12-15-2012

Simulation of Multispecies Gas Flows using the Discontinuous Galerkin Method

Lei Liang

Simulation of multispecies gas flows using the Discontinuous Galerkin method

By

Lei Liang

A Dissertation
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in Engineering
in the Department of Aerospace Engineering

Mississippi State, Mississippi

December 2012

Simulation of multispecies gas flows using the

Discontinuous Galerkin method

By

Lei Liang

Approved:

Pasquale Cinnella Professor of Aerospace Engineering (Major Professor)	Edward Allen Luke Associate Professor of Computer Science and Engineering (Director of Dissertation)
J. Mark Janus Associate Professor and Graduate Coordinator of Aerospace Engineering (Committee Member)	David S. Thompson Associate Professor of Aerospace Engineering (Committee Member)
Xiaoling Tong Assistant Research Professor of Computational Engineering (Committee Member)	Sarah A. Rajala Dean of the James Worth Bagley College of Engineering

Name: Lei Liang

Date of Degree: December 15, 2012

Institution: Mississippi State University

Major Field: Engineering

Major Professor: Dr. Pasquale Cinnella

Director of Dissertation: Dr. Edward Allen Luke

Title of Study: Simulation of multispecies gas flows using the Discontinuous Galerkin method

Pages of Study: 169

Candidate for Degree of Doctor of Philosophy

Truncation errors and computational cost are obstacles that still hinder large-scale applications of the Computational Fluid Dynamics method. The discontinuous Galerkin method is one of the high-order schemes utilized extensively in recent years, which is locally conservative, stable, and high-order accurate. Besides that, it can handle complex geometries and irregular meshes with hanging nodes.

In this document, the nondimensional compressible Euler equations and Reynolds-Averaged Navier-Stokes equations are discretized by discontinuous Galerkin methods with a two-equations turbulence model on both structured and unstructured meshes. The traditional equation of state for an ideal gas model is substituted by a multispecies thermodynamics model in order to complete the governing equations. An approximate Riemann solver is used for computing the convective flux, and the diffusive flux is approximated with some internal penalty based schemes. The temporal discretization of the partial dif-

ferential equations is either performed explicitly with the aid of Rung-Kutta methods or with semi-implicit methods. Inspired by the artificial viscosity diffusion based limiter for shock-capturing method, which has been extensively studied, a novel and robust technique based on the introduction of mass diffusion to the species governing equations to guarantee that the species mass fractions remain positive has been thoroughly investigated. This contact-surface-capturing method is conservative and a high order of accuracy can be maintained for the discontinuous Galerkin method. For each time step of the algorithm, any trouble cell is first caught by the contact-surface discontinuity detector. Then some amount of mass diffusions are added to the governing equations to change the gas mixtures and arrive at an equilibrium point satisfying some conditions. The species properties are reasonable without any oscillations.

Computations are performed for many steady and unsteady flow problems. For general non-mixing fluid flows, the classical air-helium shock bubble interaction problem is the central test case for the high-order discontinuous Galerkin method with a mass diffusion based limiter chosen. The computed results are compared with experimental, exact, and empirical data to validate the fluid dynamic solver.

Key words: Computational Fluid Dynamics, Discontinuous Galerkin, diffusion based, high order, shock capturing, basis function, unstructured

DEDICATION

I dedicate this dissertation to my Dad and Mom for encouraging and supporting me to reach my dreams.

ACKNOWLEDGEMENTS

I would like to thank my major professor, Pasquale Cinnella, for providing me the opportunity of studying at MSU. Thanks for bringing me into the wonderful field of gas physics and aerodynamics and guiding me in finishing the academic requirements of my Doctoral degree.

I would especially like to express my gratitude to my dissertation director Edward Allen Luke for supporting me while doing the research. Many thanks for his suggestions, guidance, and dedication. Thank you for offering me this interesting and challenging research topic, providing me a prospective outlook and teaching me to be a good researcher.

Thanks to Eric M. Collins for his patience and hospitality in teaching me the basics of discontinuous Galerkin method and solver at the beginning of my research, and helping me in solving issues with the current solver. Usually discussions and conversations with him gave me new ideas and contributions to my research work. Special thanks to Xiaoling Tong, Yang Zhang, and Qiuhan Xue for providing technical support in fluid physics, solver specific and algorithm specific, and their encouragement in doing the research. I feel so lucky to be one member of this active and strong academic group. Thanks to the Simulation and Design Center at the High Performance Computing Collaboratory at Mississippi State University for providing the hardware facility for operating and managing the solver on large-scale clusters.

I would like to give special thanks to lots of friends in the city of Starkville where we shared many joyful times and played together. All of these will remain important memories for the rest of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

Greek

$\tilde{\alpha}_i$ Wave length

$\tilde{\gamma}, \gamma$ Isentropic index

Γ_h An approximation of domain boundary

δ_{ij} Kronecker delta

ε_e A piecewise smooth function

η_1, η_2, η_3 Computational space coordinates, face local coordinates

κ Turbulent kinetic energy

λ Thermal conductivity coefficient

λ_t Turbulent thermal conductivity

$\tilde{\lambda}_i$ Eigenvalues

μ Molecular viscosity coefficient

μ_t Eddy viscosity, turbulent viscosity

ξ_1, ξ_2, ξ_3 Local collapsed coordinates, cell local coordinates

ρ Fluid density

ρ_1, ρ_2 Fluid densities in a mixture

ρ_i/ρ Mass fraction of each species

Σ_h An approximation of element interface

$\mathrm{d}\sigma$ Surface element operator in integration

$\hat{\tau}$ Turbulent shear stress tensor

τ General viscous stress tensor, laminar shear stress tensor

ψ Weighting function

ω Turbulent dissipation rate

ω_q Weight for each quadrature point

Ω Entire domain

Ω_h An approximation of entire domain Ω

$\partial\Omega$ Boundary of the entire domain

$\mathrm{d}\Omega$ Volume element operator in integration

Latin

$\mathscr{A}$ Nonlinear function operator of conservative variables U

b Domain element boundary after discretization

$\mathscr{B}_h$ Boundary of discretized domain

C Coefficient of shock length scale

$\tilde{C}_{v,N2}, \tilde{C}_{v,O2}$ Specific heat at constant volume for species equilibrium portion of energy

$\tilde{C}_{p,N2}, \tilde{C}_{p,O2}$ Specific heat at constant pressure for species equilibrium portion of energy

$\tilde{C}_v, C_v$ Mixture specific heat at constant volume

$\tilde{C}_p, C_p$ Mixture specific heat at constant pressure

$D_{1,2}, D_{2,1}$ Species diffusion coefficients

e_0 Total energy

e_{N2}, e_{O2} Internal energy per unit mass of each species

e The internal energy per unit mass of a mixture, element after discretization

$e_{ref,N2}, e_{ref,O2}$ Reference value for internal energy

$\tilde{e}_{N2}, \tilde{e}_{O2}$ Equilibrium portion of the internal energy of each species

$\mathbf{F}_j, \mathbf{F}$ Inviscid flux tensor

$\mathbf{g}$ Arbitrary weighting vector function

$\mathbf{G}_j$ Diffusive flux tensor

$\mathbf{G}$ Arbitrary test vector function space

h A measure of length scale based on the grid size

h_0 Total enthalpy

i Element interface after discretization

$\mathscr{I}_h$ Interface of discretized domain

J Jacobian matrix

$M_{ij}, \mathbf{M}$ Element mass matrix

$M_i, \hat{M}_i$ Molar mass of species i, molecular weight

nq Number of quadrature points in integration

ndf Polynomial number of degree of freedom

n_i Molar concentration of species i

Ns Number of species in a mixture

OP Solution order of accuracy

p Polynomial degree, fluid pressure

$P^k(e)$ Space of polynomial functions of degree at most k

q_j Heat flux vector

$\tilde{\mathbf{r}}^{(i)}$ Right eigenvectors

$\tilde{R}, R$ Mixture gas constant

$\hat{R}$ Universal gas constant

$\mathbf{R}(U)$ Global residual vector

S_e Smoothness indicator

S_L, S_R Fastest signal velocities perturbing the initial data state U_L, U_R

$\mathbf{S}$ Source vector

S_{ij} Mean strain-rate tensor

t Time

T Translational temperature

T_{ref} Reference value of temperature

$\mathscr{T}_h$ Elements of discretized domain

u_i Cartesian component of fluid velocity in index notation

U Conservative variable vector, polynomial coefficients

V Volume, arbitrary test function space

V^p An approximation to the solution vector U^p in p-multigrid method

$\mathbf{V}_s$ Species diffusion velocity

x_1, x_2, x_3 Global Cartesian coordinates

X_i Mole fraction of the species component

y_i Mole fraction of species i

Y_s, Y_i Species mass fraction

Operators

$\mathscr{J}$, $\mathscr{J}^0$ Vector jump operator

$\mathbf{r}_h^\sigma$ Local lift operator in BR scheme

$\mathbf{R}_h$ Global lift operator in BR scheme

$\langle(\cdot)\rangle$ Arithmetic average operator

$[(\cdot)], \| \cdot \|$ Jump operator

$\tilde{(\cdot)}$ Averaged value, Roe averaged value

$\{(\cdot)\}, \{ \cdot \}$ Average operator, mean opeartor

$(\cdot, \cdot)$ Stand inner product in $L_2(\Omega_e)$

Dimensionless parameters

CFL Courant Friedrichs Lewy number

Ma Mach number

Pr Prandtl number

Pr_t Turbulent Prandtl number

Re Reynolds number

Superscripts and Subscripts

^{b} Boundary value

$_{l}$ Left state value

$_{r}$ Right state value

$^{-}$ Nondimensionalized variables

$^{+}$ Interface value taken from the exterior

$^{-}$ Interface value taken from the interior

$\tilde{(\cdot)}$ Equilibrium portion of energy

$\hat{}$ Turbulent quantities

CHAPTER 1

INTRODUCTION

Fluid dynamics plays an important role in understanding the designed performance of many industrial products, particularly aircraft. Originally there were two main approaches to solving fluid dynamics problems: experimental and theoretical. The experimental method is the most realistic and straightforward, and leads to the development of techniques for understanding complicated physical phenomena of fluid flows. However, the cost of required equipment is even higher for large-scale applications and it is difficult to measure flow quantities in some specific situations. The theoretical or analytical method provides us with general information in mathematical form, which usually can describe only linear problems and is limited to very simple geometries and physical conditions. In recent years, the hardware of computers have improved dramatically, both in terms of processors and memory capacity. With this increase of computational capability, Computational Fluid Dynamics (CFD) has become a competitive way of studying fluid problems. This method has many attractive properties in helping people understand flow problems through numerical simulations: low cost, handling complicated geometries, solving nonlinear equations and unsteady flow. Leading edge CFD methods are at the foundation of this document.

1.1 CFD background

Based on the physics of fluid, the Euler equations are derived without considering viscous effects from shear stress and thermal conductivity effects from temperature gradients. These equations are simple and widely used for gasdynamics research. The Navier-Stokes equations are derived from physics with some hypotheses as a basic model for realistic fluid problems. For laminar flows, the governing equations can be solved numerically to produce excellent results. But for turbulent flows, which are commonly seen in engineering problems, the resources needed to resolve all turbulent length scales using the governing equations are far beyond current capability. Then appropriate approximations of the time dependent Navier-Stokes equations are necessary. Among the different methods in use, the Reynolds-Averaged Navier-Stokes (RANS) model is the most widely accepted for practical engineering problems. It is computationally efficient and can yield reasonable and accurate results for steady flow from low-speed to supersonic without separation. However, since it builds on an averaging process, this scheme can not give an accurate prediction of unsteady features, nor shock wave boundary layer interactions.

Large Eddy Simulation (LES) is higher-order modeling method which has advantages over the RANS models. The large eddies which interact with the mean flow to transport the primary components in momentum and energy are solved by a Navier-Stokes solver, and the small isotropic eddies which dissipate the energy are modeled. Unfortunately, for cases at wall-bounded regions where shear stress is relatively large from low to high Reynolds numbers, many fine meshes are required so that the computational cost becomes large [13]. Direct numerical simulation (DNS) using Navier-Stokes equations theoretically

can achieve more accurate predictions, but the high cost makes it virtually prohibitive in engineering applications. The hybrid LES-RANS method [14] is a reasonable alternative approach. It applies the RANS model to the near-wall viscous layer and the LES model to capture large unsteady features away from boundaries. For second order of accuracy, the numerical noise may overwhelm the LES sub-grid scale, hence high order schemes are demanded to make this approach applicable in solving fluid problems.

In this work, the Euler equations have been primarily used although preliminary results have been obtained with the RANS equations. Within the framework of the RANS model, a properly selected turbulence model is needed in order to close the governing equations. By far the most popular turbulence models applied to flow viscosity and heat transfer computations are the low-Reynolds number two-equation turbulence models, including the κ-ϵ and κ-ω models. These models often offer a good balance between complexity and accuracy [28] and here κ-ω model is our choice because of its robustness. At present, the RANS equations are closed by the high- or low-Reynolds number κ-ω turbulence model of Wilcox [59]. In order to deal with the stiffness of the κ-ω equations, a non-standard implementation of the κ-ω turbulence model has been used: the $\ln \omega$ is taken other than ω itself.

CFD methods have been widely used in engineering in the design and simulation of real fluid problems. However, truncation errors and computational cost are obstacles that still hinder large-scale applications of CFD. The Loci platform, originally developed by Dr. Luke at Mississippi State University, has been used in this work. Loci is both a $C++$ library and a programming framework specifically designed for developing compu-

tational simulations of physical fields, including CFD. One advantage the framework provides is automatic parallelization. Consequently, it provides strong support for problems that require exceptional computational costs, including much higher resolution simulations for increased accuracy in complicated geometry, high-order discontinuous Galerkin (DG) methods for unstructured meshes, and coupled Navier-Stokes equations that include physical models featuring multiple species. The high order numerical schemes are the research objects of this document.

1.2 Different high order discretization methods

In the current numerical simulation of flow problems, the governing equations need to be discretized in space, satisfying some essential stability conditions. The most popular methods are Finite Difference (FD), Finite Volume (FV) and Finite Element (FE) techniques, which have been widely applied in software developed by both academia and industry. Nowadays in the aerodynamics field, the large majority of numerical methods are at most second order accurate with the solution error decreasing on the order of $O(h^2)$, where h is a measure of length scale based on the grid size. In order to achieve high order of accuracy and reduce the effect of truncation errors, special techniques are required to construct high order numerical schemes for different types of discretization methods.

The idea behind the FD method is to replace the analytical time and space derivatives in the partial differential equations (PDEs) with discretized approximations. Then the obtained algebraic equations can be solved using classical numerical approximation schemes. Usually there are many forms of discretized derivatives to choose from and the approach

that maintains stability of the system and costs less is generally preferred. The FD method has the benefit of simplicity and efficiency in programming, and relatively low computational cost for simple geometry. High order of accuracy can be achieved by using more accurate approximations of the derivatives and the resulting truncation errors would be decreasing as $O(h^{OP})$ with $OP > 1$, where OP is the solution order of accuracy. However, since the FD method was originally designed and thoroughly investigated on structured grids, it takes a complicated process to accommodate to unstructured grids [38]. In addition, considering the stability of the solver, the geometry of the unstructured domain can not be sophisticated and the high order derivatives can not be approximated by complicated formulas. Therefore the FD method is better used in fundamental research on simple geometries.

The FV method, on the other hand, originates from a more physical approach. Based on the integral form of the conservation laws, the numerical flux at the element interface is evaluated to approximate the physical contribution from neighboring cells, then the solution is updated through solving the governing equations with the fluxes just computed. The popular upwind scheme [47] that is based on the characteristics of wave propagation is usually accepted in computing the convective flux components. High order solutions within FV methods can be obtained with the aid of reconstruction procedures. However, the reconstruction stencil for three-dimensional unstructured grids is fairly large and the computational cost is therefore expensive for large-scale fluid problems. So far the FV method is the dominant CFD method in applications because of its robustness, efficiency

and effectiveness in handling complex geometries. The problem of how to retain high order of accuracy while decreasing the size of the reconstruction stencil is still an open question.

The theory of the FE method is different from both of the methods already mentioned. The PDEs are first multiplied by an arbitrary test function and then integrated by parts. The solution is represented by a linear combination of so called ansatz functions which are piecewise polynomials. This strategy of discretizing the governing equations and representing the solution constitutes the kernel of a typical FE method. Depending on the specific type of test and ansatz functions, different kinds of FE methods have been developed, such as Galerkin method, Petrov-Galerkin method, etc. The classical FE methods can be divided into two branches, continuous and discontinuous methods. The continuous FE methods usually produce oscillations when solving Navier-Stokes equations, and explicit artificial dissipation has to be added to keep stability. Moreover, a fully coupled system of equations needs to be solved for each time step satisfying the continuity requirement at the element interface, which could cause extra burden on computations and then degrade efficiency. Discontinuous FE methods, especially the discontinuous Galerkin method where test and ansatz functions are identical, do not have global continuity constraints, so that the approximation space is different from the continuous solution space. Even the DG method has some stability problems when explicit time iteration is used. In this document implicit and semi-implicit methods have also been considered as alternatives. Because the DG method has a lot of advantages, this method was chosen as our main tool for discretizing the governing equations. Compared to the FD and FV methods applied to high order schemes, the FE method can simply construct a high order scheme by

increasing the degree of test or ansatz functions, thus the discretization stencil is compact without considering extra reconstruction procedure. This renders the DG method suitable for hp-adaption (p for polynomial degree). More details will be discussed in a subsequent chapter when p-multigrid scheme and time integration are introduced.

1.3 The Discontinuous Galerkin method

The word Galerkin comes from the person who first published on the topic of approximating solution of differential equations in 1915 [27]. After that a family of procedures named after Galerkin have been used within finite elements [60]. Among these procedures, the DG method was originally formulated by Reed and Hill [46] to solve the neutron transport problem. Lesaint and Raviart [37] conducted a preliminary analysis of this method in the following years. This method was not very popular until the middle of the 1980's. The revival and subsequent development of the DG method is due primarily to the work of Cockburn and Shu [19, 21]. The DG method was extended from solving linear hyperbolic equations to systems of multidimensional nonlinear hyperbolic equations. Also Cockburn and Shu were among the first to apply the DG method to the inviscid equations of gas dynamics [20]. The DG method is a locally conservative, stable, and high-order accurate method which can easily handle complex geometries and irregular meshes with hanging nodes by using polynomials of different degrees in different elements [18].

The first formula for discretizing the viscous terms of the Navier-Stokes equations in conjunction with the DG method was proposed by Bassi and Rebay [10]. Although some viscous test cases have been simulated successfully, this formulation (called the

BR1 scheme) was showing poor convergence for odd-order polynomial approximations and was unstable for certain model problems. Later Bassi and Rebay developed an improved scheme called BR2 which has much better numerical properties. Besides that, several other schemes for evaluating the diffusive fluxes were developed at that time: the main contributions are by Cockburn and Shu [22], Lomtev and Karniadakis [39], and Baumann and Oden [12]. Recently the DG method has been applied to turbulence modeling so that the RANS equations with turbulence models can be discretized in this framework. Bassi and Rebay [8] took the κ-ω model to the RANS solver where the turbulence parameters κ and ω are discretized using polynomial approximations in exactly the same manner as the conservative variables in the RANS equations. However, some numerical difficulties are still occurring. One is the stiffness caused by the highly non-linear source terms of the turbulence model equations, another is stiffness resulting from the grid stretching needed to resolve the near-wall behavior of the turbulent quantities. Realizability constraints and Schwarz inequality can be used to improve the solution [8]. Meanwhile, implicit time integration scheme of the fully coupled RANS and κ-ω system equations can be a solution for the grid stretching problem.

In shock regions, the high-order DG method is not very sensitive to the type and alignment of the mesh and it can minimize the interaction between shock wave and element interface so that this method is capable of sub-cell shock resolution [16]. Moreover, the DG method is inherently adapted to many solver acceleration strategies, including multigrid and hp-refinement techniques. Therefore for doing simulations of high Reynolds number

hypersonic turbulent flows with high order of accuracy, the DG method would possibly be the best choice.

1.4 Shock-capturing method

For convection dominated problems, especially for hyperbolic conservation laws, the existence of discontinuities (such as shock waves and contact discontinuities in high speed gas dynamics) is a problem for the numerical schemes currently in use. A shock wave is an important phenomenon in high-speed flow simulations. It is essentially a form of wave that propagates pressure disturbances, and in an extremely thin region the flow characteristics change drastically. Numerically capturing shock waves with computational tools is a basic requirement and essential capability of a numerical solver. Traditionally for low order of accuracy (first order), the Godunov type or the Roe type numerical schemes can resolve discontinuities monotonically without spurious numerical oscillations [47]. However, these methods excessively smear discontinuities and even in smooth region the solutions contain large numerical dissipation. Shocks are extended over several grid cells with poor resolutions. In recent years, many high order shock-capturing schemes have been designed and applied to flow problems successfully.

One practical approach for shock capturing is introducing limiting procedures. These methods were originally designed in the context of FD and FV discretization schemes and subsequently applied to the DG framework. The classical Runge-Kutta discontinuous Galerkin (RKDG) methods [23] combine approximate Riemann solvers and nonlinear operators (slope limiter) to satisfy Total Variation Bounded (TVB) in the mean. It is still an

open question to extend these limiters to higher order approximations because they reduce the approximation order to linear or constant even when the oscillations have been avoided.

The key in shock-capturing schemes is controlling dissipation. Another effective approach, called artificial viscosity, makes use of dissipation by adding artificial viscosity to the system to spread the discontinuity over a length scale so that it can be resolved in the approximating space by interpolating functions. Persson and Peraire [44] introduced a p-dependent artificial viscosity that scales like h/p, where h is length scale of grid size and p is interpolating polynomial degree, thus obtaining a shock with a width of $\delta \simeq Ch/p$ with C being constant coefficient. They also provide a robust discontinuity detection procedure, properly indicating the location of the shock wave where the amount of numerical dissipation is going to be added. Basically artificial diffusion techniques provide a robust and accurate approach to capturing shocks with high-order DG methods. However, the extra viscous-type terms in the governing equations require special ways to solve the nonlinear high-order derivatives, which complicate the solver.

Most recently, high order accurate essentially nonscillatory (ENO) [32] and weighted essentially nonscillatory (WENO) schemes [36] have become popular in numerical solutions of hyperbolic PDEs. These schemes have the capability to achieve arbitrarily high order formal accuracy in smooth regions while preserving nonlinear stability, nonoscillatory characteristic, and sharp discontinuity profiles with additional degrees of freedom. They are robust and suitable for problems containing both strong discontinuities and complex smooth solution features. However, the computational overhead is excessive for high order

approximations, and applying these methods to unstructured grids and three dimensions is still an open issue.

In this work we take the diffusion-based artificial viscosity method with shock detector to capture the shock wave. In fact, results will show that it is a robust and efficient way to achieve good sub-cell resolution of shocks. There are several alternatives on how to incorporate the artificial viscosity to the governing equations. Bassi and Rebay's shock-capturing approach [6] explicitly adds to the DG discretized equations an artificial viscosity term to control the high-order components of the numerical solution within elements and preserve the spatial resolution of discontinuities. Persson and Peraire's shock-capturing method [44] includes a discontinuity sensor and two options to control the amount of added viscosity by means of Laplacian type or physical type. All of these will be explored in detail in following chapters.

1.5 Contact-surface-capturing method

The physical phenomena occuring in nonequilibrium hypersonic flows are extremely complex and have not been completely understood for the past sixty years [48]. The simulation of nonequilibrium hypersonic flows has focused on chemical nonequilibrium. For example, a chemical kinetic model for air was employed to define the production terms in the species continuity equations. Initially 12 species and 64 chemical reactions composed the chemical model, then calculations performed for specific altitude and velocity regimes made it possible to eliminate many trivial reactions from the model so as to obtain a simplified model of 11 chemical species and 11 reactions involving only neutral species plus

15 reactions for charged species. In the thermodynamic area, the kinetic theory of gas was adopted. Since it did not consider the internal structure of atoms and molecules, the higher degrees of excitation beyond translation motion had to be resolved by approximations, and the vibrational and electronic degrees of freedom were assumed to remain in thermal equilibrium with translation while chemical reactions proceed at finite rates. The free-electron temperature was assumed to be equal to the heavy-particle translational temperature [24].

