well as at the left boundary. Reflection boundary conditions are used at the top and bottom walls. When h is chosen according to (12.30), the mesh will automatically preserve the mesh angle,

which is shown in Figure 12.3. Figure 12.4 presents pressure and Mach number distributions after the steady state solution is obtained. The shocks and slipline are seen sharply resolved.
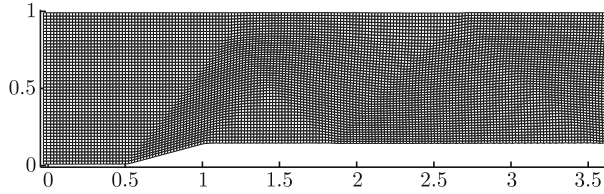
Figure 12.3   Computational mesh due to the implementation of mesh angle preservation

(a) pressure contours

(b) mach contours
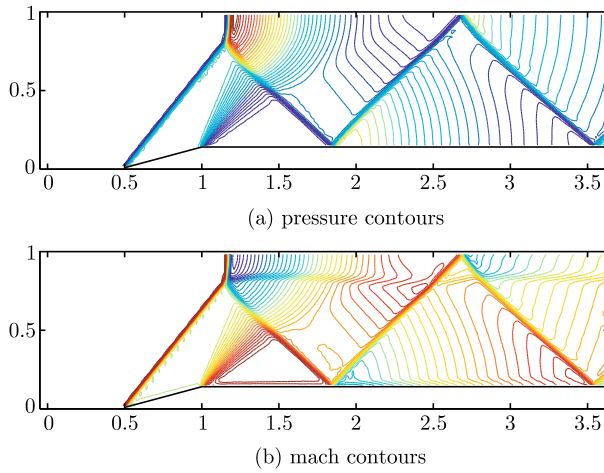
Figure 12.4   Computed flow with shock seflection in a channel

Example 3: viscous flow above an oscillating wall. This is called Stokes' second problem, which considers fluid motion above an infinite flat plate executing sinusoidal oscillations parallel to itself. The fluid above the plate is initially stationary. The governing equation for velocity u in the x-direction is

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2}, \tag{12.31}$$

with the boundary conditions

$$u_{\text{wall}}(0, t) = u_0 \cos \omega t, \quad u(\infty, t) = 0.$$

The exact solution for the above problem is

$$u(y, t) = u_0 e^{-y\sqrt{\omega/2\nu}} \cos\left(\omega t - y\sqrt{\frac{\omega}{2\nu}}\right). \tag{12.32}$$

At $y = 4\sqrt{\nu/\omega}$, the amplitude of u is equal to $u_0 \exp(-4/\sqrt{2}) = 0.05u_0$, which means that the influence from the wall is confined within a distance of order $\delta \sim 4\sqrt{\nu/\omega}$. Since the gas-kinetic scheme solves the compressible Navier-Stokes equations, in order to simulate the above incompressible limiting solution the Mach number for the compressible flow takes a small value, i.e., $M = 0.15$. The kinematic viscosity coefficient takes a value $\nu = 0.00046395$, and a mesh size 10×70 mesh points is used.

In the current calculation, we have used two ways to determine the mesh velocity. In the first case, we used the purely Lagrangian method for the viscous solution, where the mesh velocity follows the fluid one. Due to large velocity shear in the boundary layer, the Lagrangian method will stretch the mesh severely. This is the main reason why nobody really used Lagrangian method for the viscous flow computation. However, it is still theoretically interesting to see the solution using gas-kinetic Lagrangian method. Figure 12.5 shows the mesh (left) and velocity (right) at time $\omega t = \pi/25$ when the mesh follows the fluid velocity. Even though the mesh has been stretched greatly, it is surprising that the numerical solution is very close to the exact solution. This proves the robustness and accuracy of the current kinetic scheme for the viscous computation. With increasing time, the mesh will be stretched further until 20 mesh points are not enough to follow the time increasing velocity arc-length. Eventually, the computation will stop. If the
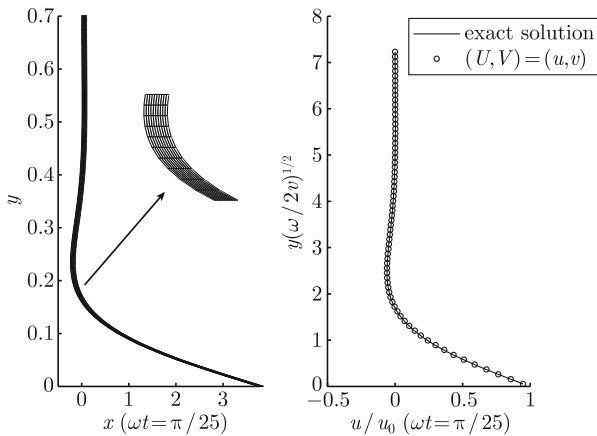
Figure 12.5   Lagrangian gas-kinetic scheme for viscous flow. Mesh (left) and velocity (right) distributions at time $\omega t = \pi/25$

mesh velocity at all mesh points follow the wall velocity, such as $(U, V) = (u_{\text{wall}}, 0)$, the mesh will not get tangled. Figure 12.6 shows the mesh and simulation results at $\omega t = \pi/2$.
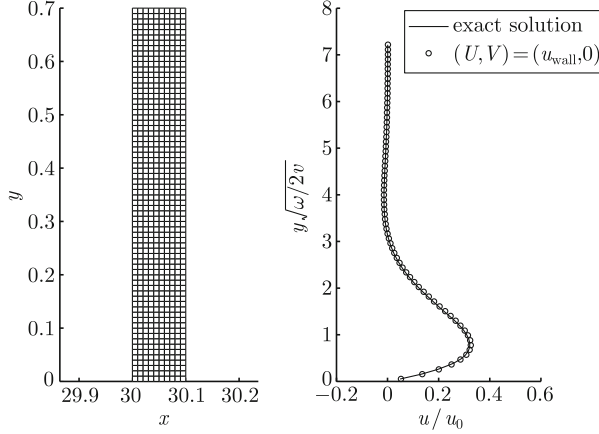
Figure 12.6   Gas-kinetic scheme for viscous flow with $(U, V) = (u_{\text{wall}}, 0)$. Mesh (left) and velocity (right) distributions at time $\omega t = \pi/2$

Example 4: freely falling plates. In the 2-D experiment conducted by Andersen, Persavento and Wang[7], a small rectangular aluminum plate was designed to fall freely in a water tank. For the falling plate, many physical quantities were measured, such as the plate trajectory and falling speed. The fluid force and torque on the plate were calculated according to the experimental data. Here we apply the above UC method to solve the 2-D Navier-Stokes equations to study the rich hydro dynamical behavior of falling plates. In the following, we simulate the plate movement under the following conditions: the plate thickness to width ratio $\beta = 1/8$, the plate thickness $h = 0.081$ cm, the density of the fluid $\rho_f = 1.0g$ cm^{-3}, the density of the plate $\rho_b = 2.7g$ cm^{-3}, and their ratio $\rho_b/\rho_f = 2.7$. In the current simulations, the Reynolds number for a rectangular plate is $Re = lu_t/\nu = 837$. The radius of the computational domain has a value about five times the length of the longer axis of the plate, i.e., $r = 5L$ and L is the chord length. In order to confirm the convergence of the computational results, a larger domain with $r = 10L$ was also used. In both cases, two stretched meshes are generated around the falling plates with 200×50 and 400×100 mesh points separately. Since the

solution close to the plate is well resolved by the small cell size and different mesh
stretching parameters are used, the simulation results on different computational
domain basically capture mesh refinement effects.

Following the experiment, the rectangle plate is released at an initial angle
of 0.25π radian with respective to the horizontal axis, with an initial velocity
$\boldsymbol{u} = (-8.92$ cm/s, -8.92 cm/s$)$. The computational mesh around the rectangle
is fixed with the rigid body motion of the plate, as shown in Figure 12.7. With the
above initial condition, the rectangular plate first has a fluttering motion from side
to side, then after a while it starts to tumble. Since the experimental results pre-
sented in [7] were for the tumbling section, the numerical data in the correspond-
ing section is extracted and compared with the experimental data. Figure 12.8
shows the rectangular plate trajectory of the current simulation and experimental
measurement[7]. The agreement is seen very good. Quantitatively, the averaged x-
and y-direction velocity components become $(V_x = 15.1$ cm/s, $V_y = -11.8$ cm/s$)$,
and the angular velocity is $(\omega = 15.0$ rad/s$)$, which have an excellent match with
the experimental results (see Table 12.1).
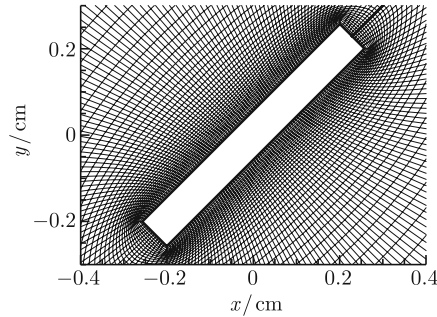
Figure 12.7   computational mesh fixed to the rectangle

Table 12.1   Experimental and numerical falling plate averaged translational and
angular velocities

	V_x /(cm/s)	V_y/(cm/s)	ω/(rad/s)
experiment (Andersen et al. [7])	15.9±0.3	−11.5±0.5	14.5±0.3
computation	15.1±0.30	−11.8±0.37	15.0±0.36

Based on the numerical computations, Figures 12.9∼12.11 present the compo-
nents of the fluid forces and their torque as a function of the plate orientation

(phase) and evolution time respectively. In order to get a comparison on the dynamical forces for different plate shapes, the computational results for an ellipse is also included [8]. In Figure 12.9, the horizontal axis of the left plot is the angle of the tumbling plate. However, the horizontal axis of the right plot uses the real evolution time. The phase differences between the rectangle and ellipse plates movement can be clearly observed from the right plot. From these figures, reasonable agreements between the experimental data [7] and the current numerical solutions are obtained. The elliptic plate rotates much faster than the rectangular one. The vorticity field for a full tumble rotation around the rectangular plate is shown in Figure 12.12.
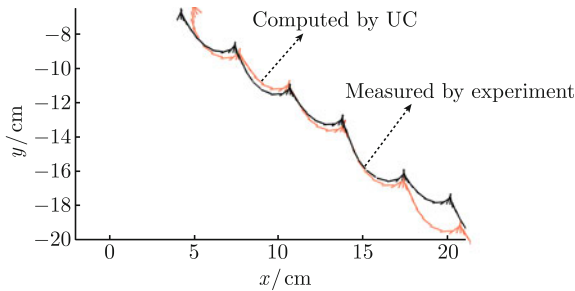
Figure 12.8   Trajectories of the falling rectangle in the tumbling section: experiment and computation solutions
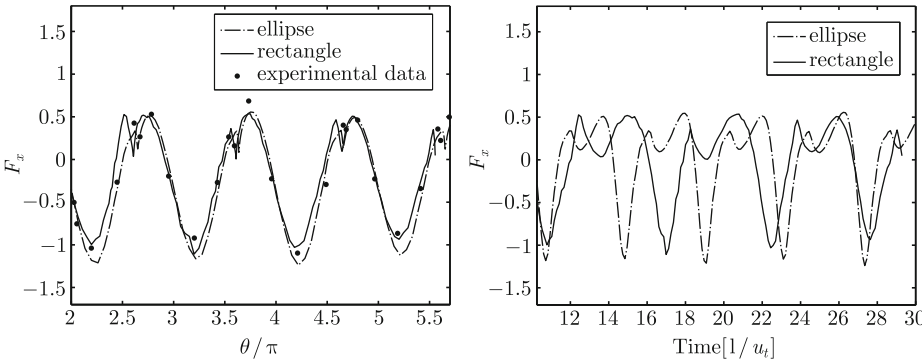
Figure 12.9 x-direction fluid force F_x on the plate during the tumbling process for both ellipse and rectangle. The horizontal axis are rotational angle (left) and dimensionless time (right)
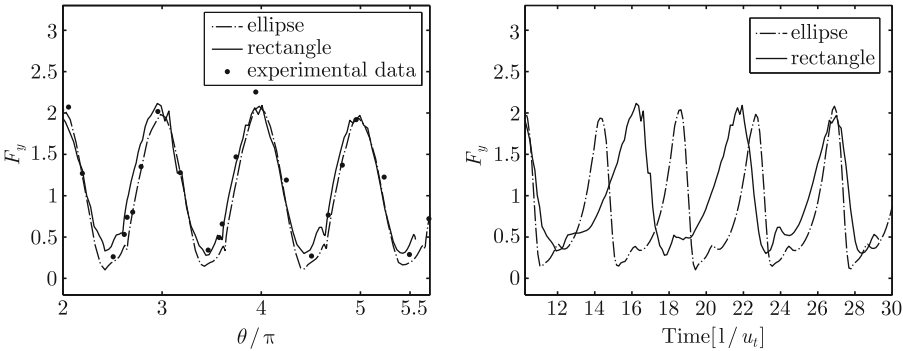
Figure 12.10 y-direction fluid force F_y on the plate during the tumbling process for both ellipse and rectangle. The horizontal axis are rotational angle (left) and dimensionless time (right)
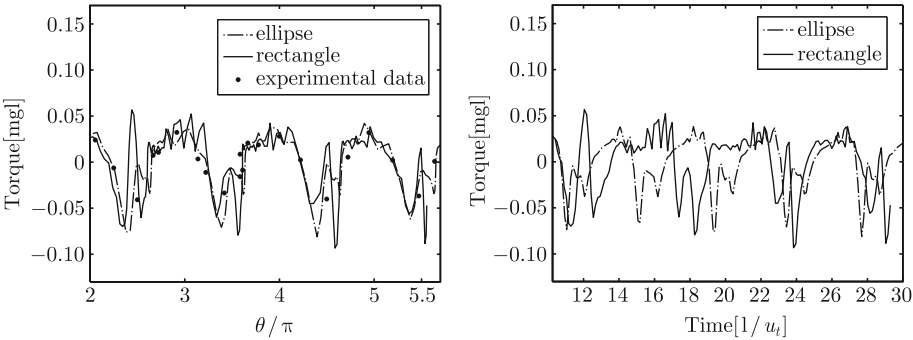
Figure 12.11   Torques on the plate during the tumbling process for both ellipse and rectangle. The horizontal axis are rotational angle (left) and dimensionless time (right)
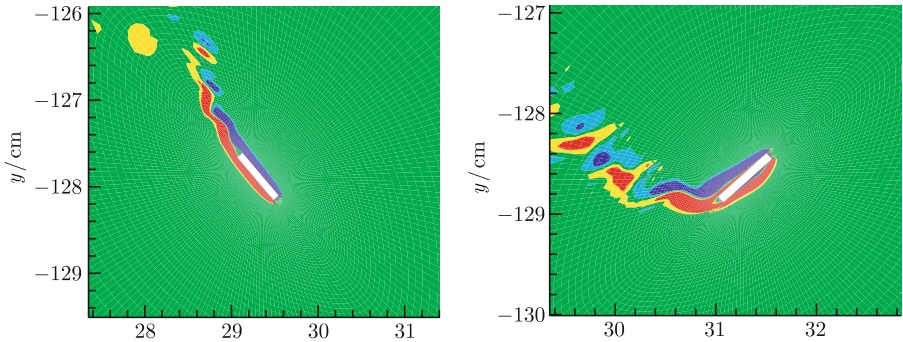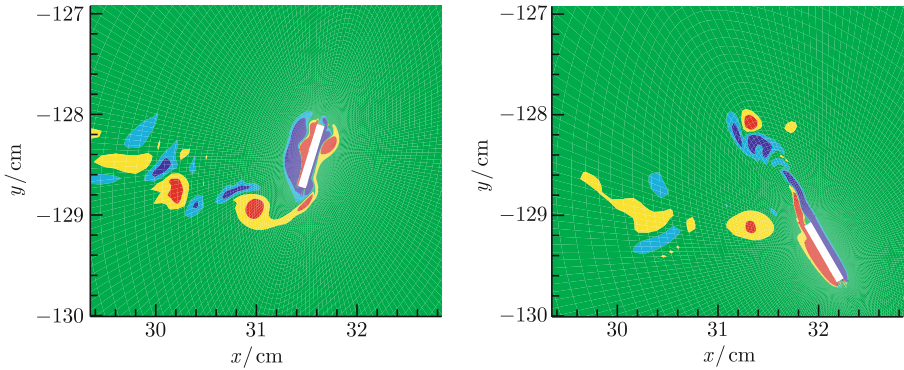
Figure 12.12   Vorticity field of a falling rectangle plate at four instants during a full rotation