Understanding the basic theory of gasdynamics and fluid physics is necessary for a researcher who wants to develop implementation of prevailing numerical schemes to real gas problems. In the framework of the research for this document, some fundamental conclusions have been drawn based on the facts in which a two species gas model (nitrogen/oxygen pair or air/helium pair) has been selected for implementation and a frozen flow assumption has been made so that the thermodynamic properties can be computed simply without nonequilibrium contributions. All of the above simplifications provide a starting point for the current effort of applying the DG method to multiple species gas and capturing the species interface (contact surface) with high order approximations. The progress made in this dissertation is just the beginning of more sophisticated real gas or fluid simulations with high order accurate DG method, where the thermal-chemical nonequilibrium effects are considered in full. The thermal nonequilibrium energy is derived from different temperatures and the chemical reactions from multiple species are incorporated into the species continuity equations as source terms.

Flows involving multiple species are common in engineering applications of fluid vehicles. In a special case, flows composed of two fluids which do not mix and are separated

by a sharp fluid interface are found in many engineering and physical applications. Even if two-fluid dynamics problems have been extensively examined from both analytical and experimental points of view, solving this challenging problem by numerical methods is an important component of multiple species gasdynamics modeling and still in working progress. In contrast to the single-species flows for which many accurate and efficient numerical schemes have been developed and successfully applied to engineering applications, there are difficulties in designing simulation tools for general two-species flows because few accurate numerical schemes have been used to solve the sharp species interface effectively. Therefore, the research detailed in this document has concentrated on the development of a highly accurate DG numerical solver for the simulation of compressible, unsteady and inviscid two-fluid flows described by the three-dimensional Euler equations of gas dynamics.

The diffusion based artificial viscosity limiting strategies of shock capturing have been widely studied for many years. The principle of this approach has been described and can be used to initiate the study of novel contact-surface-capturing method. The idea is to detect the trouble cells in the solution domain where negative species mass fraction could cause nonphysical effects at the element interface and destroy the accuracy of the flux computations. Then artificial mass diffusion is added to the species continuity components of the governing equations with explicit time iterations at a pseudo time level, until the species mass fraction is positive and meaningful. After that, the corrected species component properties can be used to compute the numerical flux at the cell interface when advancing the solution in time. More details will be discussed in a following chapter.

1.6 Objectives of the present research effort

The long-term goal of the present research effort is to achieve accurate simulations of thermal-chemical nonequilibrium gas mixtures for high temperature and high Mach number conditions. The Loci software framework provides the basic coding environment and data structure support in building the current CFD solver. The Linux system and cluster of high performance computing nodes and processors provide the computing facilities and a massively parallel computing platform. The short-term goal of this effort is setting up the complete algorithm implementation for high-order DG method. The new solver has been built for solving three-dimensional RANS equations on unstructured grids, to prove the capability of computing diffusion components accurately. A two species gas model is adopted in this research, and typical test cases of air-helium shock-bubble interaction problem have been used as the primary examples to validate the algorithm. Because high-order DG methods have not yet been largely used in the simulation of multiple species flows, then our major contribution is preliminarily: constructing an efficient and robust higher-order DG method based solver with a mass diffusion based limiter model, and capturing the fluids with sharp material interface accurately in the high speed flow simulations. The numerical solutions can be achieved with physically meaningful gas mixture properties without losing order of accuracy when diffusion based limiters are used.

The outline of this document is the following. In Chapter 2, governing equations and physical models are briefly described. In Chapter 3, the basic mathematical representation of the DG method is introduced, including basis functions and geometrical issues. In Chapter 4, the numerical discretization of the equations in space by the DG method is discussed

in detail, and typical methods of computing the fluxes are introduced. In Chapter 5, the approach for time integration of the resulting system of equations is discussed, including acceleration strategies. In Chapter 6, details of the diffusion based limiter are presented. Numerical results of typical test cases are presented in Chapter 7 and some conclusions are given in Chapter 8.

CHAPTER 2

GOVERNING EQUATIONS

In this chapter the basic fluid dynamics governing equations are described. The nondimensional governing equations are supplemented with some physical models or closure models as required. Two-equations turbulence models and thermodynamics models for multiple species are introduced in this chapter as well.

2.1 Euler equations

Without considering viscous effects and thermal conduction, for a mixture of two species, the governing Euler equations in differential form read:

$$\frac{\partial U}{\partial t} + \frac{\partial \mathbf{F}_j}{\partial x_j} = 0 \tag{2.1}$$

where the vectors are:

$$U = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho e_0 \end{bmatrix} \qquad \mathbf{F}_j = \begin{bmatrix} \rho_1 u_j \\ \rho_2 u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ \rho u_j h_0 \end{bmatrix} \quad j = 1, 2, 3 \tag{2.2}$$

In these equations, a gas mixture composed of only two species is used. The fluid densities are ρ_1 and ρ_2, and the density of the mixture is $\rho = \rho_1 + \rho_2$. Also, t denotes time, x_1, x_2, and x_3 are global Cartesian coordinates, u_i is the fluid velocity in direction x_i, p is the fluid pressure and δ_{ij} is the Kronecker delta function. Other variables are defined as following: the total enthalpy h_0 and total energy e_0 are related by:

$$h_0 = e_0 + \frac{p}{\rho} \tag{2.3}$$

2.2 Thermodynamics model

In this research, a two-species model has been used as a simplified model for multiple species gas problems, but the number of species and components of the gas mixture can be extended easily depending on the specific engineering requirements. Typically we choose a two-species air model composed of nitrogen molecules and oxygen molecules as a simplified model for the general situation. The flow is considered to be frozen chemically and the individual components are thermally perfect, so that the mixture properties are only dependent on the mass fractions.

2.2.1 Gas mixture properties

Following the work of Grossman and Cinnella [30, 56], at high temperature the internal energy per unit mass of each species is assumed to be the sum of two portions, one in thermodynamic equilibrium and one in a nonequilibrium state, the latter being modeled by appropriate production rates. For situations where the gas temperature is not too high,

only the thermodynamic equilibrium portion of the internal energy per unit mass of each species is considered:

$$\begin{aligned} e_{N2} &= \tilde{e}_{N2}(T) \\ e_{O2} &= \tilde{e}_{O2}(T) \end{aligned} \tag{2.4}$$

where $\tilde{e}_{N2}$ and $\tilde{e}_{O2}$ are the known equilibrium portion of the energy and T is the translational temperature. It is convenient to express $\tilde{e}_{N2}$ and $\tilde{e}_{O2}$ in terms of the specific heat at constant volume $\tilde{C}_{v,N2} = d\tilde{e}_{N2}/dT$ and $\tilde{C}_{v,O2} = d\tilde{e}_{O2}/dT$ as follows:

$$\begin{aligned} \tilde{e}_{N2} &= \int_{T_{ref}}^{T} \tilde{C}_{v,N2}(\tau)\mathrm{d}\tau + e_{ref,N2} \\ \tilde{e}_{O2} &= \int_{T_{ref}}^{T} \tilde{C}_{v,O2}(\tau)\mathrm{d}\tau + e_{ref,O2} \end{aligned} \tag{2.5}$$

where $e_{ref,N2}$ and $e_{ref,O2}$ are the reference value for the internal energy, and T_{ref} is the reference value of temperature, normally set to zero for simplicity. The internal energy per unit mass of the mixture becomes:

$$e = \sum_{i=1}^{2} \frac{\rho_i}{\rho} e_i = \frac{\rho_{N2}}{\rho} e_{N2} + \frac{\rho_{O2}}{\rho} e_{O2} = \frac{\rho_{N2}}{\rho} \tilde{e}_{N2} + \frac{\rho_{O2}}{\rho} \tilde{e}_{O2} \tag{2.6}$$

where the ratio ρ_i/ρ is the mass fraction of the i-th species. Then the mixture specific heats and the gas constant can be defined by means of the general mixture rule, as follows:

$$\begin{aligned} \tilde{C}_v = \sum_{i=1}^{2} \frac{\rho_i}{\rho} \tilde{C}_{vi} = \frac{\rho_{N2}}{\rho} \tilde{C}_{v,N2} + \frac{\rho_{O2}}{\rho} \tilde{C}_{v,O2} \\ \tilde{C}_p = \sum_{i=1}^{2} \frac{\rho_i}{\rho} \tilde{C}_{pi} = \frac{\rho_{N2}}{\rho} \tilde{C}_{p,N2} + \frac{\rho_{O2}}{\rho} \tilde{C}_{p,O2} \\ \tilde{R} = \sum_{i=1}^{2} \frac{\rho_i}{\rho} \tilde{R}_i = \frac{\rho_{N2}}{\rho} \tilde{R}_{N2} + \frac{\rho_{O2}}{\rho} \tilde{R}_{O2} = \tilde{C}_p - \tilde{C}_v \end{aligned} \tag{2.7}$$

and it is possible to define an isentropic index $\tilde{\gamma}$:

$$\tilde{\gamma} = \frac{\tilde{C}_p}{\tilde{C}_v} \tag{2.8}$$

2.2.2 Equation of state

For conditions in which each individual species behaves as a thermally perfect gas, the thermal equation of state will relate pressure to translational temperature, in accordance to Dalton's law:

$$p = \sum_{i=1}^{2} \rho_i R_i T = \rho_{N2} R_{N2} T + \rho_{O2} R_{O2} T = \rho \tilde{R} T \tag{2.9}$$

where the density of the mixture is:

$$\rho = \sum_{i=1}^{2} \rho_i = \rho_{N2} + \rho_{O2} \tag{2.10}$$

The relationship between pressure and specific internal energy is defined implicitly through the temperature. The caloric equation of state is:

$$\begin{aligned} e &= \sum_{i=1}^{2} \frac{\rho_i}{\rho} [\int_{T_{ref}}^{T} \tilde{C}_{vi}(\tau) \mathrm{d}\tau + e_{ref,i}] \\ &= \frac{\rho_{N2}}{\rho} [\int_{T_{ref}}^{T} \tilde{C}_{v,N2}(\tau) \mathrm{d}\tau + e_{ref,N2}] + \frac{\rho_{O2}}{\rho} [\int_{T_{ref}}^{T} \tilde{C}_{v,O2}(\tau) \mathrm{d}\tau + e_{ref,O2}] \end{aligned} \tag{2.11}$$

2.3 Navier-Stokes equations

When the effects of viscosity and thermal conductivity are not neglected, the Euler equations cannot properly describe the nature of fluid now. The transport phenomena of momentum and heat by shear stress and temperature gradient are considered by adding a diffusive flux tensor to the governing equations. Derived from physical laws of conserva-

tion, the mathematical representation of the governing equations for viscous flows, called the Navier-Stokes equations, are given in vector form as:

$$\frac{\partial U}{\partial t} + \frac{\partial \mathbf{F}_j}{\partial x_j} + \frac{\partial \mathbf{G}_j}{\partial x_j} = 0 \tag{2.12}$$

where the conservative variables vector and inviscid flux vector have been defined in Equation (2.2). The extra term is the divergence of the diffusive flux tensor defined by:

$$\mathbf{G}_j = \begin{bmatrix} -\rho_1 \mathbf{V}_1 \\ -\rho_2 \mathbf{V}_2 \\ -\tau_{j1} \\ -\tau_{j2} \\ -\tau_{j3} \\ -(u_1\tau_{j1} + u_2\tau_{j2} + u_3\tau_{j3}) - q_j - \sum_s \rho_s h_s \mathbf{V}_s \end{bmatrix} \quad j = 1, 2, 3 \tag{2.13}$$

For air roughly being considered as a mixture of nitrogen and oxygen gases, two individual species conservation laws have been used to substitute the original mass conservation equation. Mass diffusion contribution has been added to the Euler governing equations in Equation (2.1) for the case of multiple species. In Equation (2.13) the species diffusion velocities need to be defined [40]. Fick's law of diffusion is employed to model species diffusion velocity $\mathbf{V}_s$:

$$\begin{aligned} \rho_1 \mathbf{V}_1 &= \rho D_{1,2} \nabla Y_1 \\ \rho_2 \mathbf{V}_2 &= \rho D_{2,1} \nabla Y_2 \end{aligned} \tag{2.14}$$

where $D_{1,2}$ and $D_{2,1}$ are the species diffusion coefficients and Y_s is the species mass fraction. More details of mass diffusion process will be discussed in a following chapter.

The τ_{ij} is a viscous stress tensor as:

$$\tau = \begin{pmatrix} \tau_{11} & \tau_{12} & \tau_{13} \\ \tau_{21} & \tau_{22} & \tau_{23} \\ \tau_{31} & \tau_{32} & \tau_{33} \end{pmatrix} \tag{2.15}$$

In this research air is used, which is assumed to be a Newtonian fluid for which the Boussinesq hypothesis is valid. Then τ_{ij} is symmetric and a linear function of the velocity gradients:

$$\tau_{ij} = \mu\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big) \tag{2.16}$$

where μ is the molecular viscosity coefficient or dynamic viscosity coefficient, and δ_{ij} is the Kronecker delta function. Using kinetic theory, μ is a known function of temperature for monoatomic gases. At moderate temperatures, air is composed of largely diatomic gases and we can still approximately use the relation between μ and T. The popular semi-empirical formula of Sutherland's law reads:

$$\frac{\mu}{\mu_0} = \Big(\frac{T}{T_0}\Big)^{\frac{3}{2}}\frac{T_0 + S}{T + S} \tag{2.17}$$

with μ_0, T_0 and S being constant values. The heat flux vector q_j in the energy equation of the viscous flux tensor is modeled according to Fourier's law as:

$$q_j = -\lambda\frac{\partial T}{\partial x_j} \tag{2.18}$$

where the thermal conductivity coefficient λ is related to the molecular viscosity coefficient μ and the specific heat capacity at constant pressure C_p by the nondimensional Prandtl number Pr in this form:

$$\lambda = \frac{\mu \cdot C_p}{Pr} \tag{2.19}$$

For air the Prandtl number is approximately constant for temperature between 200K and 600K and equal to 0.72. This is true for the cases used in this study.

2.4 Reynolds-Averaged Navier-Stokes equations

Laminar flow does only exist in very limited cases, and turbulent flows are dominant in most engineering problems, which complicates the simulation of practical flows. If the Navier-Stokes equations were still solved to obtain the solution of turbulent flow, then the amount of grid cells required to capture the turbulent length scales would be prohibitively large. Hence a more practical method is employing a reasonable approximation of the time dependent Navier-Stokes equations for high Reynolds number flow problems. The RANS equations result from time-averaging the governing equations. Physically they describe the motion of the fluid by using time-averaged values of the flow quantities. This approximation requires some models to characterize the properties of the turbulent quantities in order to close the system. The topic of turbulence models is going to be discussed in a following chapter. First of all, RANS equations are introduced as:

$$\frac{\partial U}{\partial t} + \frac{\partial \mathbf{F}_j}{\partial x_j} + \frac{\partial \mathbf{G}_j}{\partial x_j} = 0 \tag{2.20}$$

with F and G are defined in Equation (2.2) and Equation (2.13). Here τ_{ij} and q_j are modified to accommodate the effect of turbulent quantities as:

$$\begin{aligned} \hat{\tau}_{ij} &= 2(\mu+\mu_t)\Big(S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big) - \frac{2}{3}\rho\kappa\delta_{ij} \\ &= (\mu+\mu_t)\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big) - \frac{2}{3}\rho\kappa\delta_{ij} \qquad (2.21) \\ \hat{q}_j &= -\Big(\lambda+\lambda_t\Big)\frac{\partial T}{\partial x_j} \qquad (2.22) \end{aligned}$$

$$\lambda_t = \frac{\mu_t \cdot C_p}{Pr_t} \tag{2.23}$$

$$S_{ij} = \frac{1}{2}\Big(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\Big) \tag{2.24}$$

Here S_{ij} is the mean strain-rate tensor and μ_t is the eddy viscosity, or turbulent viscosity. Also, κ is the turbulent kinetic energy and λ_t is turbulent thermal conductivity. The Pr_t is the turbulent Prandtl number, which is taken constant and equal to 0.9. We find that if the form of eddy viscosity and turbulent thermal conductivity is similar to that of laminar flow viscosity and thermal conductivity respectively, then a simple way of computing turbulent viscous stress tensor and heat flux is to replace μ and λ with $\mu + \mu_t$ and $\lambda + \lambda_t$ as shown in Equation (2.21) and Equation (2.22). Notice that λ_t is a function of μ_t, then we are still in need of a function for the eddy viscosity in order to close the system mathematically. For a mixture of multiple species, the species viscosity and thermal conductivity coefficients are computed individually and the mixture properties are obtained by Wilke's rule:

$$\begin{aligned}
\mu &= \sum_{i=1}^{Ns} W_i \cdot \mu_i \\
\lambda &= \sum_{i=1}^{Ns} W_i \cdot \lambda_i \\
W_i &= \frac{X_i}{\sum_{j=1}^{Ns} X_j \cdot \Phi_{ij}} \\
\Phi_{ij} &= \frac{1}{\sqrt{8}}\Big(1 + \frac{M_i}{M_j}\Big)^{-\frac{1}{2}} \cdot \Big[1 + \sqrt{\frac{\mu_i}{\mu_j}}\big(\frac{M_i}{M_j}\big)^{\frac{1}{4}}\Big]^2
\end{aligned} \tag{2.25}$$

where Ns is number of species in the mixture, X_i is the mole fraction of the species component, and M_i is the molecular weight for each species.

2.5 Turbulence model and coupled equations

The classical Wilcox [59] $\kappa-\omega$ two-equations turbulence model is coupled to the mean flow governing equations. This model includes not only turbulent velocity scales, but also the turbulent length scales. Therefore, the RANS equations coupled with κ-ω equations are:

$$\frac{\partial U}{\partial t}+\frac{\partial \mathbf{F}_j}{\partial x_j}+\frac{\partial \mathbf{G}_j}{\partial x_j}+\mathbf{S}=0 \tag{2.26}$$

where the vectors are defined as follows:

$$U=\begin{bmatrix}\rho_1\\ \rho_2\\ \rho u_1\\ \rho u_2\\ \rho u_3\\ \rho e_0\\ \rho\kappa\\ \rho\tilde{\omega}\end{bmatrix}\quad \mathbf{F}_j=\begin{bmatrix}\rho_1 u_j\\ \rho_2 u_j\\ \rho u_1 u_j+p\delta_{1j}\\ \rho u_2 u_j+p\delta_{2j}\\ \rho u_3 u_j+p\delta_{3j}\\ \rho u_j h_0\\ \rho\kappa u_j\\ \rho\tilde{\omega}u_j\end{bmatrix}\quad \mathbf{G}_j=\begin{bmatrix}-\rho_1\mathbf{V}_1\\ -\rho_2\mathbf{V}_2\\ -\hat{\tau}_{1j}\\ -\hat{\tau}_{2j}\\ -\hat{\tau}_{3j}\\ -u_i\hat{\tau}_{ij}-\hat{q}_j-\sum\rho_s h_s\mathbf{V}_s\\ -(\mu+\sigma^*\hat{\mu}_t)\frac{\partial\kappa}{\partial x_j}\\ -(\mu+\sigma\hat{\mu}_t)\frac{\partial\tilde{\omega}}{\partial x_j}\end{bmatrix} \tag{2.27}$$

$$\mathbf{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta^* \rho \hat{\kappa} e^{\tilde{\omega}_r} \\ -\tau_{ij}\frac{\partial u_i}{\partial x_j} + \beta^* \rho \hat{\kappa} e^{\tilde{\omega}_r} \\ -\frac{\alpha}{\hat{\kappa}}\tau_{ij}\frac{\partial u_i}{\partial x_j} + \beta \rho e^{\tilde{\omega}_r} - (\mu + \sigma \hat{\mu}_t)\frac{\partial \tilde{\omega}}{\partial x_k}\frac{\partial \tilde{\omega}}{\partial x_k} \end{bmatrix} \qquad i,j = 1,2,3 \tag{2.28}$$

Here laminar flow variables have been defined in Equation (2.2). τ is turbulent stress tensor defined as:

$$\tau_{ij} = 2\hat{\mu}_t\left(S_{ij} - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \frac{2}{3}\rho\hat{\kappa}\delta_{ij} \tag{2.29}$$

Turbulent quantities κ stands for turbulent kinetic energy and the new variable $\tilde{\omega} = \ln\omega$ is used instead of ω which is the turbulent dissipation rate. $\tilde{\omega}_r$ comes from realizability constraints and the Schwarz inequality [8] where it is defined as $\tilde{\omega}_r = max(\tilde{\omega}, \tilde{\omega}_{r0})$, $\tilde{\omega}_{r0}$ is computed from the inequility. Other quantities are:

$$\hat{\kappa} = max(0, \kappa) \qquad \hat{\mu}_t = \alpha^* \rho \hat{\kappa} e^{-\tilde{\omega}_r} \tag{2.30}$$

γ, Pr and Pr_t are the ratio of gas specific heats, the molecular and turbulent Prandtl numbers (constant for perfect gases). The value of the κ-ω closure parameters α, α^*, β, β^*, σ, σ^* are those of the low-Reynolds κ-ω model in [8].

2.6 Nondimensional form of the governing equations

In order to reduce truncation errors from the finite precision of computers and decrease the condition number of the linear system when implicit time integration is used, some operations are made to guarantee that all variables are scaled to approximately the same order of magnitude. Scaling the variables and normalizing the governing equations is a necessary procedure in the solver implementation. Based on five reference quantities: length, density, velocity, viscosity, and mixture gas constant defined from the initial freestream conditions, other parameters can be made dimensionless consequently.