12.6   Conclusion

In this chapter, the unified coordinate approach is successfully applied to the kinetic theory-based equation. In particular, the Boltzmann BGK-type equation is written in a generalized form (12.15) under Hui's unified coordinate transformation. A unified numerical scheme for the viscous solution is then developed. This is a finite volume gas-kinetic scheme on a moving mesh in the Eulerian space and the mesh velocity can be properly chosen to capture flow movement with moving boundaries. The Eulerian and Lagrangian methods are two limiting cases for the current scheme. The current unified gas-kinetic method has been applied to many flow problems, such as the free surface flow and Mach reflection inside a channel, where both inviscid and viscous solutions have been accurately obtained. The great advantage of using unified coordinates, i.e., the Hui's transformation, is that a variable and arbitrary mesh velocity can be used and the geometrical conservation laws are naturally coupled with the fluid dynamic equations. The physical and geometrical variables can be updated simultaneously. Due to the simple straight line particle motion and the relaxation term in the gas-kinetic model, the numerical treatment of the complicated viscous terms in the Navier-Stokes equations under the unified coordinate transformation can be avoided.

References

[1] S. Chapman and T.G. Cowling. *The Mathematical theory of Non-Uniform Gases*. third edition. Cambridge: Cambridge University Press, 1990.

[2] P.L. BHATNAGAR, E.P. GROSS AND M. KROOK . A Model for Collision Processes in Gases I: Small Amplitude Processes in Charged and Neutral One-Component Systems. *Phys. Rev.* , 94: 511-525, 1954.

[3] K. XU. A gas-kinetic BGK scheme for the Navier-Stokes equations and its connection with artificial dissipation and Godunov method. *J. Comput. Phys.*, 171: 289-335, 2001.

[4] C.Q. JIN AND K. XU. An adaptive grid method for two dimensional viscous flows. *J. Comput. Phys.*, 218: 68-81, 2006.

[5] W.H. HUI, P.Y. LI AND Z.W. LI. A unified coordinate system for solving the two-dimensional Euler equations. *J. Comput. Phys.*, 153: 596-637, 1999.

[6] C.W. HIRT AND B.D. NICHOLS. Volume of fluid (VOF) method for the dynamics of free surface boundaries. *J. Comput. Phys.*, 39: 210-225, 1981.

[7] A. ANDERSEN, U. PERSAVENTO AND Z. JANE WANG. Unsteady aerodynamics of fluttering and tumbling plates. *J. Fluid Mech.*, 541: 65-90, 2005.

[8] C. Q. JIN AND K. XU. Numerical study of the unsteady aerodynamics of freely falling plates. *Communications in Computational Physics*, 3:834-851, 2008.

Chapter 13
Summary

A system of unified coordinates (UC) has been introduced via transformation (6.1). It has three degrees of freedom — the mesh velocity — and unifies the traditional Eulerian and Lagrangian systems while including them as special cases. Based on (6.1), contributions are made to CFD as follows.

Theoretically,

(1) The governing equations of fluid flow in any moving coordinates can be written as a system of closed conservation PDEs; consequently, shocks can be captured correctly and the effects of mesh movement on the flow are fully accounted for.

(2) The system of Lagrangian gas dynamics equations is written in conservation PDE form, putting it on the same footing as Eulerian gas dynamics and providing a foundation for developing Lagrangian schemes as moving mesh schemes in Eulerian space.

(3) The Lagrangian system of gas dynamics equations in 2-D and 3-D is shown to be weakly hyperbolic whereas the Eulerian one is fully hyperbolic; they are, therefore, not equivalent to each other.

Computationally,

(4) The UC is superior to both Eulerian and Lagrangian systems in that contact discontinuities are resolved sharply without mesh tangling.

(5) For flow past a body, the UC avoids the tedious and time-consuming task of mesh generation; the mesh in UC is automatically generated by the flow.

(6) Boundary conditions on moving boundaries are satisfied easily in UC, making it efficient to compute fluid-solid interaction problems, as examplified by the falling-plate problem in Chapter 12.

(7) The UC approach provides a new dynamic moving mesh method which

has the distinct feature that the effects of mesh movement on the flow are fully accounted for.

Appendix A

Riemann Problem for 1-D Flow in the Unified Coordinate

The Riemann problem for 1-D unsteady flow written in the unified coordinates is

$$
\begin{cases}
\dfrac{\partial \boldsymbol{E}}{\partial \lambda} + \dfrac{\partial \boldsymbol{F}}{\partial \xi} = 0, \quad \lambda > 0, \\[2mm]
\boldsymbol{Q}(0, \xi) = \begin{cases} \boldsymbol{Q}_l, & \xi < 0, \\ \boldsymbol{Q}_r, & \xi > 0, \end{cases}
\end{cases}
\tag{A.1}
$$

where $\boldsymbol{Q}_l$ and $\boldsymbol{Q}_r$ are the constant vectors representing the flow states on the left and right side, respectively. From now on, we shall consider only the simpler case when h is a constant in the range $0 \leqslant h \leqslant 1$. The general case when h is a function of the coordinates is discussed in this book. With $h = \text{const.}$, (4.6) is a system of conservation law equations with constant coefficients and a solution to the Riemann problem depends on $\zeta = \dfrac{\lambda}{\xi}$ alone, i.e., it is a self-similar solution of the form $\boldsymbol{Q} = \boldsymbol{Q}(\zeta)$. It is constructed by piecing the smooth solutions with the discontinuous solutions.

Nonlinearity of Characteristic Fields

Case 1: $\sigma_1 = 0$. We have

$$
\boldsymbol{r}_1 \cdot \nabla \sigma_1 = 0.
\tag{A.2}
$$

Hence, the characteristic field corresponding to $\sigma_1 = 0$ is linearly degenerate.

Case 2: $\sigma_2 = \dfrac{(1-h)u}{A}$. We have

$$
\boldsymbol{r}_2 \cdot \nabla \sigma_2 = \left[\dfrac{(1-h)u}{A} \right]_\rho = 0.
\tag{A.3}
$$

So the characteristic field corresponding to $\sigma_2 = \dfrac{(1-h)u}{A}$ is also linearly degenerate.

Case 3: $\sigma_\pm = \dfrac{(1-h)u \pm a}{A}$. We have

$$\nabla \sigma_\pm = \left(\pm \frac{\gamma}{2a\rho A}, \mp \frac{a}{2\rho A}, \frac{(1-h)}{A}, -\frac{\sigma_\pm}{A} \right),$$ (A.4)

hence

$$\boldsymbol{r}_\pm \cdot \nabla \sigma_\pm = \pm \frac{\gamma+1}{2a\rho A} \neq 0$$ (A.5)

and the $\sigma_\pm$ characteristic fields are genuinely nonlinear.