- Length($L_{ref} = d$): characteristic length (chord length, cylinder diameter, etc.)
- Density($\rho_{ref} = \rho_\infty$): Freestream density
- Velocity($\upsilon_{ref} = a_\infty$): Freestream speed of sound
- Viscosity($\mu_{ref} = \mu_\infty$): Freestream viscosity
- Gas constant($R_{ref} = R_\infty$): Given species mass fraction, $R_\infty = Y_{1\infty}R_1 + Y_{2\infty}R_2$

Meanwhile, some nondimensional derived variables are given here:

- Mach number: $Ma_\infty = \frac{U_\infty}{a_\infty}$
- Reynolds number: $Re_\infty = \frac{\rho_\infty a_\infty L_{ref}}{\mu_\infty}$
- Prandtl number: $Pr_\infty = \frac{\mu_\infty C_p}{\lambda_\infty}$
- Temperature: $T_\infty = \frac{a_\infty^2}{R_\infty}$

In the following, the nondimensional quantities are displayed with an overbar ($\bar{x}$), then after scaling the derived variables are given here:

$$\bar{t} = \frac{a_\infty t}{L_{ref}}, \quad \bar{p} = \frac{p}{\rho_\infty a_\infty^2}, \quad \bar{e} = \frac{e}{a_\infty^2}, \quad \bar{h}_0 = \frac{h_0}{a_\infty^2}, \quad \bar{a} = \frac{a}{a_\infty}, \quad \bar{\kappa} = \frac{\kappa}{a_\infty^2},$$

$$
\begin{aligned}
&\bar{\omega} = \omega \frac{L_{ref}}{a_\infty}, \quad \bar{h} = \frac{h}{a_\infty^2}, \quad \bar{x} = \frac{x}{L_{ref}}, \quad \bar{y} = \frac{y}{L_{ref}}, \quad \bar{z} = \frac{z}{L_{ref}}, \quad \bar{\rho} = \frac{\rho}{\rho_\infty}, \\
&\bar{e}_0 = \frac{e_0}{a_\infty^2}, \quad \bar{u} = \frac{u}{a_\infty}, \quad \bar{T} = \frac{T}{T_\infty} = \frac{T R_\infty}{a_\infty^2}, \quad \bar{\mu} = \frac{\mu}{\mu_\infty}, \quad \bar{\tau}_{ij} = \frac{\tau_{ij}}{\rho_\infty a_\infty^2}, \\
&\bar{q}_j = \frac{q_j}{\rho_\infty a_\infty^3}, \quad \bar{\lambda} = \frac{\lambda}{\mu_\infty R_\infty}, \quad \bar{\tilde{\omega}} = \tilde{\omega} - \ln \frac{a_\infty}{L_{ref}} = \ln \omega - \ln \frac{a_\infty}{L_{ref}} = \ln \bar{\omega}, \\
&\bar{\hat{\mu}}_t = \frac{\hat{\mu}_t}{\mu_\infty}, \quad \bar{\rho}_1 = \frac{\rho_1}{\rho_\infty}, \quad \bar{\rho}_2 = \frac{\rho_2}{\rho_\infty}, \quad \bar{\hat{\kappa}} = \frac{\hat{\kappa}}{a_\infty^2}, \quad \bar{\hat{\tau}}_{ij} = \frac{\hat{\tau}_{ij}}{\rho_\infty a_\infty^2}, \quad \bar{\hat{q}}_j = \frac{\hat{q}_j}{\rho_\infty a_\infty^3}, \\
&\bar{\lambda}_t = \frac{\lambda_t}{\mu_\infty R_\infty}, \quad \bar{D}_s = \frac{D_s \rho_\infty}{\mu_\infty}, \quad \bar{C}_v = \frac{C_v}{R_\infty}, \quad \bar{C}_p = \frac{C_p}{R_\infty}, \quad \bar{\gamma} = \gamma, \\
&\bar{R} = \frac{R}{R_\infty}, \quad \bar{\nu} = \frac{\nu \rho_\infty}{\mu_\infty}, \quad \overline{\rho_s \mathbf{V}_s} = \frac{\rho_s \mathbf{V}_s L_{ref}}{\mu_\infty}
\end{aligned}
\tag{2.31}
$$

After substituting the dimensionless variables for the parameters in the original RANS equations and rewriting the equations in terms of nondimensional variables, we can get the normalized governing equations as follows. Note that the overbar has been removed for clarity:

$$
\frac{\partial U}{\partial t} + \frac{\partial \mathbf{F}_j}{\partial x_j} + \frac{\partial \mathbf{G}_j}{\partial x_j} + \mathbf{S} = 0 \tag{2.32}
$$

where the vectors are :

$$
U = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho e_0 \\ \rho \kappa \\ \rho \tilde{\omega} \end{bmatrix} \quad \mathbf{F}_j = \begin{bmatrix} \rho_1 u_j \\ \rho_2 u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ \rho u_j h_0 \\ \rho \hat{\kappa} u_j \\ \rho \tilde{\omega} u_j \end{bmatrix} \quad \mathbf{G}_j = \begin{bmatrix} -\rho_1 \mathbf{V}_1 \\ -\rho_2 \mathbf{V}_2 \\ -\hat{\tau}_{1j} \\ -\hat{\tau}_{2j} \\ -\hat{\tau}_{3j} \\ -u_i \hat{\tau}_{ij} - \hat{q}_j - \sum \rho_s h_s \mathbf{V}_s \\ -\frac{1}{Re_\infty}(\mu + \sigma^* \hat{\mu}_t) \frac{\partial \kappa}{\partial x_j} \\ -\frac{1}{Re_\infty}(\mu + \sigma \hat{\mu}_t) \frac{\partial \tilde{\omega}}{\partial x_j} \end{bmatrix} \tag{2.33}
$$

$$\mathbf{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \tau_{ij}\frac{\partial u_i}{\partial x_j} - \beta^*\rho\hat{\kappa}e^{\tilde{\omega}_r} \\ -\tau_{ij}\frac{\partial u_i}{\partial x_j} + \beta^*\rho\hat{\kappa}e^{\tilde{\omega}_r} \\ -\frac{\alpha}{\hat{\kappa}}\tau_{ij}\frac{\partial u_i}{\partial x_j} + \beta\rho e^{\tilde{\omega}_r} - \frac{1}{Re_\infty}(\mu + \sigma\hat{\mu}_t)\frac{\partial\tilde{\omega}}{\partial x_k}\frac{\partial\tilde{\omega}}{\partial x_k} \end{bmatrix} \qquad i,j = 1,2,3 \tag{2.34}$$

with the nondimensional turbulent quantities and thermal conduction quantities defined here:

$$\begin{aligned} \tau_{ij} &= \frac{1}{Re_\infty}\hat{\mu}_t\Big[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big] - \frac{2}{3}\rho\hat{k}\delta_{ij} \\ \hat{\tau}_{ij} &= \frac{1}{Re_\infty}(\mu + \hat{\mu}_t)\Big[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\Big] - \frac{2}{3}\rho\hat{k}\delta_{ij} \\ \hat{q}_j &= -\frac{1}{Re_\infty}\Big(\lambda + \lambda_t\Big)\frac{\partial T}{\partial x_j} \end{aligned} \tag{2.35}$$

CHAPTER 3

THE DISCONTINUOUS GALERKIN METHOD

Theoretically, the DG discretization can be performed on any type of elements or mesh cells. In this document, both structured and unstructured grids have been used for three-dimensional computations. In the DG framework, a family of orthogonal, hierarchical bases, which are obtained from tensor product of Jacobi polynomial, are used as ansatz and test functions, up to fourth order of accuracy. Moreover, transformation between different spaces is an essential part of the current implementation. The global physical reference coordinate (Cartesian coordinate) is transformed into the local computational space, therefore the basis functions operate relative to the computational space and integration rules apply for each element. The popular Gaussian integration and quadrature rules will be discussed in detail in the following, because they are at the foundation of this effort.

3.1 Unstructured elements and basis functions system

In computational aerodynamics, where unstructured meshes are employed to accommodate geometric complexity, low-order finite elements and finite volumes are the prevailing discretization strategies. With the need for accurate solutions of viscous flow for aerodynamic configurations, high order discretization on unstructured meshes are of interest. The standard polynomial spectral methods provide high order accuracy but they are

limited to simple geometries. Also extending spectral methods to complex geometries has some limitations in adaptive capability. Thus we take the discontinuous Galerkin spectral method [57] as a general framework. This method employs standard hybrid grids so as to provide great flexibility in discretization. The unified polynomial basis, which is hierarchical and readily used in p-refinement, has been used. Moreover, data are expanded as tensor products and thus three-dimensional operations are reduced to a series of inexpensive one-dimensional operations. And the discontinuous Galerkin projections can be easily formulated to provide locality and robustness. Based on these advantages of this spectral method, the basis functions are constructed for hexahedra, prism, pyramid and tetrahedra elements as a way for computing ansatz and test functions in our DG framework for the governing equations.

The unified hybrid expansion basis functions using hexahedral, prismatic, pyramidic and tetrahedral domains in three dimensions are analyzed in a diagrammatic representation of the local collapsed coordinate systems in Figure 3.1. Details on how these coordinate systems are collapsed can be seen in [57].

After the collapsed coordinate systems have been established, the basis function expansions are defined as orthogonal in an L^2 inner product. That is, in hexahedral domain, the basis function is constructed as tensor product of three one-dimensional expansions. Warburton [57] has proved that this method of constructing the basis function is computationally attractive since the key operations, such as differentiation and integration, can be performed as a series of one-dimensional expansions. In the unstructured domain of collapsed coordinate systems, a similar process can be used to construct the basis func-

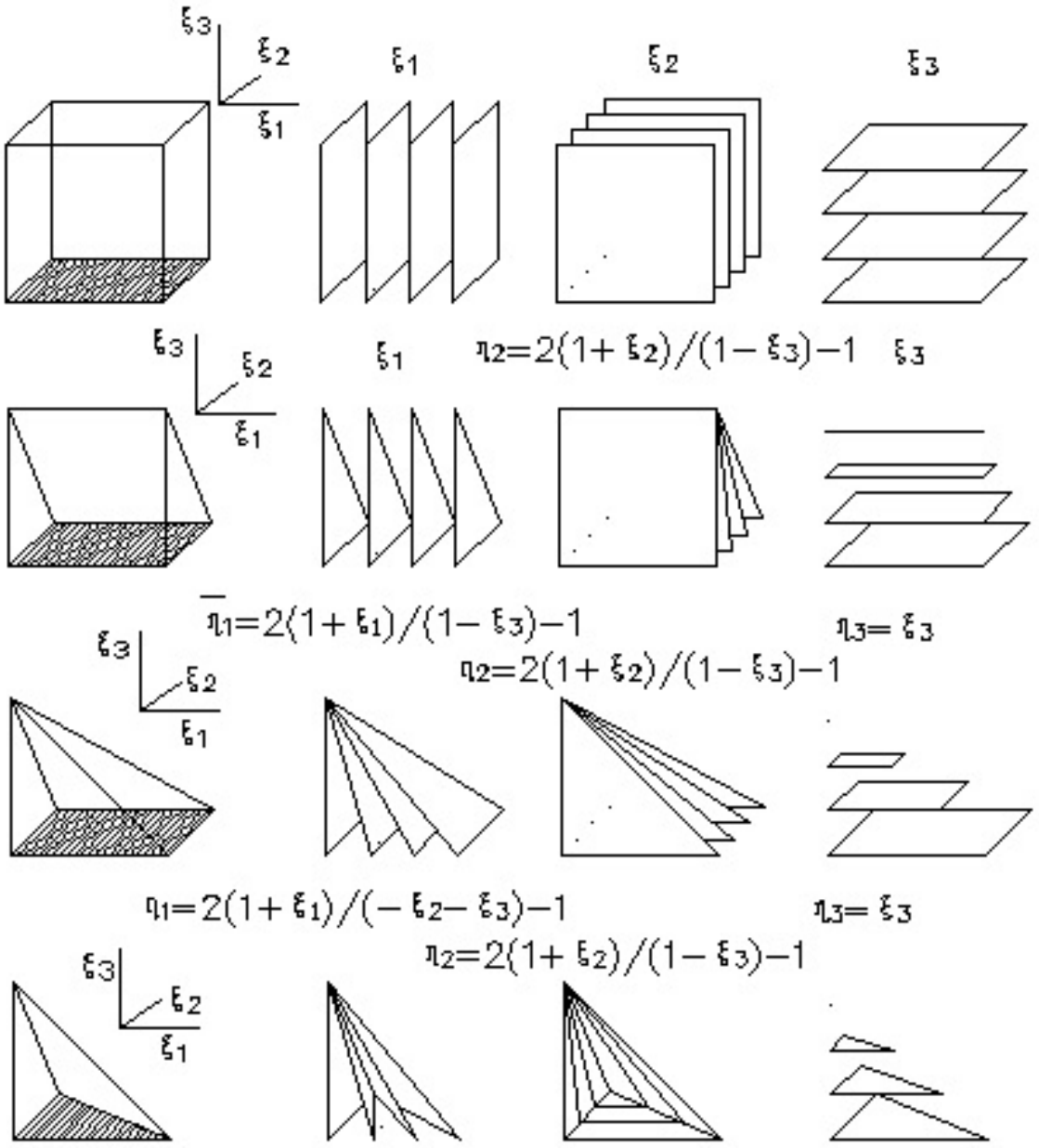

Figure 3.1

Local collapsed Cartesian coordinate systems in hexahedral, prismatic, pyramidic and tetrahedral domains

tion as a product involving tensors of three dimensions. This process sets up the relation between the local collapsed coordinate systems to the computational coordinate systems, specifically for solution representation, where the basic operation can be applied under these systems. Here the basis functions for hybrid domains are briefly summarized.

Based on the function $P_p^{\alpha,\beta}(z)$ which is the pth-order Jacobi polynomial, the principal functions $\tilde{\psi}_i^a(z)$, $\tilde{\psi}_{ij}^b(z)$, $\tilde{\psi}_{ijk}^c(z)$ for orthogonal expansions are defined:

$$\begin{aligned} \tilde{\psi}_i^a(z) &= P_i^{0,0}(z) \\ \tilde{\psi}_{ij}^b(z) &= \Big(\frac{1-z}{2}\Big)^i P_j^{2i+1,0}(z) \\ \tilde{\psi}_{ijk}^c(z) &= \Big(\frac{1-z}{2}\Big)^{i+j} P_k^{2i+2j+2,0}(z) \end{aligned} \tag{3.1}$$

where i, j, k is the index of the i, j, k-th expansion polynomial for each degree of freedom in one dimension, and z is the coordinate value at individual dimensions of the system.

Now three-dimensional expansions are defined in terms of the principal functions as:

$$\begin{aligned} \text{Hexahedral} \quad \phi_{pqr}(\xi_1,\xi_2,\xi_3) &= \tilde{\psi}_p^a(\xi_1)\tilde{\psi}_q^a(\xi_2)\tilde{\psi}_r^a(\xi_3) \\ \text{Prismatic} \quad \phi_{pqr}(\xi_1,\xi_2,\xi_3) &= \tilde{\psi}_p^a(\xi_1)\tilde{\psi}_q^a(\eta_2)\tilde{\psi}_{qr}^b(\xi_3) \\ \text{Pyramidic} \quad \phi_{pqr}(\xi_1,\xi_2,\xi_3) &= \tilde{\psi}_p^a(\bar{\eta_1})\tilde{\psi}_q^a(\eta_2)\tilde{\psi}_{pqr}^c(\eta_3) \\ \text{Tetrahedral} \quad \phi_{pqr}(\xi_1,\xi_2,\xi_3) &= \tilde{\psi}_p^a(\eta_1)\tilde{\psi}_{pq}^b(\eta_2)\tilde{\psi}_{pqr}^c(\eta_3) \end{aligned} \tag{3.2}$$

where

$$\eta_1 = \frac{2(1+\xi_1)}{(-\xi_2-\xi_3)} - 1,\ \bar{\eta_1} = \frac{2(1+\xi_1)}{(1-\xi_3)} - 1,\ \eta_2 = \frac{2(1+\xi_2)}{(1-\xi_3)} - 1,\ \eta_3 = \xi_3 \tag{3.3}$$

are the three-dimensional normalized coordinates. These expansions are polynomials in terms of local collapsed coordinates, therefore can be reformulated as functions of the

computational coordinate systems. On element faces, the basis functions can be computed in the same way by simply substituting the coordinate values into the formulation and interpolate to find the local collapsed coordinates in space. In this implementation, basis functions for all element types are used up to fourth order of approximation accuracy. The details of basis functions for each degree of freedom for hexahedral element are given in APPENDIX A.

3.2 Transformation to computational space

In order to generalise all operations on different element geometry, the physical element domains (Cartesian coordinates) need to be transformed to a computational reference space (direct coordinate system) which is locally identical to the coordinates in hexahedral element space. Now spatial Jacobian matries are defined here, and the computational coordinate systems can be transformed from the global Cartesian coordinate systems as shown in Figure 3.2 (taking prism domain as an example).

Using the prism element as an example for general unstructured domains for computation, a reference prism can be mapped from the computational space (ξ, η, ζ) to an arbitrary prism in the physical space (x, y, z) with a linear transformation:

$$\begin{aligned}\bar{X}(\xi,\eta,\zeta) = \quad & \frac{1}{8}((1-\xi)(1-\eta)(1-\zeta)\bar{X}_0 + (1+\xi)(1-\eta)(1-\zeta)\bar{X}_1 + \\ & (1-\xi)(1+\eta)(1-\zeta)\bar{X}_2 + (1+\xi)(1+\eta)(1-\zeta)\bar{X}_3 + \\ & (1-\xi)(1-\eta)(1+\zeta)\bar{X}_4 + (1+\xi)(1-\eta)(1+\zeta)\bar{X}_5 + \\ & (1-\xi)(1+\eta)(1+\zeta)\bar{X}_6 + (1+\xi)(1+\eta)(1+\zeta)\bar{X}_7) \end{aligned} \tag{3.4}$$

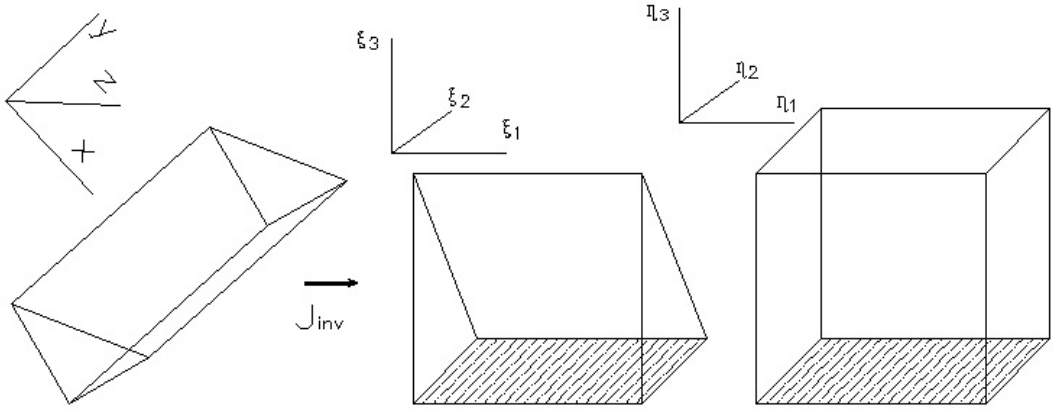

Figure 3.2

Coordinates transformation from global Cartesion to local collapsed and normalized coordinates

here $\bar{X}_0$ to $\bar{X}_7$ are the nodal positions at global coordinates, some of which could be collapsed to a common value on unstructured elements. Hence the Jacobian of the transformation is defined as:

$$J = \left| \frac{\partial(X, Y, Z)}{\partial(\xi, \eta, \zeta)} \right| = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \zeta} \end{pmatrix} = const. \tag{3.5}$$

Note that the Jacobian is independent of ξ, η and ζ and identical to the twice the volume of the prism element for this case. As an alternative method, Bezier curves with Berstein polynomials [45] can be used to represent the geometry, and can be differentiated to compute the Jacobian directly.

3.3 Gaussian integration and quadrature rules

The numerical integration from cell and face integrals is performed in the computational reference domain. For simplicity, the Gaussian integration is used to approximate the numerical integration on each element for both cell integrals and face integrals. Let us take the numerical flux integration of cell and face components in discretized Euler equations as an example.

For cell integration, applying Gaussian rule to the following equations results in:

$$\begin{aligned}
&\int_{\Omega} \mathbf{F}(U(\vec{x}(\xi_1,\xi_2,\xi_3))) \cdot \nabla\psi_k(\vec{x}(\xi_1,\xi_2,\xi_3))\mathrm{d}\Omega \\
&= \int_{\xi_1}\int_{\xi_2}\int_{\xi_3} J^{-1}(\xi_1,\xi_2,\xi_3)\mathbf{F}(\xi_1,\xi_2,\xi_3)\cdot\nabla\psi_k(\xi_1,\xi_2,\xi_3)J(\xi_1,\xi_2,\xi_3)\mathrm{d}\xi_1\mathrm{d}\xi_2\mathrm{d}\xi_3 \\
&= \sum_{q=1}^{nq} \omega_q J^{-1}(\xi_{1,q},\xi_{2,q},\xi_{3,q})\mathbf{F}(\xi_{1,q},\xi_{2,q},\xi_{3,q})\cdot\nabla\psi_k(\xi_{1,q},\xi_{2,q},\xi_{3,q})|J(\xi_{1,q},\xi_{2,q},\xi_{3,q})|
\end{aligned}$$

where $\mathrm{d}\Omega$ is volume element operator in integration. The element face integration can be approximated by a weighted summation as:

$$\begin{aligned}
&\oint_{\partial\Omega} \widehat{\mathbf{F}}(U(\vec{x}(\xi_1,\xi_2,\xi_3))) \cdot \mathbf{n}\psi_k(\vec{x}(\xi_1,\xi_2,\xi_3))\mathrm{d}\sigma \\
&= \int_{\eta_1}\int_{\eta_2} J^{-1}(\eta_1,\eta_2)\widehat{\mathbf{F}}(\eta_1,\eta_2)\cdot\mathbf{n}\psi_k(\eta_1,\eta_2)J(\eta_1,\eta_2)\mathrm{d}\eta_1\mathrm{d}\eta_2 \\
&= \sum_{q=1}^{nq} \omega_q J^{-1}(\eta_{1,q},\eta_{2,q})\widehat{\mathbf{F}}(\eta_{1,q},\eta_{2,q})\cdot\mathbf{n}\psi_k(\eta_{1,q},\eta_{2,q})|J(\eta_{1,q},\eta_{2,q})|
\end{aligned}$$

where $\mathrm{d}\sigma$ is surface element operator in integration, nq is the number of quadrature points for integration, ω_q is the weight for each quadrature point, ξ_i are the local coordinates on cells and η_i are the faces local coordinates. The choice of reference coordinates to two-dimensional and three-dimensional elements and the details of the quadrature points and integration are given in APPENDIX B.

3.4 Basis function and mass matrix

Three-dimensional ansatz and test functions up to fourth order of accuracy have been computed for this study. Lastly, one important thing needs to be mentioned which is so called element mass matrix M_{ij}. This mass matrix is used mainly for time integration and iterations. It is especially useful in implicit time integration methods. The three-dimensional mass matrix is:

$$M_{ij} = \int_{\xi_1} \int_{\xi_2} \int_{\xi_3} \psi_i(\xi_1, \xi_2, \xi_3) \psi_j(\xi_1, \xi_2, \xi_3) |J(\xi_1, \xi_2, \xi_3)| \mathrm{d}\xi_1 \mathrm{d}\xi_2 \mathrm{d}\xi_3 \tag{3.6}$$

where i and j are degrees of freedom for each quadrature point. The mass matrix is symmetric, and for orthogonal polynomial functions, this mass matrix is diagonal. The mass matrix is stored and can be reused in each time integration.

CHAPTER 4

DISCONTINUOUS GALERKIN SPACE DISCRETIZATION

In this chapter, the Euler equations and RANS equations are discretized in space using the discontinuous Galerkin method. Details of applying the DG method to the governing equations and turbulence model are introduced. Several classical methods to compute the inviscid flux and viscous flux are mentioned, and boundary conditions treatments are also given for completeness.

4.1 Discretization of Euler equations

The differential form of the Euler equations, Equation (2.1), is multiplied by an arbitrary weighting function ψ and integrated over the domain as:

$$\int_{\Omega} \frac{\partial U}{\partial t} \psi \mathrm{d}\Omega + \int_{\Omega} \nabla \cdot \mathbf{F}(U) \psi \mathrm{d}\Omega = 0 \tag{4.1}$$

The flux term is integrated by parts (using the Gaussian divergence theorem) to obtain the weak form:

$$\int_{\Omega} \nabla \cdot \mathbf{F}(U) \psi \mathrm{d}\Omega = \int_{\partial\Omega} (\mathbf{F}(U) \cdot \mathbf{n}) \psi \mathrm{d}\sigma - \int_{\Omega} \mathbf{F}(U) \cdot \nabla \psi \mathrm{d}\Omega \tag{4.2}$$

where $\partial\Omega$ denotes the boundary of the solution domain Ω. Now consider Ω_h as an approximation of entire domain Ω, with Σ_h and Γ_h being defined as an approximation of element interface and domain boundary separately. A discretization (here discretize the domain by

unstructured meshes) of the domain Ω_h is a set of non-overlapping elements $\mathscr{T}_h = \{e\}$ with the corresponding interface $\mathscr{I}_h = \{i\}$ and boundary $\mathscr{B}_h = \{b\}$. Let us consider the functions space:

$$V_h := \left\{ v \in L^2(\Omega_h) : \psi|_e \in P^k(e) \quad \forall e \in \mathscr{T}_h \right\} \tag{4.3}$$

where $P^k(e)$ denotes the space of polynomial functions of degree at most k in element e. Equation (4.1) can be rewritten as the summations of integrals over the entire computational domain and $U_h \in V_h$ so that:

$$\begin{aligned} \sum_{e \in \mathscr{T}_h} \int_{\Omega_e} \frac{\partial U_h}{\partial t} \psi_h \mathrm{d}\Omega \quad &- \quad \sum_{e \in \mathscr{T}_h} \int_{\Omega_e} \mathbf{F}(U_h) \cdot \nabla \psi_h \mathrm{d}\Omega \\ &+ \quad \sum_{e \in \mathscr{T}_h} \int_{\partial \Omega_e} \mathbf{F}(U_h) \cdot \mathbf{n} \psi_h \mathrm{d}\sigma = 0 \end{aligned} \tag{4.4}$$

holds for an arbitrary test function $\psi \in V_h$. The various summations in Equation (4.4) are mandatory since the integration by part rule can not be applied to the entire doman Ω_h because of the discontinuities at element interfaces. So we need to split the integral over Ω_h into the group of integrals over the elements of $\mathscr{T}_h$ and then apply integration by parts to each integral of the group. This is a valid operation since the functions are locally continuous inside the elements of $\mathscr{T}_h$ [8].

Let us now rewrite the various summations in Equation (4.4) as integrals over the entire domain Ω_h, the interface Σ_h and the boundary Γ_h. The summations of integrals over the element domains Ω_e are obviously equivalent to integrals over the entire domain Ω_h. Then the system of equations over the domain can be rewritten as:

$$\int_{\Omega_h} \frac{\partial U_h}{\partial t} \psi_h \mathrm{d}\Omega - \int_{\Omega_h} \mathbf{F}(U_h) \cdot \nabla \psi_h \mathrm{d}\Omega + \int_{\Sigma_h} \mathbf{F}(U_h) \cdot \mathbf{n} \psi_h \mathrm{d}\sigma = 0 \tag{4.5}$$

Now we need to use proper numerical method to compute each part of the above equations. The cell integration which is the second term in Equation (4.5) can be approximated by integration rules in a straightforward fashion:

$$\int_{\Omega_h} \mathbf{F}(U_h) \cdot \nabla \psi_h \mathrm{d}\Omega \approx \sum_{q=1}^{nq} \omega_q J^{-1} \mathbf{F} \cdot \nabla \psi_k |J| \tag{4.6}$$

where $k = 0 \ldots ndf$ with ndf polynomial number of degree of freedom. This flux contribution has been evaluated as the summation of the integration from each quadrature point in the cell. The face flux integration, which is the third term in Equation (4.5), can be represented as:

$$\begin{aligned} &\int_{\Sigma_h} \mathbf{F}(U_h) \cdot \mathbf{n} \psi_h \mathrm{d}\sigma = \int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}) \psi_h \mathrm{d}\sigma \\ &= \int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)(\psi_h^- - \psi_h^+) \mathrm{d}\sigma + \int_{\Gamma_h} \mathbf{F}(U_h^b) \cdot \mathbf{n} \psi_h \mathrm{d}\sigma \end{aligned} \tag{4.7}$$

where numerical flux $\widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)$ is an approximation of the physical flux at interface, and $\mathbf{F}(U_h^b)$ is flux at boundary. The first term on the second line is the numerical flux approximation from element interfaces, and the second term is the numerical flux on the boundary of the domain. The well developed approximate Riemann solver [51] can now be used. The details of the approximation to the face flux contribution are given in a following section.

4.2 Discretization of RANS equations

The dimensionless RANS equations coupled with κ-ω turbulence model equations are given in Equation (2.32). Based on that, we can start applying DG space discretization to the compact form of RANS equations and κ-ω turbulence model equations as:

$$\frac{\partial U}{\partial t} + \nabla \cdot \mathbf{F}(U) + \nabla \cdot \mathbf{G}(U, \nabla U) + \mathbf{S}(U, \nabla U) = 0 \tag{4.8}$$

The differential form is multiplied by an arbitrary weighting function ψ and integrated over the domain as:

$$\begin{aligned} \int_\Omega \frac{\partial U}{\partial t}\psi \mathrm{d}\Omega \quad &+ \quad \int_\Omega \nabla \cdot \mathbf{F}(U)\psi \mathrm{d}\Omega + \int_\Omega \nabla \cdot \mathbf{G}(U, \nabla U)\psi \mathrm{d}\Omega \\ &+ \quad \int_\Omega \mathbf{S}(U, \nabla U)\psi \mathrm{d}\Omega = 0 \end{aligned} \tag{4.9}$$

where the convective flux integrations have been computed in Equation (4.6) and Equation (4.7) from last section. Now let us consider the diffusive flux contribution which is the third term in Equation (4.9). This flux can be reformulated as a first-order system as:

$$\begin{cases} \mathbf{Z} = \nabla U \\ \nabla \cdot \mathbf{G}(U, \nabla U) = \nabla \cdot [\mathscr{A}(U)\mathbf{Z}] \end{cases} \tag{4.10}$$

Note that it has been proved that the diffusive flux $\mathbf{G}(U, \nabla U)$ can be represented as a nonlinear function operator $\mathscr{A}(U)$ of conservative variables U multiplied by the solution gradient $\mathbf{Z}$ [8]. Then this first-order system is integrated by parts to obtain the weak form as:

$$\begin{cases} \int_\Omega \mathbf{g} \cdot \mathbf{Z} \mathrm{d}\Omega = \int_\Omega \mathbf{g} \cdot \nabla U \mathrm{d}\Omega = -\int_\Omega U \nabla \cdot \mathbf{g} \mathrm{d}\Omega + \int_{\partial\Omega} U \mathbf{g} \cdot \mathbf{n} \mathrm{d}\sigma \\ \int_\Omega \nabla \cdot \mathbf{G}(U, \nabla U)\psi \mathrm{d}\Omega = \int_\Omega \nabla \cdot [\mathscr{A}(U)\mathbf{Z}]\, \psi \mathrm{d}\Omega = \\ \qquad\qquad -\int_\Omega [\mathscr{A}(U)\mathbf{Z}] \cdot \nabla\psi \mathrm{d}\Omega + \int_{\partial\Omega} [\mathscr{A}(U)\mathbf{Z}] \cdot \mathbf{n}\psi \mathrm{d}\sigma \end{cases} \tag{4.11}$$

where $\mathbf{g}$ is an arbitrary weighting vector function (note: in this context, we used bold for vector and normal for scalar parameters). Substitute Equation (4.11) to Equation (4.9), we can get:

$$\begin{cases} \int_\Omega \mathbf{g}\cdot\mathbf{Z}\mathrm{d}\Omega = \int_\Omega \mathbf{g}\cdot\nabla U\mathrm{d}\Omega = -\int_\Omega U\nabla\cdot\mathbf{g}\mathrm{d}\Omega + \int_{\partial\Omega} U\mathbf{g}\cdot\mathbf{n}\mathrm{d}\sigma \\ \int_\Omega \frac{\partial U}{\partial t}\psi\mathrm{d}\Omega - \int_\Omega \mathbf{F}(U)\cdot\nabla\psi\mathrm{d}\Omega + \int_{\partial\Omega}\mathbf{F}(U)\cdot\mathbf{n}\psi\mathrm{d}\sigma \\ \qquad\qquad - \int_\Omega [\mathscr{A}(U)\mathbf{Z}]\cdot\nabla\psi\mathrm{d}\Omega + \int_{\partial\Omega}[\mathscr{A}(U)\mathbf{Z}]\cdot\mathbf{n}\psi\mathrm{d}\sigma + \int_\Omega \mathbf{S}(U,\mathbf{Z})\psi\mathrm{d}\Omega = 0 \end{cases} \tag{4.12}$$

Let us consider the functions space:

$$\begin{aligned} V_h &:= \left\{\psi\in L^2(\Omega_h) : \psi|_e \in P^k(e) \quad \forall e\in\mathscr{T}_h\right\} \\ \mathbf{G}_h &:= \left\{\mathbf{g}\in [L^2(\Omega_h)]^d : \mathbf{g}|_e \in [P^k(e)]^d \quad \forall e\in\mathscr{T}_h\right\} \end{aligned} \tag{4.13}$$

Equation (4.12) can be rewritten as the summations of integrals over the entire computational domain and $U_h \in V_h$ so that:

$$\begin{aligned} \sum_{e\in\mathscr{T}_h}\int_{\Omega_e}\mathbf{g}_h\cdot\mathbf{Z}_h\mathrm{d}\Omega &= -\sum_{e\in\mathscr{T}_h}\int_{\Omega_e} U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega + \sum_{e\in\mathscr{T}_h}\int_{\partial\Omega_h}\widehat{U}_h\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma \\ \sum_{e\in\mathscr{T}_h}\int_{\Omega_e}\frac{\partial U_h}{\partial t}\psi_h\mathrm{d}\Omega &- \sum_{e\in\mathscr{T}_h}\int_{\Omega_e}\mathbf{F}(U_h)\cdot\nabla\psi_h\mathrm{d}\Omega + \sum_{e\in\mathscr{T}_h}\int_{\partial\Omega_e}\widehat{\mathbf{F}}(U_h^-,U_h^+;\mathbf{n})\psi_h\mathrm{d}\sigma \\ &- \sum_{e\in\mathscr{T}_h}\int_{\Omega_e}[\mathscr{A}_h(U_h)\mathbf{Z}_h]\cdot\nabla\psi_h\mathrm{d}\Omega + \sum_{e\in\mathscr{T}_h}\int_{\partial\Omega_e}\left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right]\cdot\mathbf{n}\psi_h\mathrm{d}\sigma \\ &+ \sum_{e\in\mathscr{T}_h}\int_{\Omega_e}\mathbf{S}(U_h,\mathbf{Z}_h)\psi_h\mathrm{d}\Omega = 0 \end{aligned} \tag{4.14}$$

holds for arbitrary test functions $\psi\in V_h$ and $\mathbf{g}\in\mathbf{G}_h$. $\widehat{U}_h$ and $\left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right]$ are numerical flux approximation of U and $[\mathscr{A}(U)\mathbf{Z}]$ at element interface. Now taking the procedure similar to Equation (4.5), and rewriting the system of equations over the domain as:

$$\int_{\Omega_h}\mathbf{g}_h\cdot\mathbf{Z}_h\mathrm{d}\Omega = -\int_{\Omega_h}U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega + \int_{\Sigma_h}\widehat{U}_h\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma$$

$$\begin{aligned}\int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega &- \int_{\Omega_h} \mathbf{F}(U_h)\cdot\nabla\psi_h \mathrm{d}\Omega + \int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n})\psi_h \mathrm{d}\sigma \\ &- \int_{\Omega_h} [\mathscr{A}_h(U_h)\mathbf{Z}_h]\cdot\nabla\psi_h \mathrm{d}\Omega + \int_{\Sigma_h} \left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right]\cdot\mathbf{n}\psi_h \mathrm{d}\sigma \\ &+ \int_{\Omega_h} \mathbf{S}(U_h, \mathbf{Z}_h)\psi_h \mathrm{d}\Omega = 0 \end{aligned} \tag{4.15}$$

Since the convective components of the flux contribution are identical to those from the Euler equations, here the computation of the diffusive terms is our concern, and several methods can be utilized to obtain the numerical approximation of the diffusion [8, 22]. The details of both convective and diffusive flux approximations will be discussed in the following sections.

4.3 Computing the convective flux

The typical methods used to compute the element interface convective flux in finite volumes can be applied directly to the DG discretized governing equations. Currently four numerical schemes have been examined and used to compute the convective flux. In this research, since two species gas models have been used, the standard form of the numerical flux approximation is modified to accommodate the species mass fractions.

4.3.1 Roe flux difference splitting

Two frequently used operators are introduced at first. The arithmetic average of a quantity f will be denoted by:

$$\langle f \rangle = \frac{f_l + f_r}{2} \tag{4.16}$$

with the subscript l indicating the left state and the subscript r the right state, respectively. Also the jump of a quantity f will be defined as:

$$[f] = f_r - f_l \tag{4.17}$$

The solution of the approximate Riemann problem [51] involves the determination of the cell interface fluxes as a summation over wave speeds:

$$\mathbf{F}_i = \langle \mathbf{F} \rangle - \frac{1}{2}([\mathbf{F}]_A + [\mathbf{F}]_B + [\mathbf{F}]_C) \tag{4.18}$$

where the absolute value of the wave speeds must be taken into the formula for the jumps in the fluxes, which involves projecting $[\mathbf{F}]$ onto the eigenvectors:

$$[\mathbf{F}] = [\mathbf{F}]_A + [\mathbf{F}]_B + [\mathbf{F}]_C = \sum_{i=1}^{6} \tilde{\alpha}_i |\tilde{\lambda}_i| \tilde{\mathbf{r}}^{(i)} \tag{4.19}$$

Then the derived Roe interface flux approximation is:

$$\mathbf{F}_i = \frac{1}{2}\left(\mathbf{F}_l + \mathbf{F}_r - \sum_{i=1}^{6} \tilde{\alpha}_i |\tilde{\lambda}_i| \tilde{\mathbf{r}}^{(i)}\right) \tag{4.20}$$

where $\tilde{\alpha}_i$, $\tilde{\lambda}_i$ and $\tilde{\mathbf{r}}^{(i)}$ are the wave strengths, eigenvalues and right eigenvectors respectively. A simple algorithm for computing the Roe numerical flux according to the above formulae is:

1. Compute the Roe averaged values of $\tilde{u}, \tilde{v}, \tilde{w}, \tilde{H}$ and $\tilde{a}$.
2. Compute the averaged eigenvalues $\tilde{\lambda}_i$.
3. Compute the averaged right eigenvectors $\tilde{\mathbf{r}}^{(i)}$.
4. Compute the wave strengths $\tilde{\alpha}_i$.
5. Use all derived quantities to compute the interface flux $\mathbf{F}_i$ from Equation (4.20).

In above formulae, the $[\mathbf{F}]_A$ term corresponds to the repeated eigenvalue $\lambda_i = \tilde{U}_n$ ($\tilde{U}_n$ is contravariant velocity) and may be written from [56] as:

$$[\mathbf{F}]_A = ([\rho] - \frac{[p]}{\tilde{a}^2}) \cdot |\tilde{U}_n| \cdot \begin{bmatrix} \tilde{\rho}_1 \\ \tilde{\rho}_2 \\ \tilde{U}_x \\ \tilde{U}_y \\ \tilde{U}_z \\ (\tilde{h}_0 - \frac{\tilde{a}^2}{\tilde{\gamma}-1}) \end{bmatrix} + \tilde{\rho} \cdot |\tilde{U}_n| \cdot \begin{bmatrix} [\frac{\rho_1}{\rho}] \\ [\frac{\rho_2}{\rho}] \\ ([U_x] - [U_n] \cdot n_x) \\ ([U_y] - [U_n] \cdot n_y) \\ ([U_z] - [U_n] \cdot n_z) \\ \Theta \end{bmatrix} \tag{4.21}$$

where:

$$\Theta = (\tilde{U}_x \cdot [U_x] + \tilde{U}_y \cdot [U_y] + \tilde{U}_z \cdot [U_z]) - \tilde{U}_n \cdot [U_n] - (\tilde{\Psi}_1 [\frac{\rho_1}{\rho}] + \tilde{\Psi}_2 [\frac{\rho_2}{\rho}]) \tag{4.22}$$

Similarly, the $[\mathbf{F}]_B$ and $[\mathbf{F}]_C$ corresponding to the eigenvalues $\tilde{U}_n + \tilde{a}$ and $\tilde{U}_n - \tilde{a}$ are:

$$[\mathbf{F}]_{B,C} = \frac{[p] \pm \tilde{\rho}\tilde{a}[U_n]}{2\tilde{a}^2} \cdot |\tilde{U}_n \pm \tilde{a}| \cdot \begin{bmatrix} \tilde{\rho}_1 \\ \tilde{\rho}_2 \\ \tilde{U}_x \pm \tilde{a} \cdot n_x \\ \tilde{U}_y \pm \tilde{a} \cdot n_y \\ \tilde{U}_z \pm \tilde{a} \cdot n_z \\ (\tilde{h}_0 \pm \tilde{U}_n \cdot \tilde{a}) \end{bmatrix} \tag{4.23}$$

The appropriate averaged values of the flow variables $\tilde{\rho}$, $\tilde{U}_x (= \tilde{u}_1)$, $\tilde{U}_y (= \tilde{u}_2)$, $\tilde{U}_z (= \tilde{u}_3)$, $\tilde{\rho}_i$, $\tilde{h}_0$, $\tilde{\Psi}_i$ with $i = 1, 2$ for number of species in the mixture are defined as:

$$\begin{aligned} \tilde{\rho} &= \sqrt{\rho_r \rho_l} \\ \tilde{U}_x &= \frac{\langle U_x \sqrt{\rho} \rangle}{\langle \sqrt{\rho} \rangle} \end{aligned}$$

$$
\begin{aligned}
\tilde{U}_y &= \frac{\langle U_y\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{U}_z &= \frac{\langle U_z\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{\rho}_i &= \frac{\langle \frac{\rho_i}{\rho}\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{h}_0 &= \frac{\langle h_0\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{\Psi}_i &= \frac{R_i\tilde{T}}{\tilde{\gamma}-1} - \tilde{e}_i + \frac{\tilde{q}^2}{2}
\end{aligned} \tag{4.24}
$$

where:

$$
\begin{aligned}
\tilde{T} &= \frac{\langle T\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{e}_i &= \frac{\langle e_i\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{q}^2 &= \tilde{U}_x^2 + \tilde{U}_y^2 + \tilde{U}_z^2
\end{aligned} \tag{4.25}
$$

Also some thermodynamic properties and speed of sound are:

$$
\begin{aligned}
\tilde{\gamma}-1 &= \frac{\tilde{R}}{\tilde{C}_v^*} \\
\tilde{R} &= \sum_{i=1}^{2}\tilde{\rho}_i R_i = \frac{\langle\tilde{R}\sqrt{\rho}\rangle}{\langle\sqrt{\rho}\rangle} \\
\tilde{C}_v^* &= \sum_{i=1}^{2}\tilde{\rho}_i C_{vi}^* \\
C_{vi}^* &= \frac{1}{[T]}\int_{T_l}^{T_r}\tilde{C}_{vi}\mathrm{d}T \\
\tilde{a}^2 &= (\tilde{\gamma}-1)\Big(\tilde{h}_0 - \frac{\tilde{q}^2}{2} + \tilde{C}_v^*\tilde{T} - \sum_{i=1}^{2}\tilde{\rho}_i\tilde{e}_i\Big)
\end{aligned} \tag{4.26}
$$

In addition, an entropy fix method was proposed by Harten and Hyman [34] to the Roe flux approximation scheme. This method limits the minimum value of the wave speed to linear wave field, which is same as adding extra numerical dissipation to damp out

spurious oscillation, for the entropy condition to be satisfied. In applications, it is very useful especially when small instabilities happened around shock regions.

4.3.2 HLLC scheme

Harten, Lax and van Leer put forward the following approximate Riemann solver:

$$\tilde{U}(x,t) = \begin{cases} U_L & \text{if } \frac{x}{t} < S_L, \\ U^{HLL} & \text{if } S_L \leq \frac{x}{t} \leq S_R, \\ U_R & \text{if } \frac{x}{t} > S_R. \end{cases} \tag{4.27}$$

where U^{HLL} is the constant state vector and the speeds S_L and S_R are the fastest signal velocities perturbing the initial data state U_L and U_R respectively. Taken from Einfeldt [25]:

$$\begin{aligned} S_L &= min[\lambda_1(U_L), \lambda_1(U^{Roe})] \\ S_R &= max[\lambda_m(U^{Roe}), \lambda_m(U_R)] \end{aligned} \tag{4.28}$$

with $\lambda_1(U^{Roe})$ and $\lambda_m(U^{Roe})$ being the smallest and largest eigenvalues of the Roe matrix. This approximate solution of the Riemann problem is called the HLL Riemann solver. The approximation consists of just three constant states separated by two waves. Then the corresponding HLL intercell flux for the approximate Godunov method is given as:

$$\mathbf{F}^{HLL} = \begin{cases} \mathbf{F}_L & \text{if } S_L > 0, \\ \frac{S_R\mathbf{F}_L - S_L\mathbf{F}_R + S_L S_R(U_R - U_L)}{S_R - S_L} & \text{if } S_L \leq 0 \leq S_R, \\ \mathbf{F}_R & \text{if } S_R < 0. \end{cases} \tag{4.29}$$

The HLLC scheme [52] is a modification of the HLL scheme, whereby the missing contact and shear waves in the Euler equations are restored. The HLLC approximate Riemann solver is given as follows:

$$\mathbf{F}^{HLLC} = \begin{cases} \mathbf{F}_L & \text{if } S_L > 0, \\ \mathbf{F}(U_L^*) & \text{if } S_L \leq 0 < S_M, \\ \mathbf{F}(U_R^*) & \text{if } S_M \leq 0 \leq S_R, \\ \mathbf{F}_R & \text{if } S_R < 0. \end{cases} \tag{4.30}$$

where the intermediate states are given as:

$$\begin{aligned} U_L^* &= \begin{bmatrix} \rho_{1L}^* \\ \rho_{2L}^* \\ (\rho u)_L^* \\ (\rho v)_L^* \\ (\rho w)_L^* \\ (\rho e_0)_L^* \end{bmatrix} = \Omega_L \begin{bmatrix} \rho_{1L}(S_L - q_L) \\ \rho_{2L}(S_L - q_L) \\ (S_L - q_L)(\rho u)_L + (p^* - p_L)n_x \\ (S_L - q_L)(\rho v)_L + (p^* - p_L)n_y \\ (S_L - q_L)(\rho w)_L + (p^* - p_L)n_z \\ (S_L - q_L)(\rho e_0)_L - p_L q_L + p^* S_M \end{bmatrix} \\ U_R^* &= \begin{bmatrix} \rho_{1R}^* \\ \rho_{2R}^* \\ (\rho u)_R^* \\ (\rho v)_R^* \\ (\rho w)_R^* \\ (\rho e_0)_R^* \end{bmatrix} = \Omega_R \begin{bmatrix} \rho_{1R}(S_R - q_R) \\ \rho_{2R}(S_R - q_R) \\ (S_R - q_R)(\rho u)_R + (p^* - p_R)n_x \\ (S_R - q_R)(\rho v)_R + (p^* - p_R)n_y \\ (S_R - q_R)(\rho w)_R + (p^* - p_R)n_z \\ (S_R - q_R)(\rho e_0)_R - p_R q_R + p^* S_M \end{bmatrix} \end{aligned} \tag{4.31}$$

and the intermediate fluxes are given:

$$\mathbf{F}_L^* = \mathbf{F}(U_L^*) = \begin{bmatrix} \rho_{1L}^* S_M \\ \rho_{2L}^* S_M \\ (\rho u)_L^* S_M + p^* n_x \\ (\rho v)_L^* S_M + p^* n_y \\ (\rho w)_L^* S_M + p^* n_z \\ ((\rho e_0)_L^* + p^*) S_M \end{bmatrix}$$

$$\mathbf{F}_R^* = \mathbf{F}(U_R^*) = \begin{bmatrix} \rho_{1R}^* S_M \\ \rho_{2R}^* S_M \\ (\rho u)_R^* S_M + p^* n_x \\ (\rho v)_R^* S_M + p^* n_y \\ (\rho w)_R^* S_M + p^* n_z \\ ((\rho e_0)_R^* + p^*) S_M \end{bmatrix} \tag{4.32}$$

where following parameters are defined:

$$\Omega_L = (S_L - S_M)^{-1}, \quad \Omega_R = (S_R - S_M)^{-1},$$

$$p^* = \rho_L(q_L - S_L)(q_L - S_M) + p_L = \rho_R(q_R - S_R)(q_R - S_M) + p_R. \tag{4.33}$$

and $q = un_x + vn_y + wn_z$, with $[n_x, n_y, n_z]^T$ being the unit vector normal to the interface. Moreover, $\rho_L = \rho_{1L} + \rho_{2L}$ and $\rho_R = \rho_{1R} + \rho_{2R}$ are relations for species density. S_M is taken from Batten [11] as:

$$S_M = \frac{\rho_R q_R(S_R - q_R) - \rho_L q_L(S_L - q_L) + p_L - p_R}{\rho_R(S_R - q_R) - \rho_L(S_L - q_L)} \tag{4.34}$$

4.3.3 HLLE scheme

Motivated by the Roe eigenvalues Einfeldt [25] proposed the alternative estimates:

$$S_L = \bar{u} - \bar{d}, \quad S_R = \bar{u} + \bar{d}. \tag{4.35}$$

for his HLLE solver, where $\bar{u}$ is numerical approximation of the velocity at the contact discontinuity, and both $\bar{u}$ and $\bar{d}$ are given here:

$$\begin{aligned} \bar{u} &= \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \\ \bar{d}^2 &= \frac{\sqrt{\rho_L}a_L^2 + \sqrt{\rho_R}a_R^2}{\sqrt{\rho_L} + \sqrt{\rho_R}} + \eta_2(u_R - u_L)^2, \\ \eta_2 &= \frac{1}{2}\frac{\sqrt{\rho_L}\sqrt{\rho_R}}{(\sqrt{\rho_L} + \sqrt{\rho_R})^2}. \end{aligned} \tag{4.36}$$

These wave speed estimates are reported to lead to effective and robust Godunov-type scheme. The formulas for S_L, S_R in Equation (4.35) are used in Equation (4.30) to create the HLLE flux equations.