Smooth Solution

The smooth solution from the $\sigma_\pm$ characteristic fields can be derived from the following system of ODE:

$$\frac{d\rho}{dp} = \frac{1}{a^2},$$ (A.6)

$$\frac{du}{dp} = \pm \frac{1}{a\rho},$$ (A.7)

$$\frac{dA}{dp} = \mp \frac{h}{\rho a \sigma_\pm}.$$ (A.8)

The solution for ρ, u, A relates the flow state $\boldsymbol{Q} = (p, \rho, u, A)^{\mathrm{T}}$ in the rarefaction fan to the initial state $\boldsymbol{Q}_0 = (p_0, \rho_0, u_0, A_0)^{\mathrm{T}}$ upstream of the fan through the following expressions:

$$\rho = \rho_0 \left(\frac{p}{p_0} \right)^{1/\gamma},$$ (A.9)

$$u \mp \frac{2a}{\gamma-1} = u_0 \mp \frac{2a_0}{\gamma-1},$$ (A.10)

$$A = A_0 \exp \left(\int_{p_0}^p \frac{\mp h \, dp}{a\rho((1-h)u \pm a)} \right).$$ (A.11)

To find the solution inside the rarefaction fan, we consider the characteristic ray through the origin $(0,0)$ and a general point (λ, ξ) inside the fan. The slope of the characteristic is

$$\frac{d\xi}{d\lambda} = \frac{\xi}{\lambda} = \sigma_\pm = \frac{(1-h)u \pm a}{A}.$$ (A.12)

Making use of (A.9)$\sim$(A.11), (A.12) becomes an equation for $p\left(\dfrac{\xi}{\lambda}\right)$ as follows:

$$f(p) = \left(\frac{\gamma + 1 - 2h}{p_0^{\frac{\gamma-1}{2\gamma}}}\right) p^{\frac{\gamma-1}{2\gamma}} - 2(1-h) + \frac{\gamma-1}{a_0}(1-h)u_0 \pm \frac{\gamma-1}{a_0}A_0\frac{\xi}{\lambda}\exp(g(p)) = 0,$$

(A.13)

where

$$g(p) = \int_{p_0}^{p} \frac{\mp h\mathrm{d}p}{a\rho((1-h)u \pm a)}.$$

(A.14)

The other flow variables ρ, u and A as functions of $\dfrac{\xi}{\lambda}$ can be easily found from (A.9)$\sim$(A.11).

Discontinuous Solutions

We start from the Rankine-Hugoniot jump conditions of the system (A.1):

$$\sigma[\rho A] = [(1-h)\rho u],$$

(A.15)

$$\sigma[\rho u A] = [(1-h)\rho u^2 + p],$$

(A.16)

$$\sigma[\rho e A] = [(1-h)\rho ue + up],$$

(A.17)

$$\sigma[A] = -[hu],$$

(A.18)

where $[.]$ denotes the jump across the discontinuity whose speed is denoted by $\sigma = \dfrac{\mathrm{d}\xi}{\mathrm{d}\lambda}$.

Case 1: shock wave. We denote the pre-shock flow state by $\boldsymbol{Q}_0 = (p_0, \rho_0, u_0, A_0)^{\mathrm{T}}$ and the post shock flow state by $\boldsymbol{Q} = (p, \rho, u, A)^{\mathrm{T}}$, respectively. Then the shock jump relations after some algebraic manipulations can be expressed in terms of $\alpha = \dfrac{p}{p_0}$ as follows:

$$\rho = \rho_0 \frac{\alpha(\gamma+1) + \gamma - 1}{\alpha(\gamma-1) + \gamma + 1},$$

(A.19)

$$u = u_0 \pm \frac{a_0}{\gamma}(\alpha - 1)\left(\frac{\dfrac{2\gamma}{\gamma+1}}{\alpha + \dfrac{\gamma-1}{\gamma+1}}\right)^{1/2},$$

(A.20)

$$A = A_0 - \frac{h(u - u_0)}{\sigma_{sh\pm}},$$

(A.21)

where

$$\sigma_{sh\pm} = \frac{(1-h)u_0}{A_0} \pm \frac{a_0}{A_0} \left(\frac{\gamma+1}{2\gamma}\alpha + \frac{\gamma-1}{2\gamma} \right)^{1/2}. \tag{A.22}$$

Formulas (A.19)$\sim$(A.22) hold for h, $0 \leqslant h \leqslant 1$.

Case 2: slip line (a). For the slip line corresponding to the slope $\sigma_2 = \frac{(1-h)u}{A}$, from Rankine-Hugoniot jump conditions (A.15)$\sim$(A.18), we find

$$u = u_0, \tag{A.23}$$

$$p = p_0, \tag{A.24}$$

$$A = A_0, \quad \text{if } h \neq 1. \tag{A.25}$$

The only variable which can change its value freely across the slip line is the density ρ.

Case 3: slip line (b). The other degenerate wave corresponds to the slope $\sigma_1 = 0$. From the Rankine-Hugoniot jump conditions, we have

$$u = u_0, \tag{A.26}$$

$$p = p_0, \tag{A.27}$$

$$\rho = \rho_0, \quad \text{if } h \neq 1. \tag{A.28}$$

The only variable which can change its value across this wave is A.

In the Lagrangian case when $h = 1$, $\sigma_1 = \sigma_2 = 0$, and the two slip lines coincide. In this case, the jump relations across the slip line are

$$u = u_0, \tag{A.29}$$

$$p = p_0, \tag{A.30}$$

but ρ and A can jump arbitrarily.

Appendix B

Computer Code for 1-D Flow in the Unified Coordinate

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C FIRST ORDER GODUNOV OR GODUNOV  SCHEME WITH MUSCL
C IN UNIFIED COORDINATES FOR ONE-DIMENSIONAL UNSTEADY FLOW.
C
C
C
C GOVERNING EQUATION
C
C                E_lamda + F_ksi = 0
C
C WHERE
C          /  rho*A  \      /   (1-h)*rho*u      \
C      E = | rho*u*A |   F = |  (1-h)*rho*u^2+p  |
C          | rho*e*A |       | (1-h)*rho*u*e+u*p |
C          \    A    /       \        -h*u       /
C HERE
C  rho : DENSITY
C  u   : VELOCITY COMPONENT IN EULERIAN COORDINATES
C  p   : PRESSURE
C  e   : INTERNAL ENERGY: 0.5*(u*u)+p/((gamma-1)*rho)
C  A   : GEOMETRICAL VARIABLE
C  h   : PARAMETER OF TRANSFORMATION  0 <= h <= 1
C
C REFERENCE:
C W. H. HUI and S. KOUDRIAKOV, THE ROLE OF COORDINATES IN THE
C         COMPUTATION OF DISCONTINUITIES IN ONE-DIMENSIONAL FLOW,
C         Computational Fluid Dynamics Journal, Vol 8, no 4,
C         January, 2000
C*********************************************************
c    PRINCIPAL VARIABLES:
C*********************************************************
C    D       : ARRAY OF DENSITIES
C    U       : ARRAY OF VELOCITIES
C    P       : ARRAY OF PRESSURES
C    C       : ARRAY OF SPEEDS OF SOUND
C    CU      : ARRAY OF GEOMETRIC VARIABLES 'A'
C    E1      : ARRAY OF CONSERVATIVE VARIABLES E1=rho*A
C    E2      : ARRAY OF CONSERVATIVE VARIABLES E2=rho*u*A
C    E3      : ARRAY OF CONSERVATIVE VARIABLES E3=rho*e*A
C    E4      : ARRAY OF CONSERVATIVE VARIABLES E4=A
C    F1,FF1  : ARRAY OF FLUXES FF1=F1=(1-h)*rho*u
C    F2,FF2  : ARRAY OF FLUXES FF2=F2=(1-h)*rho*u^2+p
C    F3,FF3  : ARRAY OF FLUXES FF3=F3=(1-h)*rho*u*e+u*p
C    F4,FF4  : ARRAY OF FLUXES FF4=F4=-h*u
C    XX      : ARRAY OF POSITIONS IN PHYSICAL SPACE
C    SPEED1  : ARRAYS OF OF SHOCK SPEEDS
```