4.3.4 VL-FVS scheme

The van Leer Flux Vector Splitting method [54] defines the positive and negative fluxes as functions of the local Mach number. They are given by:

$$\mathbf{F}(U_L, U_R) = \mathbf{F}^+(U_L) + \mathbf{F}^-(U_R) \tag{4.37}$$

where

$$\mathbf{F}^+(U) = \begin{cases} \mathbf{F}(U) & \text{if } u \geq a, \\ 0 & \text{if } u \leq -a, \\ \left\{ \begin{array}{l} \frac{1}{4}\rho a(u/a+1)^2 \equiv F_1^+ \\ F_1^+((\gamma-1)u+2a)/\gamma \\ F_1^+ v \\ F_1^+ w \\ F_1^+(\frac{((\gamma-1)u+2a)^2}{2(\gamma^2-1)} + \frac{v^2}{2} + \frac{w^2}{2}) \end{array} \right\} & \text{otherwise.} \end{cases} \tag{4.38}$$

and

$$\mathbf{F}^-(U) = \mathbf{F}(U) - \mathbf{F}^+(U) \tag{4.39}$$

The scheme was originally a simple way to implement upwind differencing and can produce steady shock profiles. But it leads to numerical diffusion of contact discontinuities at rest.

4.4 Computing diffusive flux

Since the second order derivatives with DG methods have been rewritten as a system of first order equations in Equation (4.15), then some techniques [2] can be used to approximate the diffusive flux contribution based on this form. Bassi and Rebay's BR2 method [8], and Cockburn and Shu's LDG method [22] are implemented in this research, and are analyzed in detail next.

4.4.1 Bassi-Rebay method (BR2)

In Equation (4.15), the convective flux have been computed numerically from Equation (4.6) and Equation (4.7) so they are just symbolically represented by (A) and (B) for simplicity. The key issue now is numerically solving the cell interface integrals of the system equations, which now can be written as:

$$\begin{aligned}
\int_{\Omega_h} \mathbf{g}_h \cdot \mathbf{Z}_h \mathrm{d}\Omega &= -\int_{\Omega_h} U_h \nabla \cdot \mathbf{g}_h \mathrm{d}\Omega + \int_{\Sigma_h} \widehat{U}_h \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma \\
\int_{\Omega_h} \frac{\partial U_h}{\partial t} \psi_h \mathrm{d}\Omega &- (A) + (B) - \int_{\Omega_h} [\mathscr{A}_h(U_h)\mathbf{Z}_h] \cdot \nabla \psi_h \mathrm{d}\Omega \\
&+ \int_{\Sigma_h} \left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right] \cdot \mathbf{n} \psi_h \mathrm{d}\sigma + \int_{\Omega_h} \mathbf{S}(U_h, \mathbf{Z}_h)\psi_h \mathrm{d}\Omega = 0 \quad (4.40)
\end{aligned}$$

Because U_h and $\mathscr{A}_h(U_h)\mathbf{Z}_h$ have two different values at each element boundary Σ_h, then we need to change their format first, and compute them subsequently. The face integrals from the above equations are listed as:

$$\begin{aligned}
\int_{\Sigma_h} \widehat{U}_h \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma &= \int_{\Sigma_h} \widehat{U}_h [(\mathbf{g}_h \cdot \mathbf{n})^- + (\mathbf{g}_h \cdot \mathbf{n})^+] \mathrm{d}\sigma + \int_{\Gamma_h} U_h^b \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma \\
\int_{\Sigma_h} \left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right] \cdot \mathbf{n} \psi_h \mathrm{d}\sigma &= \int_{\Sigma_h} [\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h] \cdot [(\psi_h \mathbf{n})^- + (\psi_h \mathbf{n})^+] \mathrm{d}\sigma \\
&\quad + \int_{\Gamma_h} [\mathscr{A}_h(U_h)\mathbf{Z}_h]^b \cdot \mathbf{n} \psi_h \mathrm{d}\sigma \quad (4.41)
\end{aligned}$$

where U_h^b and $[\mathscr{A}_h(U_h)\mathbf{Z}_h]^b$ are the values of U_h and $[\mathscr{A}_h(U_h)\mathbf{Z}_h]$ at element boundary. We introduce a "vector jump operator" $\mathscr{J}$:

$$\mathscr{J} x = x^- \mathbf{n}^- + x^+ \mathbf{n}^+, \quad \mathscr{J} \cdot \mathbf{y} = \mathbf{y}^- \cdot \mathbf{n}^- + \mathbf{y}^+ \cdot \mathbf{n}^+ \quad (4.42)$$

then we have:

$$\int_{\Sigma_h} \widehat{U}_h \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma = \int_{\Sigma_h} \widehat{U}_h \mathscr{J} \cdot \mathbf{g}_h \mathrm{d}\sigma + \int_{\Gamma_h} U_h^b \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma$$

$$\int_{\Sigma_h}\left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right]\cdot\mathbf{n}\psi_h\mathrm{d}\sigma=\int_{\Sigma_h}[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h]\cdot\mathscr{J}\,\psi_h\mathrm{d}\sigma + \int_{\Gamma_h}[\mathscr{A}_h(U_h)\mathbf{Z}_h]^b\cdot\mathbf{n}\psi_h\mathrm{d}\sigma \tag{4.43}$$

so Equation (4.40) becomes:

$$\int_{\Omega_h}\mathbf{g}_h\cdot\mathbf{Z}_h\mathrm{d}\Omega=-\int_{\Omega_h}U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega+\int_{\Sigma_h}\widehat{U}_h\mathscr{J}\cdot\mathbf{g}_h\mathrm{d}\sigma+\int_{\Gamma_h}U_h^b\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma \tag{4.44}$$

$$\int_{\Omega_h}\frac{\partial U_h}{\partial t}\psi_h\mathrm{d}\Omega-(A)+(B)-\int_{\Omega_h}[\mathscr{A}_h(U_h)\mathbf{Z}_h]\cdot\nabla\psi_h\mathrm{d}\Omega+\int_{\Sigma_h}[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h]\cdot\mathscr{J}\,\psi_h\mathrm{d}\sigma + \int_{\Gamma_h}[\mathscr{A}_h(U_h)\mathbf{Z}_h]^b\cdot\mathbf{n}\psi_h\mathrm{d}\sigma+\int_{\Omega_h}\mathbf{S}(U_h,\mathbf{Z}_h)\psi_h\mathrm{d}\Omega=0 \tag{4.45}$$

In the following we focus on the parts of $\widehat{(\cdot)}$ from the above discretized equations, as $\widehat{U}_h$ and $\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h$.

First, choose $\widehat{U}_h=\{U_h\}$, which is:

$$\{(\cdot)\}=\begin{cases}\frac{1}{2}[(\,)^-+(\,)^+] & \text{on } \Sigma_h\\ (\quad) & \text{on } \Gamma_h\end{cases} \tag{4.46}$$

so Equation (4.44) changes to:

$$\int_{\Omega_h}\mathbf{g}_h\cdot\mathbf{Z}_h\mathrm{d}\Omega=-\int_{\Omega_h}U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega+\int_{\Sigma_h}\{U_h\}\,\mathscr{J}\cdot\mathbf{g}_h\mathrm{d}\sigma+\int_{\Gamma_h}U_h^b\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma \tag{4.47}$$

Using the identity formula derived from Bassi and Rebay's paper [8]:

$$\begin{aligned}\int_{\Omega_h}(U_h\nabla\cdot\mathbf{g}_h+\mathbf{g}_h\cdot\nabla U_h)\mathrm{d}\Omega&=\int_{\Sigma_h}[\{U_h\}\,\mathscr{J}\cdot\mathbf{g}_h+\{\mathbf{g}_h\}\cdot\mathscr{J}U_h]\mathrm{d}\sigma\\&\quad+\int_{\Gamma_h}U_h\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma\\-\int_{\Omega_h}U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega&=\int_{\Omega_h}\mathbf{g}_h\cdot\nabla U_h\mathrm{d}\Omega-\int_{\Sigma_h}[\{U_h\}\,\mathscr{J}\cdot\mathbf{g}_h+\{\mathbf{g}_h\}\cdot\mathscr{J}U_h]\mathrm{d}\sigma\\&\quad-\int_{\Gamma_h}U_h\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma\end{aligned} \tag{4.48}$$

substitute Equation (4.48) into Equation (4.47) results in:

$$\int_{\Omega_h} \mathbf{g}_h \cdot \mathbf{Z}_h \mathrm{d}\Omega - \int_{\Omega_h} \mathbf{g}_h \cdot \nabla U_h \mathrm{d}\Omega + \int_{\Sigma_h} \{\mathbf{g}_h\} \cdot \mathscr{J} U_h \mathrm{d}\sigma + \int_{\Gamma_h} (U_h - U_h^b) \mathbf{g}_h \cdot \mathbf{n} \mathrm{d}\sigma = 0 \quad (4.49)$$

We define:

$$\mathscr{J}^0 x = \begin{cases} (x - x^b)\mathbf{n} & \text{on } \Gamma_h \\ x^- \mathbf{n}^- + x^+ \mathbf{n}^+ & \text{on } \Sigma_h \end{cases}$$

and $\Sigma_h^0 = \Sigma_h \bigcup \Gamma_h$, finally Equation (4.44) becomes:

$$\int_{\Omega_h} \mathbf{g}_h \cdot \mathbf{Z}_h \mathrm{d}\Omega - \int_{\Omega_h} \mathbf{g}_h \cdot \nabla U_h \mathrm{d}\Omega + \int_{\Sigma_h^0} \{\mathbf{g}_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma = 0 \quad (4.50)$$

Second, choose:

$$\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h = \{\mathscr{A}_h(U_h)\mathbf{Z}_h\} \quad (4.51)$$

then Equation (4.45) changes to:

$$\begin{aligned} &\int_{\Omega_h} \frac{\partial U_h}{\partial t} \psi_h \mathrm{d}\Omega - (A) + (B) - \int_{\Omega_h} [\mathscr{A}_h(U_h)\mathbf{Z}_h] \cdot \nabla \psi_h \mathrm{d}\Omega \\ &+ \int_{\Sigma_h^0} \{\mathscr{A}_h(U_h)\mathbf{Z}_h\} \cdot \mathscr{J}^0 \psi_h \mathrm{d}\sigma + \int_{\Omega_h} \mathbf{S}(U_h, \mathbf{Z}_h)\psi_h \mathrm{d}\Omega = 0 \end{aligned} \quad (4.52)$$

Now we introduce BR1 scheme, set $\mathbf{g}_h = \nabla \psi_h$ so that Equation (4.50) is:

$$\int_{\Omega_h} \nabla \psi_h \cdot \mathbf{Z}_h \mathrm{d}\Omega - \int_{\Omega_h} \nabla \psi_h \cdot \nabla U_h \mathrm{d}\Omega + \int_{\Sigma_h^0} \{\nabla \psi_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma = 0 \quad (4.53)$$

which is equivalent to:

$$\int_{\Omega_h} \nabla \psi_h \cdot (\mathbf{Z}_h - \nabla U_h) \mathrm{d}\Omega = -\int_{\Sigma_h^0} \{\nabla \psi_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma \quad (4.54)$$

Now introduce "lift" operator as:

$$\int_{\Omega_h} \nabla \psi_h \cdot \mathbf{R}_h(\mathscr{J}^0 U_h) \mathrm{d}\Omega = -\int_{\Sigma_h^0} \{\nabla \psi_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma \quad (4.55)$$

by virtue of the global lift operator $\mathbf{R}_h$ in Equation (4.55), we can express the variable $\mathbf{Z}_h$ from Equation (4.54) in weak sense as:

$$\int_{\Omega_h} \nabla\psi_h \cdot (\mathbf{Z}_h - \nabla U_h)\mathrm{d}\Omega = \int_{\Omega_h} \nabla\psi_h \cdot \mathbf{R}_h(\mathscr{J}^0 U_h)\mathrm{d}\Omega \tag{4.56}$$

Because ψ_h and $\nabla\psi_h$ are arbitrary functions, the above relation implies that:

$$\mathbf{Z}_h - \nabla U_h = \mathbf{R}_h(\mathscr{J}^0 U_h) \quad \Longrightarrow \quad \mathbf{Z}_h = \nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h) \tag{4.57}$$

further we know:

$$\begin{aligned}
&\widehat{\mathbf{Z}_h} = \{\mathbf{Z}_h\} = \{\nabla U_h\} + \left\{\mathbf{R}_h(\mathscr{J}^0 U_h)\right\} \\
&\int_{\Omega_h} [\mathscr{A}_h(U_h)\mathbf{Z}_h] \cdot \nabla\psi_h \mathrm{d}\Omega = \int_{\Omega_h} \left[\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right] \cdot \nabla\psi_h \mathrm{d}\Omega \\
&\int_{\Sigma_h^0} \{\mathscr{A}_h(U_h)\mathbf{Z}_h\} \cdot \mathscr{J}^0\psi_h \mathrm{d}\sigma = \int_{\Sigma_h^0} \left\{\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right\} \cdot \mathscr{J}^0\psi_h \mathrm{d}\sigma
\end{aligned} \tag{4.58}$$

substitute these relations to the corresponding parts in Equation (4.52) and we get:

$$\begin{aligned}
\int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega \quad - \quad & (A) + (B) - \int_{\Omega_h} \left[\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right] \cdot \nabla\psi_h \mathrm{d}\Omega \\
+ \quad & \int_{\Sigma_h^0} \left\{\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right\} \cdot \mathscr{J}^0\psi_h \mathrm{d}\sigma \\
+ \quad & \int_{\Omega_h} \mathbf{S}(U_h, \nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\psi_h \mathrm{d}\Omega = 0
\end{aligned} \tag{4.59}$$

Now we introduce the compact BR2 scheme. Define the local lift operator on two neighboring elements:

$$\int_{\Omega_h^\sigma} \mathbf{g}_h \cdot \mathbf{r}_h^\sigma(\mathscr{J}^0 U_h)\mathrm{d}\Omega = -\int_\sigma \{\mathbf{g}_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma \tag{4.60}$$

Here Ω_h^σ is the union of the two neighboring elements, $\mathbf{r}_h^\sigma(\mathscr{J}^0 U_h)$ is a local lift operator and σ stands for interface of these two elements. The global lift operator expression can be rewritten as:

$$\begin{aligned}\int_{\Omega_h} \mathbf{g}_h \cdot \mathbf{R}_h(\mathscr{J}^0 U_h)\mathrm{d}\Omega &= -\int_{\Sigma_h^0} \{\mathbf{g}_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma = -\sum_{\sigma\in\Sigma_h^0} \int_\sigma \{\mathbf{g}_h\} \cdot \mathscr{J}^0 U_h \mathrm{d}\sigma \\ &= \sum_{\sigma\in\Sigma_h^0} \int_{\Omega_h^\sigma} \mathbf{g}_h \cdot \mathbf{r}_h^\sigma(\mathscr{J}^0 U_h)\mathrm{d}\Omega \end{aligned} \tag{4.61}$$

therefore we get the important relation:

$$\mathbf{R}_h(\mathscr{J}^0 U_h) = \sum_{\sigma\in\Sigma_h^0} \mathbf{r}_h^\sigma(\mathscr{J}^0 U_h) \tag{4.62}$$

which means the global lift operator can be represented as the summation of the local lift operator in neighboring two cells. This is a very compact form to compute the global lift operator, making the computational stencil small. Substitute $\mathbf{R}_h(\mathscr{J}^0 U_h)$ into the Equation (4.59), where the cell integration does not change, but the face integration needs to be changed, then:

$$\begin{aligned}\int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega &- (A) + (B) - \int_{\Omega_h} \left[\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right] \cdot \nabla\psi_h \mathrm{d}\Omega \\ &+ \sum_{\sigma\in\Sigma_h^0} \int_{\Sigma_h^0} \left\{\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{r}_h^\sigma(\mathscr{J}^0 U_h))\right\} \cdot \mathscr{J}^0\psi_h \mathrm{d}\sigma \\ &+ \int_{\Omega_h} \mathbf{S}(U_h, \nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\psi_h \mathrm{d}\Omega = 0 \end{aligned} \tag{4.63}$$

Finally, the DG discretization of RANS equations are expressed:

$$\begin{aligned}&\int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega - \underbrace{\int_{\Omega_h} \mathbf{F}(U_h) \cdot \nabla\psi_h \mathrm{d}\Omega}_{(A)} + \underbrace{\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)(\psi_h^- - \psi_h^+)\mathrm{d}\sigma}_{(B)} \\ &+ \underbrace{\int_{\Gamma_h} \mathbf{F}(U_h^b) \cdot \mathbf{n}\psi_h \mathrm{d}\sigma}_{(B)} - \underbrace{\int_{\Omega_h} \left[\mathscr{A}_h(U_h)(\nabla U_h + \mathbf{R}_h(\mathscr{J}^0 U_h))\right] \cdot \nabla\psi_h \mathrm{d}\Omega}_{(C)}\end{aligned}$$

$$+\underbrace{\sum_{\sigma\in\Sigma_h^0}\int_{\Sigma_h^0}\left\{\mathscr{A}_h(U_h)(\nabla U_h+\mathbf{r}_h^\sigma(\mathscr{J}^0U_h))\right\}\cdot\mathscr{J}^0\psi_h\mathrm{d}\sigma}_{(D)}$$
$$+\underbrace{\int_{\Omega_h}\mathbf{S}(U_h,\nabla U_h+\mathbf{R}_h(\mathscr{J}^0U_h))\psi_h\mathrm{d}\Omega}_{(E)}=0 \tag{4.64}$$

Here some basic considerations about how to compute each flux component in the above equations are introduced. Part(A) and (B) have been discussed in the previous section. In Part(C), as we know diffusive flux is represented as $\mathscr{A}_h(U_h)\nabla U_h$ which is a linear function of ∇U_h, then directly replace ∇U_h with $\nabla U_h+\mathbf{R}_h(\mathscr{J}^0U_h)$ in this expression $\mathscr{A}_h(U_h)\nabla U_h$, and take the same cell integration procedure as in Part (A). In Part (D), $\mathbf{r}_h^\sigma(\mathscr{J}^0U_h)$ is computed at each interface, then directly replace ∇U_h with $\nabla U_h+\mathbf{r}_h^\sigma(\mathscr{J}^0U_h)$ in face diffusive flux expression $\mathscr{A}_h(U_h)\nabla U_h$, after that all face flux integrations are summed up. In Part(E), directly replace ∇U_h with $\nabla U_h+\mathbf{R}_h(\mathscr{J}^0U_h)$ in source term, then same cell integration method applies. Simply speaking, we can write the DG discretized equations as the symbolic expression:

$$\int_{\Omega_h}\frac{\partial U_h}{\partial t}\psi_h\mathrm{d}\Omega-(A)+(B)-(C)+(D)+(E)=0 \tag{4.65}$$

4.4.2 Local Discontinuous Galerkin method

Another approach to computing the diffusive flux is introduced here. Beginning with Equations (4.15) and (4.40), the discretized RANS equations are written as:

$$\begin{aligned}\int_{\Omega_h}\mathbf{g}_h\cdot\mathbf{Z}_h\mathrm{d}\Omega &+ \int_{\Omega_h}U_h\nabla\cdot\mathbf{g}_h\mathrm{d}\Omega-\int_{\Sigma_h}\widehat{u}_h\mathbf{g}_h\cdot\mathbf{n}\mathrm{d}\sigma=0\\ \int_{\Omega_h}\frac{\partial U_h}{\partial t}\psi_h\mathrm{d}\Omega &- (A)+(B)-\int_{\Omega_h}\sigma_h\cdot\nabla\psi_h\mathrm{d}\Omega\\ &+ \int_{\Sigma_h}\widehat{\sigma_h}\cdot\mathbf{n}\psi_h\mathrm{d}\sigma+\int_{\Omega_h}\mathbf{S}(U_h,\mathbf{Z}_h)\psi_h\mathrm{d}\Omega=0\end{aligned} \tag{4.66}$$

in this formula $\sigma_h = \mathscr{A}_h(U_h)\mathbf{Z}_h$ is used to represent the diffusive flux and $u_h = U_h$ is used to represent conservative variables. Two additional numerical fluxes $\hat{\sigma}_h = \left[\mathscr{A}_h(\widehat{U_h})\mathbf{Z}_h\right]$ and $\widehat{u}_h = \widehat{U}_h$ appear in the face integrals of above equations, which are approximations of numerical traces of σ_h and U_h on element Ω_e boundaries. These numerical fluxes need to be specified by LDG method as follows. Consider two adjacent elements Ω_l and Ω_r sharing interface $\Gamma = \Omega_l \cap \Omega_r$. Now the jump $\| \cdot \|$ and mean $\{ \cdot \}$ operators are defined along the interface Γ and the exterior boundary:

$$\|u\mathbf{n}\| = \begin{cases} u_l\mathbf{n}_l + u_r\mathbf{n}_r & \text{on } \Gamma \\ u\mathbf{n} & \text{on } \partial\Omega \end{cases} \qquad \{u\} = \begin{cases} \frac{1}{2}(u_l + u_r) & \text{on } \Gamma \\ u & \text{on } \partial\Omega \end{cases} \quad \text{for scalars}$$

$$\|\sigma \cdot \mathbf{n}\| = \begin{cases} \sigma_l \cdot \mathbf{n}_l + \sigma_r \cdot \mathbf{n}_r & \text{on } \Gamma \\ \sigma \cdot \mathbf{n} & \text{on } \partial\Omega \end{cases} \qquad \{\sigma\} = \begin{cases} \frac{1}{2}(\sigma_l + \sigma_r) & \text{on } \Gamma \\ \sigma & \text{on } \partial\Omega \end{cases} \quad \text{for vectors} \tag{4.67}$$

The numerical fluxes $\hat{\sigma}_h$ and $\widehat{u}_h$ are defined as:

$$\begin{aligned} \hat{\sigma}_h &= \{\sigma_h\} + \mathbf{C_{12}}\|\sigma_h \cdot \mathbf{n}\| - C_{11}\|u_h\mathbf{n}\| \\ \hat{u}_h &= \{u_h\} - \mathbf{C_{12}} \cdot \|u_h\mathbf{n}\| \end{aligned} \tag{4.68}$$

for interior faces and:

$$\begin{aligned} \hat{\sigma}_h &= \sigma_h - C_{11}(u_h - u_D)\mathbf{n} \\ \hat{u}_h &= u_h \end{aligned} \tag{4.69}$$

for boundary faces. We find that two parameters need to be specified in this LDG method. C_{11} is a non-negative constant and it acts as a penalty parameter. $\mathbf{C_{12}}$ is a vector, which

is determined for each interior face. In this study, we chose $C_{11} = 0$ and $\mathbf{C_{12}} = \pm 0.5\mathbf{n}$, and the solver appears to be stable in numerical experiments. The first equation in Equation (4.66) is solved locally, with face numerical flux $\widehat{U}_h$ computed from above to figure out the value of $\mathbf{Z}_h$. Then substitute this $\mathbf{Z}_h$ to the second equation of Equation (4.66) to compute updated value of fluxes, including face numerical flux $\hat{\sigma}_h$. Finally the equation can be integrated by time to find the solution.

4.5 Boundary conditions

In order to facilitate the implementation of the numerical method, all boundary conditions (BC) are imposed in a weak manner. From Figure 4.1, the boundary face separates the interior and exterior regions, where the interior domain is the computational domain at which the solution is updated at each time iteration, and the exterior domain is a physical boundary domain in general, including inflow, outflow, wall and farfield domains. In the implementation, the solution state at boundary $U_h^b(U_h^i, U_h^e)$ that is required to compute the convective flux and diffusive flux, is a function of the interior state U^i and the specified bounday value U^e which will be discussed in this section. Moreover, in the implicit time iteration an explicit Jacobi matrix from the dependency of boundary conditions U_h^b to known solutions U_h^i is required, therefore this part is also included in the boundary condition implementation of this section.