```
C     SPEED2  :          OR FASTEST MACH LINES
C     TOL     : TOLERANCE ERROR
C     SEC     : PARAMETER FOR THE MUSCL UPGRADING
C                 SEC=0.0 --- FIRST-ORDER GODUNOV SCHEME
C                 SEC=0.5 --- GODUNOV SCHEME WITH MUSCL
C     DT      : TIME STEP
C     DX      : SIZE OF THE COMPUTATIONAL CELL
C     CFL     : CFL NUMBER
C     MD      : MAXIMUM DIMENSION OF ARRAYS
cCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      PARAMETER (MD=1500)
      DIMENSION D(0:MD),U(0:MD),P(0:MD),C(0:MD),CU(0:MD)
      DIMENSION E1(0:MD),E2(0:MD),E3(0:MD),E4(0:MD)
      DIMENSION F1(0:MD),F2(0:MD),F3(0:MD),F4(0:MD)
      DIMENSION FF1(0:MD),FF2(0:MD),FF3(0:MD),FF4(0:MD)
      DIMENSION XX(0:MD), SPEED1(0:MD), SPEED2(0:MD)
      COMMON /ARRAYS/  D,U,P,C,CU,E1,E2,E3,E4,F1,F2,F3,F4
      COMMON /ARRAYS/  FF1,FF2,FF3,FF4,XX
      COMMON /GAMMAS/  GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
      COMMON /SCALARS/ M,DX,DT,DOMLEN
      COMMON /STATES/ DLB,ULB,PLB,CLB,DRB,URB,PRB
      COMMON /STATES/ CRB,CULB,CURB
      COMMON /MSTATES/ DMB,UMB,PMB,CUMB
      COMMON /TSTATES/DL,UL,PL,DR,UR,PR,CUL,CUR,H
      EXTERNAL FUNCL,FUNCR,FMINMOD
      OPEN(UNIT=90,FILE='input.dat',STATUS='UNKNOWN')
      READ(90,*)XL         ! LEFT  END OF X-DOMAIN
      READ(90,*)XR         ! RIGHT END OF X-DOMAIN
      READ(90,*)M          ! NUMBER OF POINTS PROFILES
      READ(90,*)GAMMA      ! RATIO OF SPECIFIC HEATS
      READ(90,*)TIMEOUT    ! OUTPUT TIME
      READ(90,*)DLB        ! DENSITY ON LEFT SIDE
      READ(90,*)ULB        ! SPEED ON LEFT SIDE
      READ(90,*)PLB        ! PRESSURE ON LEFT SIDE
      READ(90,*)DRB        ! DENSITY ON RIGHT SIDE
      READ(90,*)URB        ! SPEED ON RIGHT SIDE
      READ(90,*)PRB        ! PRESSURE ON RIGHT SIDE
      READ(90,*)CULB       ! GEOMETRIC PARAMETER ON LEFT
      READ(90,*)CURB       ! GEOMETRIC PARAMETER ON RIGHT
      READ(90,*)H          ! PARAMETER IN TRANSFORMATION
      CLOSE(90)
C    compute gammas
      G1=(GAMMA-1.0)/(2.0*GAMMA)
      G2=(GAMMA+1.0)/(2.0*GAMMA)
      G3=2.0*GAMMA/(GAMMA-1.0)
      G4=2.0/(GAMMA-1.0)
      G5=2.0/(GAMMA+1.0)
      G6=(GAMMA-1.0)/(GAMMA+1.0)
      G7=0.5*(GAMMA-1.0)
      G8=1.0/GAMMA
      G9=GAMMA-1.0
C*********************************************
c     data for descrete calculations
C*********************************************
      CFL=0.8
      SEC = 0.0
      DOMLEN = XR-XL
      DX     = DOMLEN/REAL(M)
      TIME = 0.0
      CALL INICON1
0001  CONTINUE
C*****************************************
C     SET TRANSMISSIVE BOUNDARY CONDITIONS
```

```
C******************************************
           D(0)    = D(1)
           U(0)    = U(1)
           P(0)    = P(1)
           CU(0)   = CU(1)
           D(M+1) = D(M)
           U(M+1) = U(M)
           P(M+1) = P(M)
           CU(M+1)= CU(M)
C******************************************
C     CALCULATE FLUX ON THE INTERFACE
C******************************************
           DO 0002 I=0,M
            IF ((I .GE. 2).AND.(I .LE. (M-2))) THEN
              DTOP=D(I)-D(I-1)
              DBOT=D(I+1)-D(I)
              DL=D(I)+SEC*FMINMOD(DTOP,DBOT)
C******************************************
              UTOP=U(I)-U(I-1)
              UBOT=U(I+1)-U(I)
              UL=U(I)+SEC*FMINMOD(UTOP,UBOT)
C******************************************
              PTOP=P(I)-P(I-1)
              PBOT=P(I+1)-P(I)
              PL=P(I)+SEC*FMINMOD(PTOP,PBOT)
C******************************************
              CUTOP=CU(I)-CU(I-1)
              CUBOT=CU(I+1)-CU(I)
              CUL=CU(I)+SEC*FMINMOD(CUTOP,CUBOT)
C******************************************
              DTOP=D(I+2)-D(I+1)
              DBOT=D(I+1)-D(I)
              DR=D(I+1)-SEC*FMINMOD(DTOP,DBOT)
C******************************************
              UTOP=U(I+2)-U(I+1)
              UBOT=U(I+1)-U(I)
              UR=U(I+1)-SEC*FMINMOD(UTOP,UBOT)
C******************************************
              PTOP=P(I+2)-P(I+1)
              PBOT=P(I+1)-P(I)
              PR=P(I+1)-SEC*FMINMOD(PTOP,PBOT)
C******************************************
              CUTOP=CU(I+2)-CU(I+1)
              CUBOT=CU(I+1)-CU(I)
              CUR=CU(I+1)-SEC*FMINMOD(CUTOP,CUBOT)
            ELSE
              DL=D(I)
              UL=U(I)
              PL=P(I)
              CUL=CU(I)
              DR=D(I+1)
              UR=U(I+1)
              PR=P(I+1)
              CUR=CU(I+1)
            ENDIF
            CALL RIEMANN(PM,UM)
C*********************************************
              CL=SQRT(GAMMA*PL/DL)
              CR=SQRT(GAMMA*PR/DR)
              IF(PM.GT.PL) THEN
C    left wave is shock
                  SL=(1.0-H)*UL/CUL-CL*SQRT(G2*PM/PL+G1)/CUL
                  CUML=CUL-H*(UM-UL)/SL
```

```
                 SPEED2(I) = SL
              ELSE
C    left wave is expansion wave
                 A = PL
                 B = PM
                 CALL QK15(FUNCL,A,B,RESULT,ABSERR,RESABS,RESASC)
                 CUML=CUL*EXP(RESULT)
                 SPEED2(I) = ((1.0-H)*UL-CL)/CUL
                 IF(ABS(PM-PL) .LE. 0.000001) SPEED2(I)=0.0
              ENDIF
C***************************************************
              IF(PM.GT.PR) THEN
C    right wave is shock
                 SR=(1.0-H)*UR/CUR+CR*SQRT(G2*PM/PR+G1)/CUR
                 CUMR=CUR-H*(UM-UR)/SR
                 SPEED1(I+1) = SR
              ELSE
C    right wave is expansion wave
                 A = PR
                 B = PM
                 CALL QK15(FUNCR,A,B,RESULT,ABSERR,RESABS,RESASC)
                 CUMR=CUR*EXP(RESULT)
                 SPEED1(I+1) = ((1.0-H)*UR+CR)/CUR
                 IF(ABS(PM-PR) .LE. 0.000001) SPEED1(I+1)=0.0
              ENDIF
C*****************************************************
              S=0.0
              CALL SAMPLE(PM,UM,CUML,CUMR,S,PS,US,DS)
C*****************************************************
              F1(I) = (1.0-H)*DS*US
              F2(I) = F1(I)*US+PS
              F3(I) = F1(I)*(US*US/2.0+PS/(G9*DS))+PS*US
              F4(I) =-H*US
0002      CONTINUE
C***********************************************************
C   CHOOSING TIME STEP
C***********************************************************
          SMAX = 0.0
          DO 777 I=1,M
           IF(SMAX .LE. ABS(SPEED1(I))) SMAX=ABS(SPEED1(I))
777       CONTINUE
          DO 778 I=1,M
           IF(SMAX .LE. ABS(SPEED2(I))) SMAX=ABS(SPEED2(I))
778       CONTINUE
          IF(SMAX.GT.0.0) THEN
            DT=CFL*DX/SMAX
          ELSE
            DT=CFL*0.1*DX
          ENDIF
          IF(TIME.LE.TIMEOUT.AND.TIME+DT.GT.TIMEOUT)THEN
            DT=TIMEOUT-TIME
            TIME=TIMEOUT
          ELSE
            TIME=TIME+DT
          ENDIF
          print*,"dt=,time=",DT,TIME
          IF(ABS(TIME-TIMEOUT).LT.0.00001) GO TO 98
C***********************************************************
C   UPDATING CONSERVATIVE VARIABLES
C***********************************************************
          DO 0003 I=1,M
            E1(I)=E1(I)-(DT/DX)*(F1(I)-F1(I-1))
            E2(I)=E2(I)-(DT/DX)*(F2(I)-F2(I-1))
```