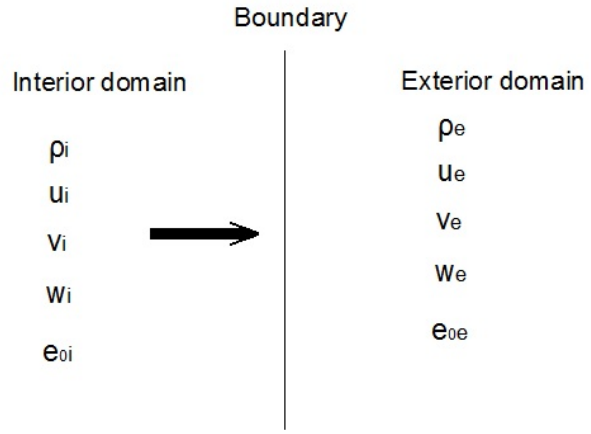

Figure 4.1

Illustration of boundary domains

4.5.1 Dirichlet BC

For dirichlet boundary condition, the solution variables need to be specified as:

$$\rho_1^b = \rho_{1,given}, \quad \rho_2^b = \rho_{2,given}, \quad u_1^b = u_{1,given}, \quad u_2^b = u_{2,given},$$
$$u_3^b = u_{3,given}, \quad p^b = p_{given}, \quad k^b = k_{given}, \quad \omega^b = \omega_{given}. \tag{4.70}$$

The convective and diffusive flux at the boundary are computed from these values. The implicit time Jacobian matrix is:

$$J = \frac{dU^b(U^i, U^e)}{dU^i} \tag{4.71}$$

and in this case $J \equiv 0$.

4.5.2 Reflecting BC

For inviscid flow at wall, the reflecting boundary condition is utilized, and the solution variables are computed as:

$$\begin{aligned}
&\rho_1^b = \rho_1^i, \qquad \rho_2^b = \rho_2^i, \qquad u_1^b = u_1^i - 2Un \cdot n_x, \\
&u_2^b = u_2^i - 2Un \cdot n_y, \qquad u_3^b = u_3^i - 2Un \cdot n_z, \qquad p^b = p^i, \\
&k^b = k^i, \qquad \omega^b = \omega^i.
\end{aligned} \tag{4.72}$$

where $Un = u_1^i \cdot n_x + u_2^i \cdot n_y + u_3^i \cdot n_z$, the fluxes at boundary are computed from these specified variables. The implicit time Jacobian matrix can be computed easily.

4.5.3 No-slip BC

For viscous flow, the no-slip wall boundary condition is applied, and the solution variables are computed as:

$$\begin{aligned}
&\rho_1^b = \rho_1^i, \qquad \rho_2^b = \rho_2^i, \qquad u_1^b = 0, \\
&u_2^b = 0, \qquad u_3^b = 0, \qquad p^b = p^i, \\
&\kappa^b = \kappa_{wall}, \qquad \omega^b = \omega_{wall}.
\end{aligned} \tag{4.73}$$

where turbulent quantities κ_{wall} and ω_{wall} are specified from Menter [42] as:

$$\begin{aligned}
\kappa_{wall} &= 0 \\
\omega_{wall} &= \frac{60\mu}{\rho\beta_1(\Delta y_1)^2}
\end{aligned} \tag{4.74}$$

In this formula Δy_1 is the distance of the first grid point from the wall, which in implementation is equal to the height of the wall boundary triangle or quadrilateral. Also $\beta_1 = 0.075$

is known constant in ω_{wall} term. Furthermore, if adiabatic walls are assumed, then heat flux at the wall and normal temperature gradient at the wall are:

$$\left(\frac{\partial T}{\partial \mathbf{n}}\right)^b = (\nabla T)^b \cdot \frac{\mathbf{n}}{|\mathbf{n}|} = 0 \tag{4.75}$$

If isothermal walls are assumed, then $T^b = T_{wall}$ specified at wall so that other variables (ρ, p) are derived from this relation. Finally, the fluxes at boundary are computed from these specified variables. The implicit time Jacobian matrix can be computed directly.

4.5.4 Extrapolation BC

At supersonic outflow boundary, all variables are extrapolated from interior. First order extrapolation boundary condition is used in this study, and the solution variables are computed as:

$$\begin{aligned} &\rho_1^b = \rho_1^i, \qquad \rho_2^b = \rho_2^i, \qquad u_1^b = u_1^i, \\ &u_2^b = u_2^i, \qquad u_3^b = u_3^i, \qquad p^b = p^i, \\ &k^b = k^i, \qquad \omega^b = \omega^i. \end{aligned} \tag{4.76}$$

the fluxes at boundary are computed from these specified variables. The implicit time Jacobian matrix is the identity matrix.

CHAPTER 5

TIME INTEGRATION METHOD

The DG discretization of the governing equations in space leads to a system of ordinary differential equations (ODEs) with respect to time in the form of:

$$\mathbf{M}\frac{dU}{dt} = \mathbf{R}(U) \tag{5.1}$$

where $\mathbf{M}$ is the global mass matrix which is block-diagonal overall. U is a global vector of degrees of freedom for each conserved variable, and $\mathbf{R}(U)$ is the global residual vector which is composed of numerical fluxes and source terms. Equation (5.1) is identical to the discretized Euler equations shown in Equation (4.5) or the discretized RANS equations introduced in Equation (4.65). The system Equation (5.1) is time dependent, and any numerical integration scheme applicable to ODEs can be used to solve it. The explicit methods are straightforward to implement, but the computational time step is limited. Therefore the convergence to a steady state could be very time consuming. This is true especially for viscous flow problems, where the required CFL number is lower and the time step is smaller in magnitude than in the case of inviscid applications. A typical explicit method is the Runge-Kutta multiple steps iteration approach, which is introduced in this chapter to solve both steady and unsteady problems. In order to overcome the difficulty of small time steps, other approaches will be used in the following. The fully implicit time integration

method can make use of large time steps, and a typical backward Euler implicit method with Newton's iterations will be given here. But in implicit methods, the Jacobian matrix is relatively large and complex and takes up memory. Another choice of time integration, with which the implementation of hp adaptive strategy in the DG framework can be fully exploited, is the semi-implicit p-multigrid acceleration strategy [9], which iterates between different solution degrees and smoothers are employed for implicit or explicit methods at each level. This time integration strategy combines the benefits of both explicit and implicit time integration methods and is the preferred choice for this study when large time steps are desirable for steady state problems.

5.1 Explicit time integration

When the explicit approach is used for the temporal discretization, the right hand side (RHS) of Equation (5.1) is taken at the previous time level t_n:

$$\mathbf{M}\frac{dU}{dt} = \mathbf{R}(U^n) \tag{5.2}$$

The classical four-stage Runge-Kutta (RK4) method is adopted for explicit time integration. The idea of this method is to evaluate the RHS of Equation (5.2) at several values of U in the time interval between $n\Delta t$ and $(n+1)\Delta t$ and to combine them in order to obtain a high-order approximation of U^{n+1}. The RK4 method can be summarized in the following:

$$\begin{aligned} U^{(0)} &= U^n, \quad k_0 = 0 \\ k_i &= \mathbf{M}^{-1}\mathbf{R}(U = U^n + \Delta t a_i k_{i-1}) \\ U^{(i)} &= U^{(i-1)} + \Delta t b_i k_i, \quad i = 1 \ldots m \end{aligned}$$

$$U^{(m)} = U^{n+1} \tag{5.3}$$

here Δt is the physical time step and a_i and b_i are constant coefficients which are shown here:

$$\begin{aligned} a &= (a_1, a_2, a_3, a_4)^T = (0, \frac{1}{2}, \frac{1}{2}, 1)^T \\ b &= (b_1, b_2, b_3, b_4)^T = (\frac{1}{6}, \frac{1}{3}, \frac{1}{3}, \frac{1}{6})^T \end{aligned} \tag{5.4}$$

The Runge-Kutta explicit time integration is used for computing both steady and unsteady problems. For steady flow simulation, the solution converges when the residual vector $\mathbf{R}$ approximates zero. For unsteady problems, the system equations are solved in time until the desired period of time has elapsed.

5.2 Implicit time integration

When a fully implicit approach is adopted, the RHS of Equation (5.1) is evaluated at the new time level t_{n+1}:

$$\mathbf{M}\frac{dU}{dt} = \mathbf{R}(U^{n+1}) \tag{5.5}$$

Here $\mathbf{M}$ is mass matrix, U is conservative variable polynomial, and $\mathbf{R}$ is the RHS residual part of discretized governing equations.

The implicit backward Euler time integration scheme is used in this study, and the discretized equations in time are given as:

$$\mathbf{M}\frac{\Delta U^n}{\Delta t} = \mathbf{R}^(U^{n+1}) \tag{5.6}$$

where

$$\Delta U^n = U^{n+1} - U^n \tag{5.7}$$

For steady flow problems, when the solution converges after some time, the global residual $\mathbf{R}(U)$ should drop to a given low value (for example 10^{-10}). For unsteady flow problems, the solution state after elapsed time t can be achieved by many time iterations. In addition, depending on the order of accuracy used, the global residual $\mathbf{R}(U)$ may not drop to such given value, and this is the inherent residual of this time iteration which cannot be eliminated through time iterations.

Solving the Equation (5.6) directly is difficult due to the nonlinear terms in the global residual function $\mathbf{R}(U^{n+1})$. A commonly accepted approach is to employ the Newton iterative method for the numerical solution of the nonlinear homogeneous equations, the Newton residual function is:

$$L(U^{n+1}) = \mathbf{M}(U^{n+1} - U^n) - \Delta t \cdot \mathbf{R}(U^{n+1}) = 0 \tag{5.8}$$

The Newton method proceeds by iteratively solving the equations as:

$$L'(U^{n+1,p})(U^{n+1,p+1} - U^{n+1,p}) = -L(U^{n+1,p}) \tag{5.9}$$

where the Newton iteration is initialized using the previous time-step value as $U^{n+1,p=0} = U^n$. Newton's method accommodates the linearized solution as an approximation to the original nonlinear system equations. After several steps, when the iteration scheme converges to U^{n+1}, the residual function $L(U^{n+1,p})$ should drop to a given low value (for example 10^{-10}). Also the Jacobian matrix $L'(U^{n+1,p})$ is given here:

$$L'(U^{n+1,p}) = \mathbf{M} - \Delta t \cdot \frac{\partial \mathbf{R}(U^{n+1})}{\partial U^{n+1}} \tag{5.10}$$

The derivation of $L'(U^{n+1,p})$ is straightforward but involved. The details of the flux Jacobian matrices are discussed in APPENDIX C. Finally, the Newton iteration scheme is written as:

$$(\frac{\mathbf{M}}{\Delta t} - \frac{\partial \mathbf{R}(U^{n+1})}{\partial U^{n+1}})(U^{n+1,p+1} - U^{n+1,p}) = -\frac{\mathbf{M}}{\Delta t}(U^{n+1,p} - U^n) + \mathbf{R}(U^{n+1}) \tag{5.11}$$

The linearized Equation (5.11) can be symbolically represented in the form:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{5.12}$$

where

$$\begin{aligned} \mathbf{A} &= \frac{\mathbf{M}}{\Delta t} - \frac{\partial \mathbf{R}(U^{n+1})}{\partial U^{n+1}}, \\ \mathbf{x} &= U^{n+1,p+1} - U^{n+1,p}, \\ \mathbf{b} &= -\frac{\mathbf{M}}{\Delta t}(U^{n+1,p} - U^n) + \mathbf{R}(U^{n+1}). \end{aligned} \tag{5.13}$$

The matrix $\mathbf{A}$ can be factored into a lower, an upper, and a diagonal block:

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U} \tag{5.14}$$

Now the PETSc [4] solver can be applied to the linearized system equations, given the lower, upper, diagonal blocks and RHS. When the solver converges with a residual drops to a specified criteria, the updated solution $U^{n+1} = U^{n+1,p+1}$ can be achieved.

5.3 Semi-implicit p-multigrid method

In the current research, the full multigrid (FMG) algorithm has been employed, see Figure 5.1. The coarser level solutions are exploited to obtain good initial guess to initialize

the computation on the finer grids. At each level, a number of pre-smoothing iterations are performed prior to restricting the solution to the next coarser level, while on the way back to finer levels, a number of post-smoothing iterations are performed after prolongation. Converging the solution fully on each level is not practical because the discretization error on the coarser level is usually above machine zero, so the solution is prolongated to the finer level when a residual-based criterion is satisfied [26].

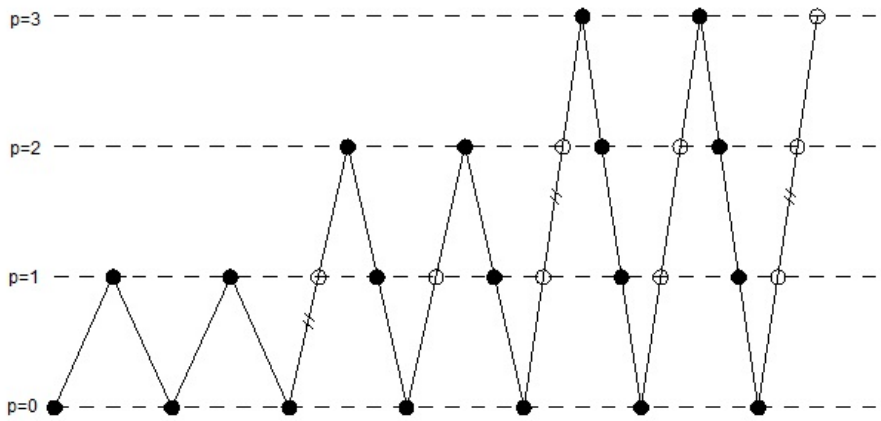

Figure 5.1

V-cycle full multigrid for $p = 3$ (•: pre-smoothing ○: post-smoothing)

5.3.1 P-multigrid algorithm

Let us illustrate the p-multigrid algorithm from the DG discretization of RANS equations in Equation (4.65). The general form can be written as:

$$\mathbf{A}^p(U^p) = \mathbf{f}^p \tag{5.15}$$

where:

$$\mathbf{A}^p(U^p) = \int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega - (A) + (B) - (C) + (D) + (E)$$

$$\mathbf{f}^p = mms \begin{cases} = 0 & \text{solving for solution} \\ \neq 0 & \text{for testing order of accuracy} \end{cases} \tag{5.16}$$

The term mms exists only when a manufactured solution method [43] is used to test the order of accuracy of p-multigrid solver for different solution orders. (The results of testing the order with manufactured solutions will be mentioned in the chapter of numerical results.) Let V^p be an approximation to the solution vector U^p and define the residual vector as:

$$\mathbf{r}^p(V^p) = \mathbf{f}^p - \mathbf{A}^p(V^p) \tag{5.17}$$

In the basic two-level multigrid method, the exact solution on the coarse level is used to correct the solution on the fine level. The correction is performed according to the following steps:

- restrict the solution and residual to the coarse level:

$$V_0^{p-1} = \tilde{\mathbf{I}}_p^{p-1} V^p, \quad \mathbf{r}^{p-1} = \mathbf{I}_p^{p-1}\mathbf{r}^p(V^p) \tag{5.18}$$

  where $\tilde{\mathbf{I}}_p^{p-1}$ and $\mathbf{I}_p^{p-1}$ are the solution and the residual restriction operators from level p to level $p-1$, respectively.

- compute the forcing term on the coarse level:

$$\mathbf{s}^{p-1} = \mathbf{A}^{p-1}(V_0^{p-1}) - \mathbf{r}^{p-1} \tag{5.19}$$

- solve problem on the coarse level:

$$\mathbf{A}^{p-1}(V^{p-1}) = \mathbf{f}^{p-1} + \mathbf{s}^{p-1} \tag{5.20}$$

- calculate the coarse grid error:

$$\mathbf{e}^{p-1} = V^{p-1} - V_0^{p-1} \tag{5.21}$$

- prolongate the coarse grid error so as to correct the fine level approximation:

$$V^p_{new} = V^p + \tilde{\mathbf{I}}^p_{p-1}\mathbf{e}^{p-1} \tag{5.22}$$

where $\tilde{\mathbf{I}}^p_{p-1}$ is the error prolongation operator, and V^p_{new} is the corrected value of V^p.

Following the method of Bassi and Rebay [9], the operators are defined next. In the framework of the DG method, if the basis functions are orthogonal and hierarchical, the computation of these multigrid operators is to be simple, thus the implementation of p-multigrid method is very appropriate for the DG method. The solution's restriction and error prolongation operators $\tilde{\mathbf{I}}^{p-1}_p$ and $\tilde{\mathbf{I}}^p_{p-1}$ are simply L^2 projections between the low-order and high-order spaces V^{p-1} and V^p:

$$\begin{aligned} \tilde{\mathbf{I}}^{p-1}_p &= (\mathbf{M}^{p-1})^{-1}\mathbf{M}^{p-1}_p \\ \tilde{\mathbf{I}}^p_{p-1} &= (\mathbf{M}^p)^{-1}(\mathbf{M}^{p-1}_p)^T \end{aligned} \tag{5.23}$$

here the matrices are:

$$\begin{aligned} \mathbf{M}^{p-1} &= [M_{ij}]^{p-1} = \int_\Omega \psi^{p-1}_i \psi^{p-1}_j \mathrm{d}\Omega \\ \mathbf{M}^{p-1}_p &= [M_{ij}]^{p-1}_p = \int_\Omega \psi^{p-1}_i \psi^{p}_j \mathrm{d}\Omega \end{aligned} \tag{5.24}$$

where ψ_i and ψ_j are solution polynomials, and i and j are the degrees of freedom of these polynomials.

An explicit expression of the residual restriction operator $\mathbf{I}^{p-1}_p$ can be obtained following the approach proposed by Fidkowski [26], which shows that in fact $\mathbf{I}^{p-1}_p = (\tilde{\mathbf{I}}^p_{p-1})^T$. The criterion used when the solver prolongates to the next finer level is described here. At the end of a V-cycle, the current residual and its L_1 norm, $|r^p|_{L_1}$ are known. The solution

vector is first prolongated to the $p+1$ level and the residual is calculated along with its L_1 norm $|r^{p+1}|_{L_1}$. Iterations on the next finer level starts when $|r^p|_{L_1} < \eta_r |r^{p+1}|_{L_1}$(usually $\eta_r = 0.5$). If this condition is not satisfied, another V-cycle at level p is carried out [26].

5.3.2 Smoothers

At each level of the p-multigrid process, the time integration method needs to be performed in order to estimate the solution at that time level. In the p-multigrid approach, we use an implicit smoother at the coarsest level and a less expensive semi-implicit Runge-Kutta scheme as smoother for other levels. The details of these strategies are discussed below. With the exception of the coarsest level $p = 0$, the m-stage semi-implicit Runge-Kutta scheme is used, which can be written as:

$$
\begin{aligned}
&U^0 = U^n \\
&\text{DO } k = 1, m \\
&\quad [\mathbf{M} + \alpha_k \Delta t \cdot \mathbf{D}(U^0)]\delta U^k = -\mathbf{M}(U^{k-1} - U^0) - \alpha_k \Delta t \cdot \mathbf{R}(U^{k-1}) \\
&\quad U^k = U^{k-1} + \delta U^k \\
&\text{END DO} \\
&U^{n+1} = U^m
\end{aligned}
\tag{5.25}
$$

where $\mathbf{D}(U^0)$ is the block diagonal part of the full Jacobian matrix $\partial \mathbf{R}(U^0)/\partial U$. Replacing full Jacobian matrix with block diagonal part makes the system matrix easy to be solved. The LU factorization can be used to solve this linear system directly. At the coarsest level

$p = 0$, an implicit iterative smoother based on the backward Euler scheme is chosen. The implicit time integration has been introduced in previous section.

CHAPTER 6

DIFFUSION BASED LIMITER

An important challenge of applying a high-order DG discretization to the governing equations on an unstructured mesh is its instability when handling flow discontinuities. Historically, many techniques including shock limiting procedures, ENO reconstruction methods, and artificial diffusion schemes have been focusing on the topic of capturing the shocks. The diffusion based limiter for shock capturing inspires our basic approach to solve the multiple species gas mixture interface discontinuities, resulting in a new mass diffusion based limiter. In this chapter, all basic limiters are reviewed and the mass contact-surface-capturing method is discussed in detail.

6.1 Shock-capturing methods summary

Discontinuous Galerkin methods have become popular over the past decades for solving the Euler and Navier-Stokes equations in gas dynamics. With high order discretization employed on either structured or unstructured meshes, these methods can directly utilize the shock-capturing techniques developed for finite volumes and they are especially suitable for creating high resolution shock profiles for high order solutions. For the specific case when first order of accuracy is employed, this DG method is consistent with classical FV approach. Then the natural dissipation (induced by the interface jump mechanism) is

sufficient to damp pressure oscillations and stabilize the solution in the presence of shock waves. However, for higher order approximations, intrinsically explicit dissipation must be added to obtain stable solutions in the region of shock discontinuity or contact surface discontinuity within the framework of the DG method.

The well developed work on shock-capturing methods in FV systems can provide insight in initiating wanted research for the DG system, especially at low order of polynomial approximation. Following the procedure in finite volume systems, the most straightforward approach proposed is decreasing the order of accuracy of the interpolating polynomial at those elements flagged by an effective sensor which has been defined by various approaches [15] to identify the elements lying in the region of discontinuities. Since the order of the interpolating polynomial has been reduced to some level, the jumps across the element interface are then enhanced correspondingly. The increase of inter-element jumps can add some amount of natural dissipation to the DG scheme which is equivalent to manually adding some dissipation to the governing equations. Even for the worst situation when the discretization is taken all the way down to piecewise constant, the DG method can still capture any shock discontinuity with the numerical fluxes computed from the approximate Riemann scheme. The way to design a robust sensor that can detect the troubled cells without invoking excessive dissipation plays an important role in this approach, and so far some practical answers are found in the recent literature [15]. However, one challenge with these limiting schemes is that they may become active away from discontinuities and then the global order of accuracy is seriously deteriorated. Accuracy around shocks can be achieved by adaptively refining the shock wave region, thus decreasing the element length

scale h, and locally the order of accuracy can be recovered to yield satisfactory results. Nevertheless, it is known that the shock waves are anisotropic with lower dimensional features. Then the designed mesh adaptive strategies need to apply refining procedure with directionality consideration in order to fit the shape of the shock, especially in high dimensional situations. Accordingly, this method is not a good choice for our high order simulation with the DG method in three-dimensional problems.

A traditional and successful approach for shock-capturing problems is the limiting technique. Originally a Total Variation Diminishing (TVD) scheme introduced by van Leer [53] was designed to facilitate the solution process in FD and FV systems. Applying the slope limiting procedure to the DG framework was extensively studied by Cockburn and Shu [21] in a series of papers, which are commonly known as the RKDG method. Basically this method took nonlinear operators (slope limiters) together with approximate Riemann solvers to satisfy Total Variation Bounded in the means (TVBM). The RKDG slope limiter is one of the most popular techniques for shock capturing in the DG method, and has been well extended by the scientific community. Although these techniques can yield satisfactory results, the order of the approximations in the vicinity of shock has to be reduced drastically. This is similar to simply degrading the order of accuracy as mentioned before, so mesh adaption procedures are still required in order to achieve a high order of accuracy. Besides, adaptive strategy for shock capturing needs to consider the directionality as well, and the extension of current RKDG slope limiter to multiple dimensional cases is still a challenging problem.

Recently a new and sophisticated approach from the FD world, a high-order non-oscillatory reconstruction known as the (weighted) essentially non-oscillatory (ENO or WENO) approach, has been extensively investigated for shock capturing with a high order of accuracy. This method was initially introduced by Harten and Osher [32], with the purpose of capturing shock wave with high order approximations. The basic idea behind this approach is constructing the solution with large stencils and additional degrees of freedom so as to resolve the shock sharp interface while preserving nonlinear stability. Originally the ENO method was used in the context of FD through Shu and Osher's work [49], and several years later extended to FV solutions [17]. Although this method has several attractive features, some aspects limiting its wide use should be mentioned here. Increasing the degree of high order approximating polynomials results in a loss of robustness while the computational cost remains relatively high. In addition, the method has largely been applied only to structured grids with no obvious extension to practical unstructured meshes. Even if Luo [41] has proposed some ideas on using the WENO method to multiple dimensional unstructured grids with the DG method, we have to admit that in three-dimensional simulations, applying the WENO limiter to the DG method is still an open topic of research.