```
                E3(I)=E3(I)-(DT/DX)*(F3(I)-F3(I-1))
                E4(I)=E4(I)-(DT/DX)*(F4(I)-F4(I-1))
0003      CONTINUE
C***********************************************************
C   DECODING TO GET PHYSICAL VARIABLES
C***********************************************************
          DO 0004 I=1,M
              UOLD = U(I)
              D(I) = E1(I)/E4(I)
              U(I) = E2(I)/E1(I)
              CU(I)= E4(I)
              TEMP = E3(I)/E1(I)-U(I)*U(I)/2.0
              P(I) = TEMP*G9*D(I)
              XX(I)=XX(I)+0.5*DT*H*(U(I)+UOLD)
0004      CONTINUE
      GO TO 0001
C***********************************************************
C   PRINT RESULTS INTO THE FILE 'results.dat'
C***********************************************************
98        OPEN(UNIT=7,FILE='results.dat',STATUS='UNKNOWN')
          DO 0005 I=1,M
              WRITE(7,6) I*DX,XX(I),D(I),U(I),P(I),
     1                  D(I)*LOG(P(I)/(D(I)**GAMMA))
0006          FORMAT(F10.6,2X,5(F12.5,1X))
0005      CONTINUE
      END


C***********************************************************
C***********************************************************
      SUBROUTINE RIEMANN(P,U)
C*************************************************
C     compute pressure P and velocity U
C     on the slip line (contact line)
C     Newton-Raphson method is used to find P
C*************************************************
      COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
      COMMON /TSTATES/DL,UL,PL,DR,UR,PR,CUL,CUR,H
      DATA TOL,NRITER/1.0E-10,50/
      DU=UR-UL
C   compute sound speeds
      CL=SQRT(GAMMA*PL/DL)
      CR=SQRT(GAMMA*PR/DR)
C   compute critical velocity
      DUCRIT=G4*(CL+CR)-DU
C   check for generation of vacuum
      IF(DUCRIT.LE.0.0) THEN
C   vacuum is generated by given data
          WRITE(6,*)'VACUUM IS GENERATED BY GIVEN DATA'
          STOP
      ENDIF
C   guess value is computed
      CALL STARTE(P,DL,UL,PL,CL,DR,UR,PR,CR)
      P0=P
      DO 0001 K=1,NRITER
          CALL PREFUN(FL,FLD,P,DL,PL,CL)
          CALL PREFUN(FR,FRD,P,DR,PR,CR)
          P   = P -(FL+FR+DU)/(FLD+FRD)
          CHA = 2.0*ABS((P-P0)/(P+P0))
          IF(CHA.LE.TOL)GOTO 0002
          IF(P.LT.0.0)P=TOL
          P0=P
0001      CONTINUE
      WRITE(6,*)'DIVERGENCE IN NEWTON-RAPHSON SCHEME'
```

```
0002    CONTINUE
C    compute U
        U = 0.5*(UL+UR+FR-FL)
        RETURN
        END
C*****************************************************
C      choose initial P for iteration
C      Reference: E.Toro "Rieman Solvers and
C                 Numerical methods for Fluid Dynamics"
C*****************************************************
        SUBROUTINE STARTE(P,DL,UL,PL,CL,DR,UR,PR,CR)
C    hybrid starter using PVRS, TRRS and TSRS
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        DATA TOL,QMAX/1.0E-10,2.0/
C    compute guess value from PVRS riemann solver
        PV   = 0.5*(PL+PR)-0.125*(UR-UL)*(DL+DR)*(CL+CR)
        PMIN = AMIN1(PL,PR)
        PMAX = AMAX1(PL,PR)
        QRAT = PMAX/PMIN
        IF(QRAT.LE.QMAX.AND.(PMIN.LE.PV.AND.PV.LE.PMAX)) THEN
C    use PVRS solution as guess
           P=AMAX1(TOL,PV)
        ELSE
           IF(PV.LT.PMIN) THEN
C    use two-rarefaction solution
              PNU = CL+CR-G7*(UR-UL)
              PDE = CL/PL**G1 + CR/PR**G1
              P   = (PNU/PDE)**G3
           ELSE
C    two-shock approximation with PV as estimate
              GEL = SQRT((G5/DL)/(G6*PL+AMAX1(TOL,PV)))
              GER = SQRT((G5/DR)/(G6*PR+AMAX1(TOL,PV)))
              P   = (GEL*PL+GER*PR-(UR-UL))/(GEL+GER)
              P   = AMAX1(TOL,P)
           ENDIF
        ENDIF
        RETURN
        END
C***********************************************
C      function for iteration
C***********************************************
        SUBROUTINE PREFUN(F,FD,P,DK,PK,CK)
C    pressure functions are evaluated
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        IF(P.LE.PK) THEN
C    rarefaction wave
           PRAT = P/PK
           F    = G4*CK*(PRAT**G1 -1.0)
           FD   = (1.0/(DK*CK))*PRAT**(-G2)
        ELSE
C    shock wave
           AK   = G5/DK
           BK   = G6*PK
           QRT  = SQRT(AK/(BK+P))
           F    = (P-PK)*QRT
           FD   = (1.0 - 0.5*(P-PK)/(BK+P))*QRT
        ENDIF
        RETURN
        END
C*****************************************************************
C*****************************************************************
      SUBROUTINE QK15(F,A,B,RESULT,ABSERR,RESABS,RESASC)
C***BEGIN PROLOGUE  QK15
```

```
      C***DATE WRITTEN   800101   (YYMMDD)
      C***REVISION DATE  870530   (YYMMDD)
      C***CATEGORY NO.  H2A1A2
      C***KEYWORDS  15-POINT GAUSS-KRONROD RULES
      C***AUTHOR  PIESSENS, ROBERT, AND DE DONCKER, ELISE,
      C             APPLIED MATH. AND PROGR. DIV. - K. U. LEUVEN
      C***PURPOSE  To compute I = Integral of F over (A,B), with error estimate
      C             and     J = integral of ABS(F) over (A,B)
      C***DESCRIPTION
      C
      C          From the book "Numerical Methods and Software"
      C                 by  D. Kahaner, C. Moler, S. Nash
      C                     Prentice Hall 1988
      C
      C          Real version
      C
      C          PARAMETERS ON ENTRY
      C             F      - Real
      C                      Function subprogram defining the integrand
      C                      FUNCTION F(X). The actual name for F needs to be
      C                      Declared E X T E R N A L in the calling program.
      C
      C             A      - Real: Lower limit of integration
      C
      C             B      - Real: Upper limit of integration
      C
      C           PARAMETERS ON RETURN
      C             RESULT - Real: Approximation to the integral I
      C                      Result is computed by applying the 15-POINT
      C                      KRONROD RULE (RESK) obtained by optimal addition
      C                      of abscissae to the7-POINT GAUSS RULE(RESG).
      C
      C             ABSERR - Real: Estimate of the modulus of the absolute error,
      C                      which should not exceed ABS(I-RESULT)
      C
      C             RESABS - Real: Approximation to the integral J
      C
      C             RESASC - Real: Approximation to the integral of ABS(F-I/(B-A))
      C                        over (A,B)
      C***REFERENCES   PIESSENS R. ET. AL., "QUADPACK: A SUBROUTINE PACKAGE FOR
      C               AUTOMATIC INTEGRATION"  SPRINGER, BERLIN 1983.
      C***ROUTINES CALLED  R1MACH
      C***END PROLOGUE  QK15
      C
            REAL A,ABSC,ABSERR,B,CENTR,DHLGTH,EPMACH,F,FC,FSUM,FVAL1,FVAL2,
           1  FV1,FV2,HLGTH,RESABS,RESASC,RESG,RESK,RESKH,RESULT,UFLOW,
           2  WG,WGK,XGK
            INTEGER J,JTW,JTWM1
            EXTERNAL F
      C
            DIMENSION FV1(7),FV2(7),WG(4),WGK(8),XGK(8)
      C
      C          THE ABSCISSAE AND WEIGHTS ARE GIVEN FOR THE INTERVAL (-1,1).
      C          BECAUSE OF SYMMETRY ONLY THE POSITIVE ABSCISSAE AND THEIR
      C          CORRESPONDING WEIGHTS ARE GIVEN.
      C
      C          XGK    - ABSCISSAE OF THE 15-POINT KRONROD RULE
      C                   XGK(2), XGK(4), ...  ABSCISSAE OF THE 7-POINT
      C                   GAUSS RULE
      C                   XGK(1), XGK(3), ...  ABSCISSAE WHICH ARE OPTIMALLY
      C                   ADDED TO THE 7-POINT GAUSS RULE
      C
      C          WGK    - WEIGHTS OF THE 15-POINT KRONROD RULE
```

```
C
C             WG      - WEIGHTS OF THE 7-POINT GAUSS RULE
C
      DATA XGK(1),XGK(2),XGK(3),XGK(4),XGK(5),XGK(6),XGK(7),XGK(8)/
     1      0.9914553711208126D+00,   0.9491079123427585D+00,
     2      0.8648644233597691D+00,   0.7415311855993944D+00,
     3      0.5860872354676911D+00,   0.4058451513773972D+00,
     4      0.2077849550078985D+00,   0.0D+00                   /
      DATA WGK(1),WGK(2),WGK(3),WGK(4),WGK(5),WGK(6),WGK(7),WGK(8)/
     1      0.2293532201052922D-01,   0.6309209262997855D-01,
     2      0.1047900103222502D+00,   0.1406532597155259D+00,
     3      0.1690047266392679D+00,   0.1903505780647854D+00,
     4      0.2044329400752989D+00,   0.2094821410847278D+00/
      DATA WG(1),WG(2),WG(3),WG(4)/
     1      0.1294849661688697D+00,   0.2797053914892767D+00,
     2      0.3818300505051189D+00,   0.4179591836734694D+00/
C
C
C             LIST OF MAJOR VARIABLES
C             -----------------------
C
C             CENTR  - MID POINT OF THE INTERVAL
C             HLGTH  - HALF-LENGTH OF THE INTERVAL
C             ABSC   - ABSCISSA
C             FVAL*  - FUNCTION VALUE
C             RESG   - RESULT OF THE 7-POINT GAUSS FORMULA
C             RESK   - RESULT OF THE 15-POINT KRONROD FORMULA
C             RESKH  - APPROXIMATION TO THE MEAN VALUE OF F OVER (A,B),
C                      I.E. TO I/(B-A)
C
C             MACHINE DEPENDENT CONSTANTS
C             ---------------------------
C
C             EPMACH IS THE LARGEST RELATIVE SPACING.
C             UFLOW IS THE SMALLEST POSITIVE MAGNITUDE.
C
C***FIRST EXECUTABLE STATEMENT  QK15
c     EPMACH = R1MACH(4)
c     UFLOW = R1MACH(1)
C
c
      EPMACH = 0.6D-07
      UFLOW = 0.12D-37

      CENTR = 0.5E+00*(A+B)
      HLGTH = 0.5E+00*(B-A)
      DHLGTH = ABS(HLGTH)
C
C             COMPUTE THE 15-POINT KRONROD APPROXIMATION TO
C             THE INTEGRAL, AND ESTIMATE THE ABSOLUTE ERROR.
C
      FC = F(CENTR)
      RESG = FC*WG(4)
      RESK = FC*WGK(8)
      RESABS = ABS(RESK)
      DO 10 J=1,3
        JTW = J*2
        ABSC = HLGTH*XGK(JTW)
        FVAL1 = F(CENTR-ABSC)
        FVAL2 = F(CENTR+ABSC)
        FV1(JTW) = FVAL1
        FV2(JTW) = FVAL2
        FSUM = FVAL1+FVAL2
```

```
           RESG = RESG+WG(J)*FSUM
           RESK = RESK+WGK(JTW)*FSUM
           RESABS = RESABS+WGK(JTW)*(ABS(FVAL1)+ABS(FVAL2))
    10 CONTINUE
        DO 15 J = 1,4
           JTWM1 = J*2-1
           ABSC = HLGTH*XGK(JTWM1)
           FVAL1 = F(CENTR-ABSC)
           FVAL2 = F(CENTR+ABSC)
           FV1(JTWM1) = FVAL1
           FV2(JTWM1) = FVAL2
           FSUM = FVAL1+FVAL2
           RESK = RESK+WGK(JTWM1)*FSUM
           RESABS = RESABS+WGK(JTWM1)*(ABS(FVAL1)+ABS(FVAL2))
    15 CONTINUE
        RESKH = RESK*0.5E+00
        RESASC = WGK(8)*ABS(FC-RESKH)
        DO 20 J=1,7
           RESASC = RESASC+WGK(J)*(ABS(FV1(J)-RESKH)+ABS(FV2(J)-RESKH))
    20 CONTINUE
        RESULT = RESK*HLGTH
        RESABS = RESABS*DHLGTH
        RESASC = RESASC*DHLGTH
        ABSERR = ABS((RESK-RESG)*HLGTH)
        IF(RESASC.NE.0.0E+00.AND.ABSERR.NE.0.0E+00)
     1  ABSERR = RESASC*AMIN1(0.1E+01,
     2  (0.2E+03*ABSERR/RESASC)**1.5E+00)
        IF(RESABS.GT.UFLOW/(0.5E+02*EPMACH)) ABSERR = AMAX1
     1  ((EPMACH*0.5E+02)*RESABS,ABSERR)
        RETURN
        END

C************************************************
C   Functions for finding pressure
c    inside left and right expansion fan
C************************************************
        REAL FUNCTION FUNCL(X)
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        COMMON /TSTATES/ DL,UL,PL,DR,UR,PR,CUL,CUR,H
        REAL X
           CL=SQRT(GAMMA*PL/DL)
           AL=(1.0-H)*(UL+2.0*CL/G9)
           AL=AL*SQRT(GAMMA*DL/(PL**G8))
           BL=G3*(1.0-H)+GAMMA
           FUNCL = H/(AL*X**G2-BL*X)
        RETURN
        END
C****************************************************
        REAL FUNCTION FUNCR(X)
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        COMMON /TSTATES/ DL,UL,PL,DR,UR,PR,CUL,CUR,H
        REAL X
           CR=SQRT(GAMMA*PR/DR)
           AR=(1.0-H)*(UR-2.0*CR/G9)
           AR=AR*SQRT(GAMMA*DR/(PR**G8))
           BR=G3*(1.0-H)+GAMMA
           FUNCR = -H/(AR*X**G2+BR*X)
        RETURN
        END
C*******************************************************
c       function limiter for second-order terms
C*******************************************************
        REAL FUNCTION FMINMOD(X,Y)
```

```
              PROD=X*Y
              IF (PROD .LE. 0.0) THEN
                FMINMOD=0.0
              ELSE
                AX=ABS(X)
                AY=ABS(Y)
              IF (AX .LE. AY) THEN
               FMINMOD=X
              ELSE
               FMINMOD=Y
              ENDIF
              ENDIF
              RETURN
              END
C******************************************************
c
C************************************************
C       sampling the solution
C       This subroutine gives exact solution
C       as a function of angle (S)
C************************************************
        SUBROUTINE SAMPLE(PM,UM,CUML,CUMR,S,P,U,D)
C     solution is sampled according to wave patterns
c     values  in: PM, UM, S
c     values out: P, U, D
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        COMMON /TSTATES/ DL,UL,PL,DR,UR,PR,CUL,CUR,H
        EXTERNAL FUNCL,FUNCR
        CL=SQRT(GAMMA*PL/DL)
        CR=SQRT(GAMMA*PR/DR)
C******************************************************
        IF(((1.0-H)*UM).GE.0.0) THEN
             CUM=CUMR
        ELSE
             CUM=CUML
        ENDIF
C******************************************************
        IF(S.LE.((1.0-H)*UM/CUM)) THEN
C     sample point is to the left of the contact
          IF(PM.LE.PL) THEN
C     left fan
              SHL=((1.0-H)*UL-CL)/CUL
              IF(S.LE.SHL) THEN
C     left data state
                D=DL
                U=UL
                P=PL
              ELSE
                CML= CL*(PM/PL)**G1
                STL= ((1.0-H)*UM-CML)/CUML
                IF(S.GT.STL) THEN
C     middle left state
                  D=DL*(PM/PL)**G8
                  U=UM
                  P=PM
                ELSE
C     an left state (inside fan)
                  TOL=1.0E-06
                  NRITER=20
                  P00=(PM+PL)/2.0
                  P=P00
                  DO 0007 K=1,NRITER
                     CALL FUNL(FFL,FDL,S,P)
```