A more direct scheme, whose underlying mechanism is easy to understand from a numerical standpoint, is explicitly adding diffusion to the governing equations to stabilize the solver in the vicinity of shock discontinuity. This idea is called the artificial viscosity method. Von Neumann and Richtmyer [55] originally proposed the idea of explicitly adding viscous terms to the governing partial differential equations. With the idea originat-

ing from Streamline Upwind Petrov-Galerkin (SUPG) methods developed by Hughes [35] in finite elements framework, Bassi and Rebay [10] and Hartmann [33] applied a residual quantities based artificial diffusion term to the Euler and Navier-Stokes equations. Some preliminary results have been obtained with this method, however some difficulties exist on determining where and how much dissipation needs to be added without smearing the sharp discontinuity profile, which limit its use as an efficient approach. At the same time, Persson and Peraire [44] suggested a new artificial viscosity term involving the mesh size h and the degree of the interpolating polynomial p. The basic idea of this method is to spread the discontinuity over a length scale through diffusive effects so that it can be resolved in the resolution of interpolating functions. With artificial viscosity added, the scales are h/p and the shock width changes to $\delta = Ch/p$ for $C \geq 1$. Following this idea, Barter and Darmofal [5] proposed a smoother representation of artificial viscosity, rather than the piecewise constant approach. These approaches are complemented with a shock detection algorithm which is based on the rate of decay of the expansion coefficients of the solution. Because current research is inspired by Persson and Peraire's shock-capturing method, the details of this approach are repeated and then the new mass diffusion based limiters are introduced in the following sections.

6.2 Shock-capturing methods

The shock-capturing approach consists of adding to the DG discretized equations an artificial viscosity term that aims at controlling the high-order modes of the numerical solution within elements while preserving the spatial resolution of discontinuities. In recent

years, many artificial viscosity shock-capturing methods have been studied. Here only some typical ones are mentioned.

6.2.1 Bassi and Rebay's method

Following the shock-capturing method proposed by Bassi and Rebay [7], the DG discretization of Euler equations in Equation (4.5) is modified according to the following equations:

$$\begin{aligned}\int_{\Omega_h} \frac{\partial U_h}{\partial t}\psi_h \mathrm{d}\Omega \;&-\; \int_{\Omega_h} \mathbf{F}(U_h)\cdot\nabla\psi_h \mathrm{d}\Omega + \int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)(\psi_h^- - \psi_h^+)\mathrm{d}\sigma \\ &+\; \sum_K \int_K \varepsilon_p(U_h^{\pm}, U_h)(\nabla_h\psi_h\cdot\mathbf{b})(\nabla_h U_h\cdot\mathbf{b})\mathrm{d}\mathbf{x} = 0 \end{aligned} \tag{6.1}$$

with the shock sensor and the pressure gradient unit vector defined as:

$$\begin{aligned}\varepsilon_p(U_h^{\pm}, U_h) &= Ch_K^2 \frac{|s_p(U_h^{\pm}, U_h)| + |d_p(U_h)|}{p(U_h)} f_p(U_h) \\ \mathbf{b}(U_h) &= \frac{\nabla_h p(U_h)}{|\nabla_h p(U_h)| + \varepsilon}\end{aligned}$$

where:

$$\begin{aligned}s_p(U_h^{\pm}, U_h) &= \sum_{i=1}^{M} \frac{\partial p(U_h)}{\partial U_h} s_i(U_h^{\pm}) \\ d_p(U_h) &= \sum_{i=1}^{M} \frac{\partial p(U_h)}{\partial U_h} (\nabla_h\cdot\mathbf{F}(U_h))_i\end{aligned} \tag{6.2}$$

and M is number of equations, and ε is machine zero. The components s_i of the function $\mathbf{s}$, defined by the solution of the problem:

$$\int_{\Omega_h} \psi_h \mathbf{s}(U_h^{\pm})\mathrm{d}\mathbf{x} = \int_{\Sigma_h} [\psi_h]\cdot(\widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-) - \mathbf{F}(U_h))^{\pm}\mathrm{d}\sigma \tag{6.3}$$

are actually the interface jump lifting in normal direction between the approximation and inviscid flux components. The pressure sensor defined as:

$$f_p(U_h) = \frac{|\nabla_h p(U_h)|}{p(U_h)} \Big(\frac{h_K}{k}\Big) \tag{6.4}$$

keeps the solution accurate in regions with smooth gradients and k is degree of freedom. The value of parameter C (typically C=0.2) is the same for different degrees of polynomial approximation. And h_K is given as:

$$h_K = \frac{1}{\sqrt{\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} + \frac{1}{(\Delta z)^2}}} \tag{6.5}$$

where Δx, Δy and Δz are the dimensions of the enclosing element K.

6.2.2 Persson and Peraire's method

Persson and Peraire's shock-capturing method consists of a discontinuity sensor and two choices of controlling the amount of added viscosity. All of them are briefly mentioned next.

6.2.2.1 Discontinuity sensor

We write the solution within each element in terms of a hierarchical family of orthogonal polynomials. The three-dimensional Jacobi polynomials are used to represent the solution by a series of continuous functions, and specifically on structured grid they degenerate to frequently used Legendre polynomials. For smooth solutions, the coefficients in the expansion are expected to decay very quickly. On the other hand, when the solution is not smooth, the strength of the discontinuity will be measured by the rate of decay of

the expansion coefficients. The solution of order p within each element can be expressed in terms of an orthogonal basis as:

$$U = \sum_{i=1}^{N(p)} U_i \psi_i \tag{6.6}$$

where $N(p)$ is the total number of terms in the expansion and ψ_i are the basis functions. In addition, we consider a truncated expansion of the same solution, only containing the terms up to order $p-1$, resulting in:

$$\hat{U} = \sum_{i=1}^{N(p-1)} U_i \psi_i \tag{6.7}$$

Within each element Ω_e, the smoothness indicator is defined as:

$$S_e = \frac{(U - \hat{U}, U - \hat{U})_e}{(U, U)_e} \tag{6.8}$$

where $(\cdot, \cdot)_e$ is the standard inner product in $L_2(\Omega_e)$. We found that for the three-dimensional Jacobi polynomial used, the expected value of S_e will scale like $\sim 1/p^6$. Once the shock has been detected, the amount of viscosity is taken to be constant over each element by the smooth function:

$$\varepsilon_e = \begin{cases} 0 & \text{if } s_e < s_0 - \kappa, \\ \frac{\varepsilon_0}{2}(1 + sin\frac{\pi(s_e - s_0)}{2\kappa}) & \text{if } s_0 - \kappa \leq s_e \leq s_0 + \kappa, \\ \varepsilon_0 & \text{if } s_e > s_0 + \kappa. \end{cases} \tag{6.9}$$

where $s_e = log_{10}(S_e)$ and the parameter $\varepsilon_0 = h/p$, $s_0 = log_{10}(1/p^6)$ and κ is chosen empirically (here $\kappa = 0.5$).

6.2.2.2 Laplacian artificial viscosity

The original Euler equations with the added diffusive model term becomes:

$$\frac{\partial U}{\partial t} + \nabla \cdot \mathbf{F}(U) = \nabla \cdot (\varepsilon \nabla U) \tag{6.10}$$

where the parameter ε controls the amount of diffusion. When characteristic quantities such as density or Mach number are applied to the discontinuity detector, the ε in Equation (6.10) will be replaced by an element-wise viscosity coefficient ε_e which vanishes in the region away from discontinuities. In order to discretize the viscous terms in the above equations, the Bassi and Rebay's BR2 model and Cockburn and Shu's LDG model are both employed to solve the artificial viscous flux contribution.

6.2.2.3 Physical artificial viscosity

An alternative viscosity model is based on the real physical dissipation of an ideal gas. Hence the equations become:

$$\frac{\partial U}{\partial t} + \nabla \cdot \mathbf{F}(U) = \nabla \cdot \mathbf{G}(U, \nabla U) \tag{6.11}$$

where the viscous flux vector is in identical form to the physical viscous flux vector, but the amount of viscosity μ and thermal conductivity κ need to be determined. Usually the viscosity coefficient is assumed to be a function of temperature, hence $\mu = \mu^*(T/T^*)^{0.5}$, where μ^* is the viscosity at sonic temperature $T^* = 2T_0/(\gamma + 1)$ and μ^* can be further investigated as a function of shock thickness δ_s. The thermal conductivity can be computed from the given Prandtl number $Pr = \mu C_p/\kappa$. Again the shock detector is used to tell where the viscous flux is added and it vanishes in the region away from discontinuities. In the

same way, the BR2 model and LDG model can be taken to treat the viscous flux component in the governing equations.

6.3 Contact-surface-capturing method

In this section, a new mass diffusion based limiter is proposed. Inspired by the idea of artificial viscosity shock-capturing methods for shock wave discontinuities, a multiple species contact-surface-capturing method is developed in this research for capturing the contact discontinuities at a high order of accuracy with the DG method. For simplicity, only two species are considered, but it is very straightforward to extend the number of species as required by specific problems. The gas mixture composed of two species is assumed frozen (no chemical reactions occur between the components). In the following, the details of this approach will be introduced.

6.3.1 Start from physics: mass transfer and heat conduction

Transport processes in the environment may be divided into two categories: convection and diffusion. Convection is transporting with the mean fluid flow from one place to another by bulk fluid motion. Density, momentum and energy can be transported by convective effects. In contrast, diffusion is transporting of quantities by the action of random motions. Diffusion has the effect of eliminating sharp discontinuities in concentration resulting in smoother and flatter profiles. Convective and diffusive processes are usually considered independent of each other. There are two types of random motion that influence a diffusion process, the random motion of molecules in a fluid, called molecular diffusion and the random eddies in turbulent flow, called turbulent diffusion or eddy dif-

fusion. Another diffusion-like process is dispersion, which comes from the difference of flow pathways or flow speeds, and is not discussed here. Together with viscous diffusion and heat conduction transport phenomena, all the above transport activities can be seen in the RANS governing equations. Now let us look at a simple example in our daily life. A spot of dye is dropped into the center of a river, just from our observation without considering energy transfer processes, convection moves the center of mass of the dye spot downstream, whereas diffusion spreads out the centered spot of dye to a larger and scattered region. If following the center of the dye spot down the river, we can see that the random motion of dye molecules across the boundaries slowly spreads out the spot, and the turbulent diffusion would occur, in a much faster manner, as a result of eddies in the river mixing the clean water from outside the spot with dye-colored water within the spot. If a line of dye were laid across the river at one point, it would be stretched out as it flowed down the river, with the center part of the line moving faster than the edges, due to the effect of dispersion spreading out in the longitudinal direction of the flow.

Although mass transfer happens whenever fluid flows in way of bulk fluid motion (convection), our interest now is on the topic of the transport process of one chemical species within a mixture of chemical species that occurs as a direct result of a concentration gradient (diffusion). The heat conduction process from temperature gradient has been widely known, and the process of mass diffusion is analogous to heat transfer. The governing equations for both cases are similar and therefore many of the relations and solution techniques that have been developed for heat transfer can be directly applied to the mass diffusion process as well. First of all, some basic relations are introduced regarding the concen-

tration of different species in a mixture. Then an important law of mass diffusion, Fick's law, is derived in a similar manner as Fourier's law for heat conduction. More details on mass diffusion constants and the compact form of mass diffusion incorporated within the governing equations are given at the end of this section.

6.3.1.1 Species concentration

A gas mixture of multiple chemical species is considered, with each species assumed frozen (no chemical reactions occur between the components). The mixture properties can be derived from the properties of the individual components and the knowledge of the composition of the mixture. The total mass of the mixture M is the sum of the mass of each component M_i, with i indicates the i-th species in the mixture:

$$M = \sum_{i=1}^{Ns} M_i \tag{6.12}$$

where Ns is the number of species present in the mixture. Dividing both sides by the volume of the mixture, the species and mixture densities are related:

$$\begin{aligned} \rho &= \sum_{i=1}^{Ns} \rho_i \\ \sum_{i=1}^{Ns} \frac{\rho_i}{\rho} &= \sum_{i=1}^{Ns} Y_i = 1 \end{aligned} \tag{6.13}$$

where the ratio of the species density to the density of the mixture ρ_i/ρ is the mass fraction Y_i of species i. It is often useful to compute composition from moles rather than mass. A mole of species i is defined as the amount of mass that is equal to the molar mass $\hat{M}_i$ of species i. The moles and mass of species i are related by:

$$N_i = \frac{M_i}{\hat{M}_i} \tag{6.14}$$

The total number of moles N in the mixture is the sum of the moles of each component N_i as:

$$N = \sum_{i=1}^{Ns} N_i \tag{6.15}$$

The mole fraction y_i of species i is the number of species moles divided by mixture moles:

$$y_i = \frac{N_i}{N} \tag{6.16}$$

The mass and moles of mixture are related by molar mass of mixture according to:

$$\hat{M} = \frac{M}{N} \tag{6.17}$$

where

$$\hat{M} = \sum_{i=1}^{Ns} y_i \hat{M}_i \tag{6.18}$$

The mass fraction and mole fraction are related according to:

$$\hat{M}_i = \frac{M_i}{N_i} = \frac{Y_i}{y_i}\frac{M}{N} = \frac{Y_i}{y_i}\hat{M} \tag{6.19}$$

The mass diffusion process of a species in the mixture is caused by differences in the concentration of this species. Concentration is defined as the amount of species per unit volume, which can be mass or mole based. If mass is employed, the concentration of species i is the species mass per unit volume which is the species density as:

$$\rho_i = \frac{M_i}{V} = \frac{Y_i M}{V} = Y_i \rho \tag{6.20}$$

where V is the volume and ρ is the mass density of the mixture. The molar concentration of species i is the species number of moles per unit volume which is the species mole density as:

$$n_i = \frac{N_i}{V} = \frac{y_i N}{V} = y_i n \tag{6.21}$$

where n is the molar density of the mixture. These quantities are defined here and will be used in the formulae of mass diffusion computation. In this research, we prefer mass of species as principal variable and species density as the concentration. All the following derived equations are based on this choice, but it is straightforward to switch to mole-based equations.

6.3.1.2 Fick's Law

Mass diffusion process from concentration gradients in a mixture is similar to the thermal conduction process due to energy transfer by temperature gradients. Now following the standard derivation of Fourier's law, we can derive the law for mass diffusion.

Consider a mixture composed of nitrogen and oxygen molecules (N_2 and O_2) initially not mixed up, as shown in Figure 6.1. Each species in the mixture is assumed frozen. Initially the left region contains only N_2 and the right region contains only O_2. Mass diffusion can move mass from regions of higher concentration to regions of lower concentration, and if left to continue indefinitely, it would eventually result in equal concentrations in the entire region and arrive at equilibrium with uniform distributions. Since species transport phenomena can be considered independent, then we take nitrogen molecules in the derivation and the equations in the following are suitable for oxygen molecules as well.

Now let us look at N_2 mole fraction changes when the mixing process starts to work. The mole fraction of N_2 is highest at the contact interface (x=0) which we called balance value (y_{bal}). The mole fraction of N_2 decreases with distance away from the contact surface, eventually reaching a value of constant mole fraction that corresponds to the region

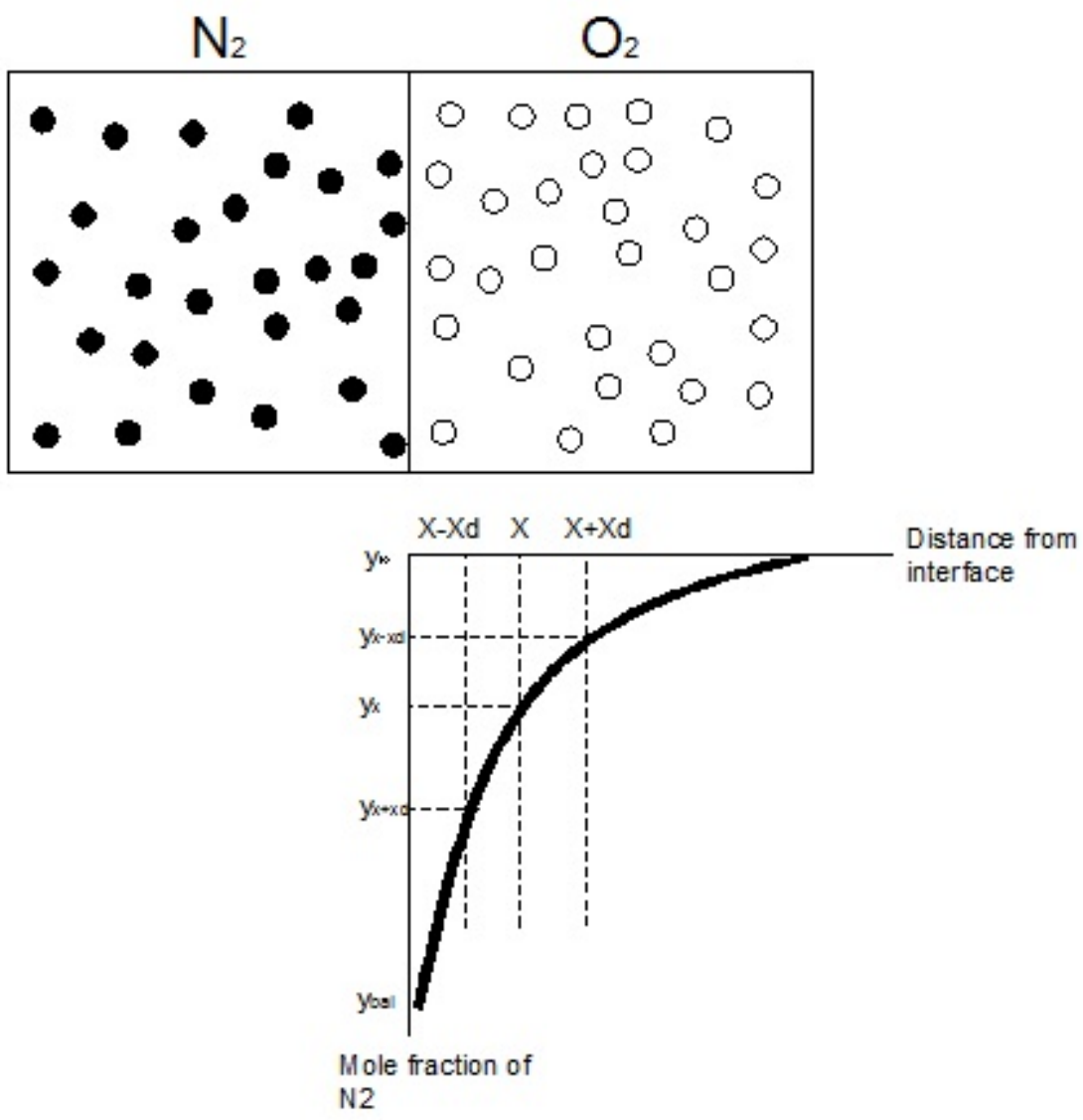

Figure 6.1

Mass diffusion process of nitrogen species in a mixture (●:N_2 molecules ○:O_2 molecules)

not affected by the diffusion process in unsteady situation. Mass transfer of N_2 is driven by the gradient of the mole fraction y_{N_2} that exists in x-direction. The flux of molecules of all species that are passing through a plane located at position x is proportional to the molar density of the mixture (n) and the mean velocity of the molecules (u) at that location. On average, these molecules experienced their last molecular interaction at $x - x_d$, where x_d is the average distance between molecular interaction (i.e., the mean free path). The number of N_2 molecules is proportional to the mole fraction of N_2 at position $x - x_d$. The molar flux of N_2 passing through a plane located at position x in the positive x-direction due to diffusion ($\dot{n}_{N_2,x^+}$) is given approximately by:

$$\dot{n}_{N_2,x^+} \approx n \cdot u \cdot y_{N_2,x-x_d} \tag{6.22}$$

where $y_{N_2,x-x_d}$ is the mole fraction of N_2 at position $x-x_d$. Similarly, the molar flux of N_2 crossing a plane in the negative x-direction due to diffusion ($\dot{n}_{N_2,x^-}$) is given approximately by:

$$\dot{n}_{N_2,x^-} \approx n \cdot u \cdot y_{N_2,x+x_d} \tag{6.23}$$

The net molar flux of N_2 from diffusion($\dot{n}_{N_2}$) is the difference between $\dot{n}_{N_2,x^+}$ and $\dot{n}_{N_2,x^-}$ which is expressed as:

$$\dot{n}_{N_2} \approx n \cdot u(y_{N_2,x-x_d} - y_{N_2,x+x_d}) \tag{6.24}$$

which can be written in another form as:

$$\dot{n}_{N_2} \approx -2n \cdot u \cdot x_d \underbrace{\frac{(y_{N_2,x+x_d} - y_{N_2,x-x_d})}{2x_d}}_{\frac{\partial y_{N_2}}{\partial x}} \tag{6.25}$$

Assume that the length scale between molecular interactions is much smaller than the one that characterizes this problem, then Equation (6.25) can be written in terms of the N_2 mole fraction gradient as:

$$\dot{n}_{N_2} \approx -2n \cdot u \cdot x_d \frac{\partial y_{N_2}}{\partial x} \tag{6.26}$$

Equation (6.26) provides the motivation for Fick's law, which states that mass diffusive function is proportional to the N_2 mole fraction gradient. Fick's law is typically written in terms of the diffusion coefficient D_{N_2,O_2} for N_2 in the mixture:

$$\dot{n}_{N_2} = -n \cdot D_{N_2,O_2} \frac{\partial y_{N_2}}{\partial x} = -D_{N_2,O_2} \frac{\partial n_{N_2}}{\partial x} = -D_{N_2,O_2} \nabla_x n_{N_2} \tag{6.27}$$

the equations above are built on mole fraction, but our interest actually is in the form of mass fraction, then Fick's law can be rewritten in mass fraction. The mass diffusive flux of N_2 ($\dot{m}_{N_2}$) is related to mass fraction gradient or mass concentration as:

$$\dot{m}_{N_2} = -\rho D_{N_2,O_2} \frac{\partial Y_{N_2}}{\partial x} = -D_{N_2,O_2} \frac{\partial \rho_{N_2}}{\partial x} = -D_{N_2,O_2} \nabla_x \rho_{N_2} \tag{6.28}$$

Equation (6.27) and Equation (6.28) are two equivalent statements of Fick's law in mass diffusion, where the N_2 mass diffusion coefficient in the mixture can be defined in the same way as the N_2 thermal conductivity coefficient in Fourier's law for conduction in the following:

$$\dot{q} = -\kappa \frac{\partial T}{\partial x} \tag{6.29}$$

In above Fick's law and Fourier's law, the diffusive flux in general is proportional to a gradient and this constant of proportionality (κ or D_{N_2,O_2}) is a property of the substance

that reflects its microscopic nature. Comparing Equation (6.26) with Equation (6.27) leads to:

$$D_{N_2,O_2} \approx u \cdot x_d \tag{6.30}$$

The diffusion coefficient is closely related to the mean velocity of the molecules and the average distance between molecular interactions. We can see that the diffusion coefficient has the dimensions of $length^2/time$, which is same as kinematic viscosity (ν). From engineering results, we notice that the diffusion coefficients, like D_{N_2,O_2} in mass diffusion, κ in heat transfer and ν in momentum transfer processes, play the similar role so that comparison of these constants may be helpful for the analysis of mass diffusion process.

Generally, the diffusion coefficient is a transport property that represents the ability of species to diffuse in a medium, for example here the case of N_2 transport in the medium of O_2. The medium can be a gas, liquid or solid. Usually diffusion coefficients are largest for gases, lower for liquids and lowest for solids. Gases have a large mean free path which, according to Equation (6.30), leads to a large diffusion coefficient. The diffusion coefficient is a mixture property that depends on the properties of all of the interacting species as well as on pressure and temperature, and this is also true for the thermal conductivity and viscosity of a gas or liquid mixture. Moreover, diffusion problems are often concerned with the mass transfer of a single species, such as above case of N_2, within an otherwise homogeneous phase, such as a pure gas of O_2 or a homogeneous gas mixture (e.g., air). Then the mixture can be treated as a binary system and the binary diffusion coefficient for species 1 through another species 2 is termed $D_{1,2}$. It is possible to show that $D_{1,2}$ must be equal to $D_{2,1}$. The mass transport properties can be found out from experimental data for

different compositions of the mixture. In this research, the N_2/O_2 pair and Air/He pair have been used.