```
                               P = P - FFL/FDL
                               CHA = 2.0*ABS((P-P00)/(P+P00))
                               IF(CHA.LE.TOL)GOTO 0009
                               IF(P.LT.0.0)P=TOL
                               P00=P
0007                   CONTINUE
                       WRITE(6,*)'DIVERGENCE IN NEWTON-FUN SCHEME'
0009                   CONTINUE
                       A=PL
                       B=P
                       CALL QK15(FUNCL,A,B,RESULT,ABSERR,RESABS,RESASC)
                       CU=CUL*EXP(RESULT)
                       U = 2.0*(CL+G7*UL+CU*S)/(2.0*(1.0-H)+G9)
                       D = DL*(P/PL)**G8
                   ENDIF
               ENDIF
           ELSE
C     left shock
               PML = PM/PL
               SL  = (1.0-H)*UL/CUL-CL*SQRT(G2*PM/PL+G1)/CUL
               IF(S.LE.SL) THEN
C     left data state
                   D = DL
                   U = UL
                   P = PL
               ELSE
C     middle left state (behind shock)
                   D = DL*(PML+G6)/(PML*G6+1.0)
                   U = UM
                   P = PM
               ENDIF
           ENDIF
        ELSE
C     right of contact
           IF(PM.GT.PR) THEN
C     right shock
               PMR = PM/PR
               SR  = (1.0-H)*UR/CUR+CR*SQRT(G2*PM/PR+G1)/CUR
               IF(S.GE.SR) THEN
C     right data state
                   D = DR
                   U = UR
                   P = PR
               ELSE
C     middle right state (behind shock)
                   D = DR*(PMR+G6)/(PMR*G6+1.0)
                   U = UM
                   P = PM
               ENDIF
           ELSE
C     right fan
               SHR=((1.0-H)*UR+CR)/CUR
               IF(S.GE.SHR) THEN
C     right data state
                   D=DR
                   U=UR
                   P=PR
               ELSE
                   CMR= CR*(PM/PR)**G1
                   STR= ((1.0-H)*UM+CMR)/CUMR
                   IF(S.LE.STR) THEN
C     middle right state
                       D=DR*(PM/PR)**G8
```

```
                        U=UM
                        P=PM
                    ELSE
C     fan right state (inside fan)
                        TOL=1.0E-06
                        NRITER=20
                        P00=(PM+PR)/2.0
                        P=P00
                        DO 0006 K=1,NRITER
                           CALL FUNR(FFR,FDR,S,P)
                           P = P - FFR/FDR
                           CHA = 2.0*ABS((P-P00)/(P+P00))
                           IF(CHA.LE.TOL)GOTO 0008
c                           print*,"pr,ffr,fdr",P,FFR,FDR
                           IF(P.LT.0.0)P=TOL
                           P00=P
0006                    CONTINUE
                        WRITE(6,*)'DIVERGENCE IN NEWTON-FUN SCHEME'
0008                    CONTINUE
                        A=PR
                        B=P
                        CALL QK15(FUNCR,A,B,RESULT,ABSERR,RESABS,RESASC)
                        CU=CUR*EXP(RESULT)
                        U = 2.0*(-CR+G7*UR+CU*S)/(2.0*(1-H)+G9)
                        D = DR*(P/PR)**G8
                    ENDIF
                 ENDIF
              ENDIF
           ENDIF
        RETURN
        END
C*********************************************
C    solution inside the fan
C*********************************************
        SUBROUTINE FUNR(FFR,FDR,S,P)
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        COMMON /TSTATES/ DL,UL,PL,DR,UR,PR,CUL,CUR,H
        EXTERNAL FUNCR
          CL=SQRT(GAMMA*PL/DL)
          CR=SQRT(GAMMA*PR/DR)
          AA=(2.0*(1.0-H)+G9)/(PR**G1)
          BB=G9*(1.0-H)*UR/CR-2.0*(1.0-H)
          CC=G9*CUR*S/CR
          A=PR
          B=P
          CALL QK15(FUNCR,A,B,RESULT,ABSERR,RESABS,RESASC)
          FFR=AA*(P**G1)+BB-CC*EXP(RESULT)
          FDR=G1*AA*(P**(-G2))-CC*EXP(RESULT)*FUNCR(P)
        RETURN
        END
C*********************************************************
        SUBROUTINE FUNL(FFL,FDL,S,P)
        COMMON /GAMMAS/ GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
        COMMON /TSTATES/ DL,UL,PL,DR,UR,PR,CUL,CUR,H
        EXTERNAL FUNCL
          CL=SQRT(GAMMA*PL/DL)
          CR=SQRT(GAMMA*PR/DR)
          AA=(2.0*(1.0-H)+G9)/(PL**G1)
          BB=G9*(1.0-H)*UL/CL+2.0*(1.0-H)
          CC=G9*CUL*S/CL
          A=PL
          B=P
          CALL QK15(FUNCL,A,B,RESULT,ABSERR,RESABS,RESASC)
```

```
               FFL=AA*(P**G1)-BB+CC*EXP(RESULT)
               FDL=G1*AA*(P**(-G2))+CC*EXP(RESULT)*FUNCL(P)
            RETURN
            END

C*********************************************
C    initila data for the Riemann problem
C*********************************************
         SUBROUTINE INICON1
         PARAMETER (MD=1500)
         DIMENSION D(0:MD),U(0:MD),P(0:MD),C(0:MD),CU(0:MD)
         DIMENSION E1(0:MD),E2(0:MD),E3(0:MD),E4(0:MD)
         DIMENSION F1(0:MD),F2(0:MD),F3(0:MD),F4(0:MD)
         DIMENSION FF1(0:MD),FF2(0:MD),FF3(0:MD),FF4(0:MD)
         DIMENSION XX(0:MD), SPEED1(0:MD), SPEED2(0:MD)
         COMMON /ARRAYS/  D,U,P,C,CU,E1,E2,E3,E4,F1,F2,F3,F4
         COMMON /ARRAYS/  FF1,FF2,FF3,FF4,XX
         COMMON /GAMMAS/  GAMMA,G1,G2,G3,G4,G5,G6,G7,G8,G9
         COMMON /SCALARS/ M,DX,DT,DOMLEN
         COMMON /STATES/ DLB,ULB,PLB,CLB,DRB,URB,PRB
         COMMON /STATES/ CRB,CULB,CURB
         COMMON /MSTATES/ DMB,UMB,PMB,CUMB
         COMMON /TSTATES/DL,UL,PL,DR,UR,PR,CUL,CUR,H
C*****************************************************
C     INITIALISATION OF PHYSICAL VARIABLES
C*****************************************************
         XX(0) = 0.0
         DO 0001 I=1,M
            XX(I) = I*DX
            X = I*DX
            IF(X.LE.(DOMLEN/2.0)) THEN
               D(I) = DLB
               U(I) = ULB
               P(I) = PLB
               C(I) = SQRT(GAMMA*PLB/DLB)
               CU(I)= CULB
            ELSE
               D(I) = DRB
               U(I) = URB
               P(I) = PRB
               C(I) = SQRT(GAMMA*PRB/DRB)
               CU(I)= CURB
            ENDIF
C*****************************************************
C     INITIALISATION OF CONSERVATIVE VARIABLES
C*****************************************************
            E1(I) = D(I)*CU(I)
            E2(I) = E1(I)*U(I)
            E3(I) = E1(I)*(U(I)*U(I)/2.0+P(I)/(G9*D(I)))
            E4(I) = CU(I)
0001     CONTINUE
         RETURN
         END
```