6.3.1.3 Compact governing equations

Since the two species mixture gas model is of interest in investigating the contact-surface-capturing method, then we will apply the mass diffusion relations of Equation (6.28) to a gas mixture composed of two general species resulting in:

$$\begin{aligned} \dot{m}_1 &= -\rho D_{1,2} \nabla Y_1 = -D_{1,2} \nabla \rho_1 \\ \dot{m}_2 &= -\rho D_{2,1} \nabla Y_2 = -D_{2,1} \nabla \rho_2 \end{aligned} \tag{6.31}$$

where the diffusion coefficients $D_{1,2} = D_{2,1}$, and the energy changes due to the mass diffusion process is obtained from [40] as:

$$de_0 = -\rho D_{1,2} \nabla Y_1 \cdot h_1 - \rho D_{2,1} \nabla Y_2 \cdot h_2 \tag{6.32}$$

Therefore, two individual species conservation laws have been used to substitute the original mass conservation equations. Mass diffusion contribution has been added to the Navier-Stokes governing equations in Equation (2.12) for the case of multiple species with the diffusion flux defined in Equation (2.13). So far the system of governing equations with mass diffusion has been set up, and this will be used in the following sections.

6.3.2 Mass diffusion based limiter

In engineering flow field simulations, multiple species flows are common for many applications. Consider a mixture of two fluids as a simplified model. In general, the

two fluids are separated by a sharp interface or discontinuity. Unfortunately there are few accurate numerical schemes designed to capture the sharp species interface effectively with high order of accuracy. Therefore an important task of this research is developing a highly accurate DG based numerical solver for the simulation of compressible, unsteady and inviscid two-fluid flows described by the three-dimensional Euler equations of gas dynamics.

The two fluids in the mixture are considered thermally perfect gases with no chemical reactions. If this mixture is composed of two gas species with close molecular weight, for example nitrogen molecules (N_2) and oxygen molecules (O_2), then their thermodynamic properties are similar so that the interface discontinuity is not large enough to bring instability from non-physical species properties. Let us look at some numerical results from the shock-bubble problem, where a shock wave of $Ma = 1.22$ is moving from left to right across on O_2 gas flow field with a gas bubble filled with N_2 located inside the domain. The DG method with fourth order of accuracy is used to simulate the unsteady process, and the initial condition is given in Figure 6.2(a). When the shock wave intersect with the N_2 bubble, the mixture density contour is shown in Figure 6.2(b) and the mass fraction of species O_2 is given in Figure 6.2(c). From these results, we find out that if species properties are similar to some extent, the contact discontinuity is very weak, and approximate Riemann schemes can capture it with a high order accurate DG method. The resulting mass fraction is always positive and properties are well maintained satisfying physical conditions. In Figure 6.3(d) and Figure 6.3(e), the contact surface is clear to see in the density contour

when the shock moves passed the N_2 bubble, and the species mass fraction is kept in a reasonable range (between 0 and 1).

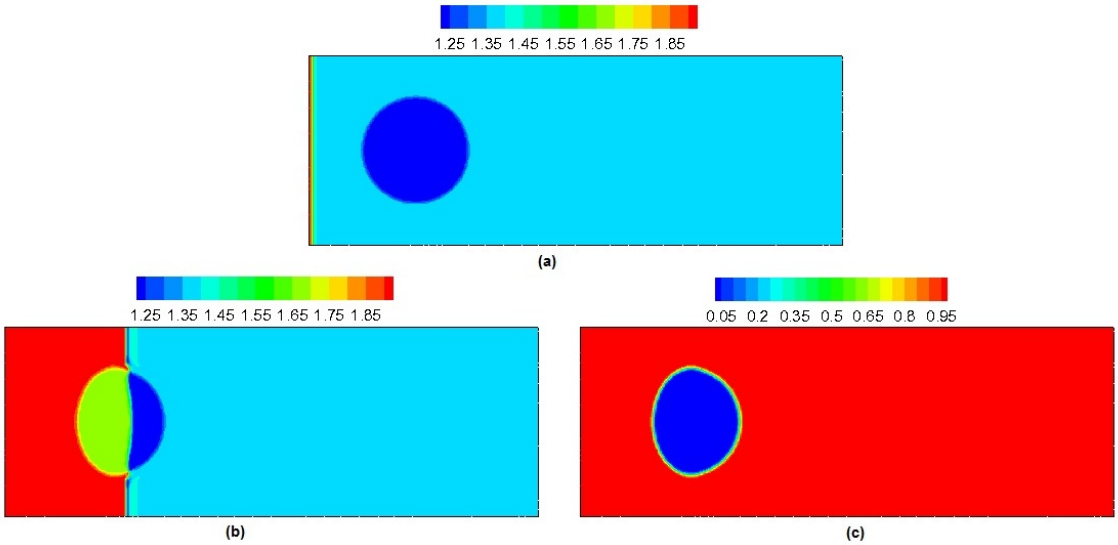

Figure 6.2

O_2/N_2 shock bubble interaction (a)initial condition (CFL=0.1) (b)mixture density contour at t=0.14s (c)O_2 species mass fraction at t=0.14s

Unfortunately, these reasonable results are not always possible for a gas mixture of any type of species. In the classical air-helium shock bubble problems, we note that a mixture of air (Air) and helium (He) with average molecular weights of 29 and 4 respectively, is a challenge for the problem just mentioned. In this case, a shock wave of $Ma = 1.22$ is travelling from left to right in the Air flow field with He gas bubble initially located in the domain. A high order ($p \geq 2$) DG method is employed with the approximate Riemann solver to compute the solution of this typical example. However, an unexpected problem happens with this high order scheme. The species density and mass fraction in the cells

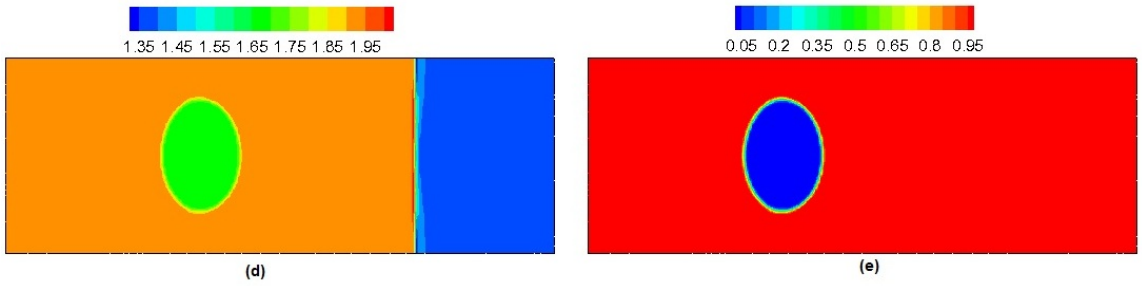

Figure 6.3

O_2/N_2 shock bubble interaction (d)mixture density contour at t=0.47s (e)O_2 species mass fraction at t=0.47s

where the contact surface is confined by high order DG approximation are experiencing large gradients and the values are negative at faces, which can produce non-physical gas properties and eventually blow up the numerical solver. In order to completely understand the nature of this problem and figure out some effective ways to eliminate these troubles, we first need to build a model for this complicated phenomenon and analyze the physical as well as numerical quantities in the contact surface region.

In the mixture, the species Air is numbered as 1 with density of ρ_1 and the species He is numbered as 2 with density of ρ_2. Moreover, the three-dimensional air-helium interaction with contact surface can be simplified to one dimension here for analysis (see Figure 6.4). From this illustration we can see that the solution profile is not piecewise constant any more, instead a high order polynomials represented solution is adopted. With high-order DG method used, the sharp contact discontinuity can be limited to a single cell, but the slope of the mass fraction of each species could be large from the solution integration strategy. At this moment, the mass fraction on the interface between this trouble cell and

its neighbor could possibly be negative as shown in this figure $Y_1 = -0.6$ and $Y_2 = -0.3$ for example although the mean values are still reasonable of $Y_1 = 0.7$ and $Y_2 = 0.3$. As we know, the mass fraction of each species should be in the range of 0 and 1 so that the numerical quantities can be consistent with the physical meaning. The non-physical mass fraction could cause the mixture thermodynamic properties, such as C_v, C_p and γ to be either negative by large amounts or positive by large amounts, which would definitely destroy the flux approximation at the cell interface hence ruin the whole integration system.

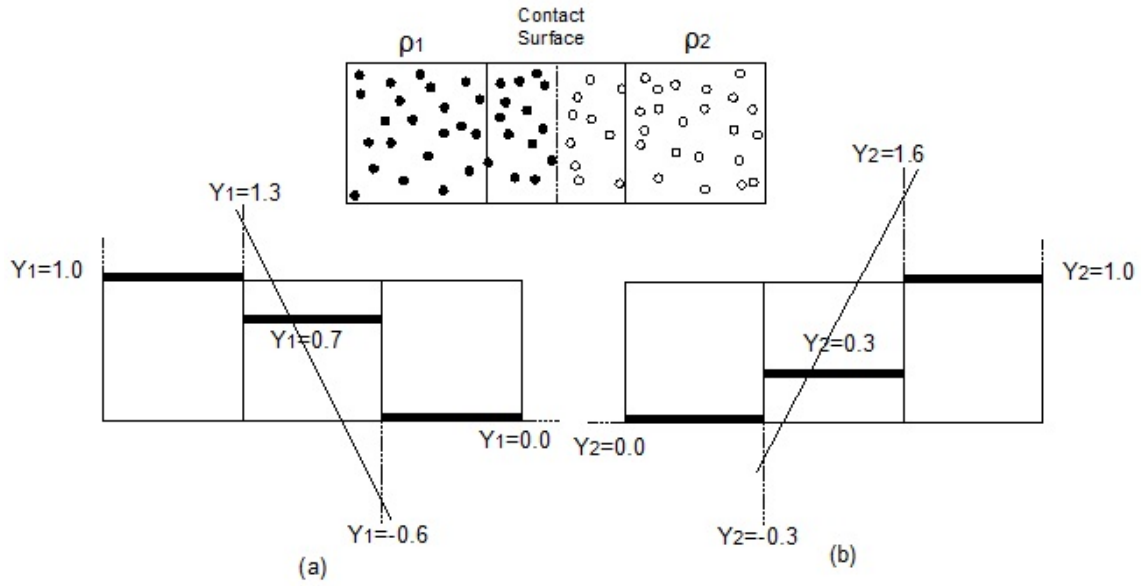

Figure 6.4

Illustration of $Air(\rho_1)/He(\rho_2)$ contact discontinuity at neighboring cells (a)Air species mass fraction (b)He species mass fraction

The artificial viscosity mass diffusion based limiter method aims at limiting the mass fraction gradient to reasonable values by way of adding mass diffusion to the original

governing equations. In Figure 6.5 the mass diffusion is supposed to be added to the system according to the formulae from last section. Then the Air molecules are mixed up with He molecules resulting in the decrease of the mass fraction slope hence improvement of the value of mass fraction at the cell interface. From the numerical results in the next chapter, we can see that the species mass fraction and density can be corrected from negative value to a reasonable positive value (for example, $Y_1 = 0.05$ and $Y_2 = 0.1$ in this figure) while the contact discontinuity can still be captured within a single cell with DG method if proper amount of diffusion is loaded. We note that the mean value of mass fraction is not changed, but the species molecular distribution is actually modified. In Figure 6.6, the corrected species mass fraction is displayed. With this method implemented, it is guaranteed that there are no non-physical quantities existing in the integration scheme, and the numerical flux approximation at cell interface can be meaningful.

Furthermore, some important features of mass diffusion based limiter are explained next. The mass diffusion based limiter coupled with the DG method can produce a high order accurate and stable solver which is conservative and consistent. The scheme is stable since the non-physical factors have been removed from the solver and all the computational results are physically meaningful. The species mass conservation laws can be repeated from Equation (2.12) and Equation (2.13) as:

$$\begin{aligned} \frac{\partial \rho_1}{\partial t} + \frac{\partial (\rho_1 u_j)}{\partial x_j} - \frac{\partial (\rho D_{1,2} \nabla Y_1)}{\partial x_j} = 0 \\ \frac{\partial \rho_2}{\partial t} + \frac{\partial (\rho_2 u_j)}{\partial x_j} - \frac{\partial (\rho D_{2,1} \nabla Y_2)}{\partial x_j} = 0 \end{aligned} \tag{6.33}$$

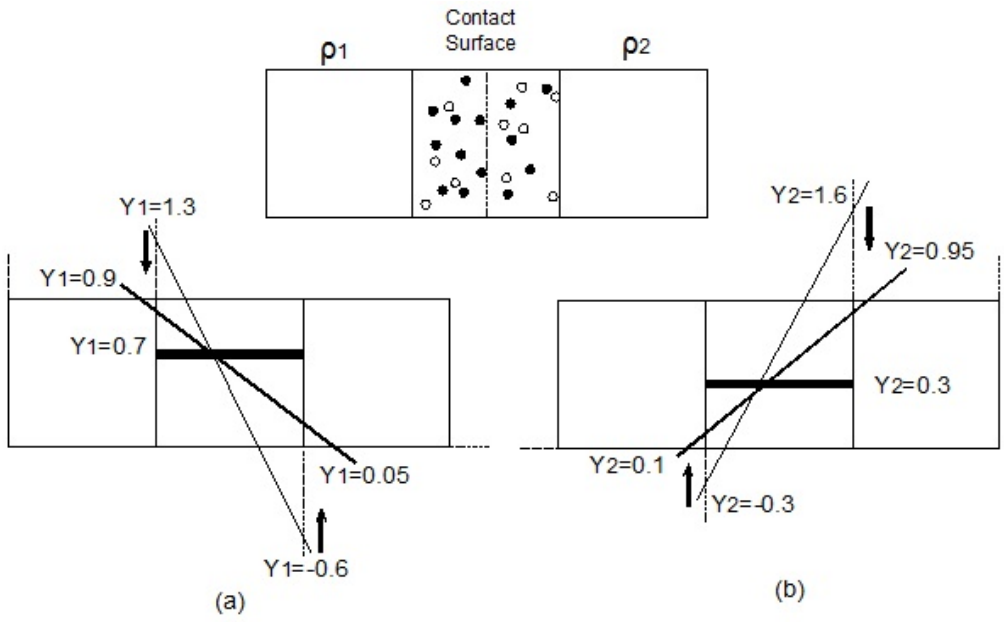

Figure 6.5

Illustration of $Air(\rho_1)/He(\rho_2)$ mass diffusion process at neighboring cells (a)Air species mass fraction (b)He species mass fraction

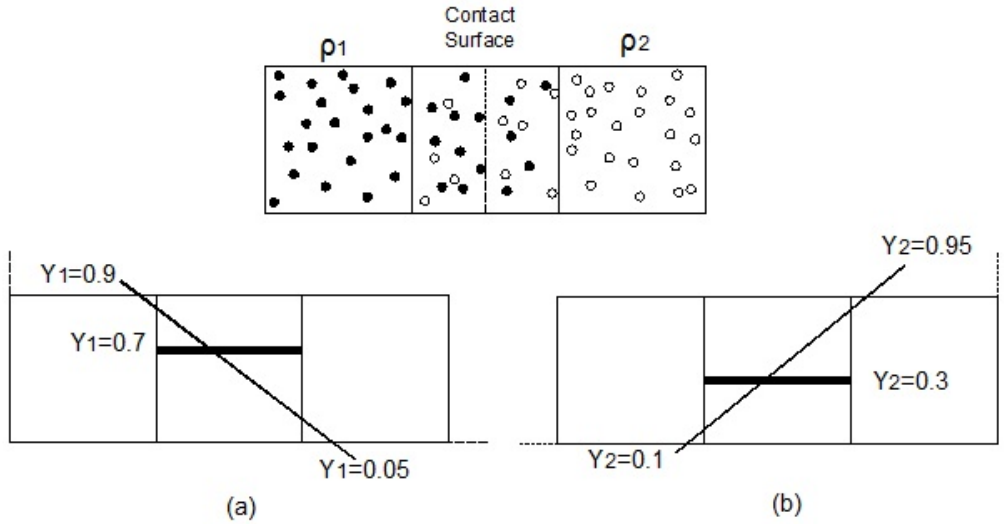

Figure 6.6

Illustration of $Air(\rho_1)/He(\rho_2)$ corrected results at neighboring cells (a)Air species mass fraction (b)He species mass fraction

We find that the mass diffusion components in the above two equations can be cancelled out if the species mass equations are summed up and the result is identical to the mass conservation laws of gas mixture. Except for the energy contribution to mass diffusion process from Equation (6.32) which is not equal to zero generally because the species enthalpy is different for each species, we can conclude that the mass conservation law is consistent with the original governing equations for mass transfer. This does make sense since the mass diffusion process only modifies the distribution of the species mass fraction in cell so as to avoid the negative density on cell boundaries and the cell interface flux has been maintained the same. This also proves that the mass diffusion process is conservative. Lastly, since this scheme is employed under the framework of DG method with high order of accuracy, an important feature is that it improves mass fractions while preserving the formal order of accuracy. This is probably the most attractive characteristic of this scheme and is better than other limiting methods, as already discussed, thus it provides a good choice for multiple species simulation with high order of accuracy.

6.3.3 Procedure of applying limiter to solver

The basic principle of mass diffusion based limiter has been discussed in detail from the last section. Now implementation steps will be covered. Based on the Euler equations from gas dynamics with DG discretization, the mass diffusion is added to the solver following the steps below to capture the sharp contact surface discontinuity with length scale less than one cell.

First of all, consider the discontinuity detector from Persson and Peraire [44] in Equation (6.8):

$$S_e = \frac{(U - \hat{U}, U - \hat{U})_e}{(U, U)_e} \tag{6.34}$$

with species density as the characteristic variable to identify the trouble cells in the solution domain where negative species mass fraction could happen, through detecting the rate of decay of the expansion coefficients of the solution. The trouble cells in which the contact discontinuity is located can be indicated by this sensor, whereas the cells that are away from the contact surface discontinuity will not be affected. The second step is to set up a solution iteration on these cells with the converging criteria as follows:

1. On cells, species density $\rho_{Air} > 0$ and species mass fraction $0 \leq Y_{Air} \leq 1$.
2. On cells, species density $\rho_{He} > 0$ and species mass fraction $0 \leq Y_{He} \leq 1$.
3. On cells, species mass fraction gradient $|dY_{Air} + dY_{He}| < \epsilon$, where ϵ is machine zero.
4. On faces interior, species density $\rho^-_{Air} > 0$ and species mass fraction $0 \leq Y^-_{Air} \leq 1$.
5. On faces interior, species density $\rho^-_{He} > 0$ and species mass fraction $0 \leq Y^-_{He} \leq 1$.
6. On faces interior, species mass fraction gradient $|dY^-_{Air} + dY^-_{He}| < \epsilon$.
7. On faces exterior, species density $\rho^+_{Air} > 0$ and species mass fraction $0 \leq Y^+_{Air} \leq 1$.
8. On faces exterior, species density $\rho^+_{He} > 0$ and species mass fraction $0 \leq Y^+_{He} \leq 1$.
9. On faces exterior, species mass fraction gradient $|dY^+_{Air} + dY^+_{He}| < \epsilon$.

Only after all of the previous conditions are satisfied for the trouble cells can the iteration process with mass diffusion terminate. Otherwise, mass diffusion is applied to modify the mass fraction of each species. For the third step, at each iteration mentioned above, a RK4 explicit time integration from Equation (5.2) is applied at pseudo time level. The pseudo

time step used is normally less than the physical time step computed from CFL number, because the mass diffusion equations need a relatively small time step in order to stabilize the solver. Now we take $\Delta t = t_s \cdot \Delta t_{phy}$ where Δt_{phy} is the physical time step given by the CFL condition and t_s is a constant parameter, usually between 0.1 and 1. The fourth step is to compute the residual component of the governing equations. At trouble cells, the governing equations for iterations include only mass diffusion flux without considering convective flux contribution so that the mixing process is going on until the mass fraction slope is not very large. Therefore, two species Air and He mass diffusion equations are inherited from Equation (2.12) with mass diffusion flux given in Equation (2.13) without considering viscosity and heat transfer. The governing equations are discretized by means of the DG method and the LDG numerical approximation scheme is used to solve the system. In the fifth step, after solving the mass diffusion equations separately at trouble cells, the mass fraction of each species should be adjusted to a reasonable value. The regions away from the contact discontinuity are not affected by the diffusion process because they are not identified by the discontinuity sensor. Then the new mass fractions are used to compute the numerical flux approximations of the original flows, thus the numerical results should be meaningful and feature reasonable gas properties.

The whole process discussed above is the mass diffusion based limiter, and can be summarized as:

1. Detecting trouble cells with contact discontinuity sensor.
2. Iterations with necessary mass fraction conditions.
3. RK pseudo time integrations.
4. Solve the DG discretized mass diffusion systems with LDG method.

5. Return to main solver and compute numerical flux from new mass fractions.

This new strategy has been successfully implemented into the current solver to capture the sharp contact surface with the DG method. We call this method contact-surface-capturing method. Numerical results on two species air-helium shock bubble problems are given in next chapter.

CHAPTER 7

RESULTS

Many numerical examples are investigated to validate the newly developed numerical solver, ranging from simple to complex geometries. The numerical schemes analyzed in previous chapters are tested here. Additionally, the newly proposed contact-surface-capturing method is examined by way of the classical shock-bubble problem with the high-order DG method. In all of the following examples, except for the case testing order of accuracy, three-dimensional grids have been used with one dimension reduced to one-cell length, resulting in the computational grid being identical to a two-dimensional grid. Hence the three-dimensional implementation of the solution on original three-dimensional mesh is the same as a two-dimensional solver on two-dimensional plain grids. The computational results are displayed on a two-dimensional plane as a convenience for the reader.

7.1 Test order of accuracy

The first simple and basic test case is showing that a high order of accuracy with the numerical implementation can be obtained which is approximately identical to the actual order of accuracy expected. The Manufactured Solution [43] method has been employed in order to test the order of approximation from the numerical solution. The basic idea behind this approach is the following: a C_2 continuous mathematic function or higher is chosen

as the exact solution to the governing equations in the computational domain, then a new numerical flux or source term constructed from this specified solution has to be added to the original governing equations. After discretization and time integration, the converged numerical result is an approximation of the original manufactured solution. The difference between the exact solution and the approximate solution is the numerical error from the approximation with a high order DG method. Theoretically, this error is proportional to the length scale $\vartheta(h^p)$ and decreases drastically by increasing the order of accuracy of the approximation. In order to compare the errors quantitively for different orders of accuracy, a series of gradually refined simple hexahedral grids has been used in the computation, decreasing the grid cell edge length by half at each time of refinement (See Figure 7.1). The manufactured solution chosen in the following cases is:

$$\begin{aligned} \rho_{ext} &= \rho_\infty(1 + 0.1sin(x)cos(y)) \\ u_{ext} &= u_\infty(1 + 0.1sin(y)cos(z)) \\ T_{ext} &= T_\infty(1 + 0.1sin(y)cos(x)) \end{aligned} \tag{7.1}$$

where ρ_{ext}, u_{ext} and T_{ext} are the manufactured solution for density, velocity and temperature respectively. The freestream conditions and x, y, z global coordinates are given. The errors for the explicit RK inviscid solver with Roe scheme at orders of accuracy ranging from first to fourth are shown in Figure 7.2 (data are listed in Table 7.1), using total energy. Furthermore, the errors for the explicit RK viscous solver with $\kappa - \omega$ turbulence model solved by BR2 scheme at orders of accuracy ranging from first to fourth are shown in Figure 7.3 (data are listed in Table 7.2), using total energy.

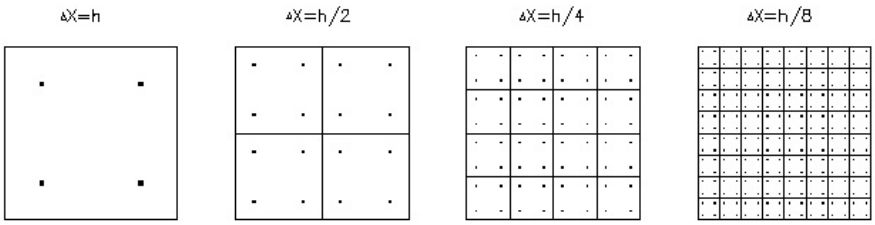

Figure 7.1

Grids with different length scales (divided by two each time, using second order quadrature points)

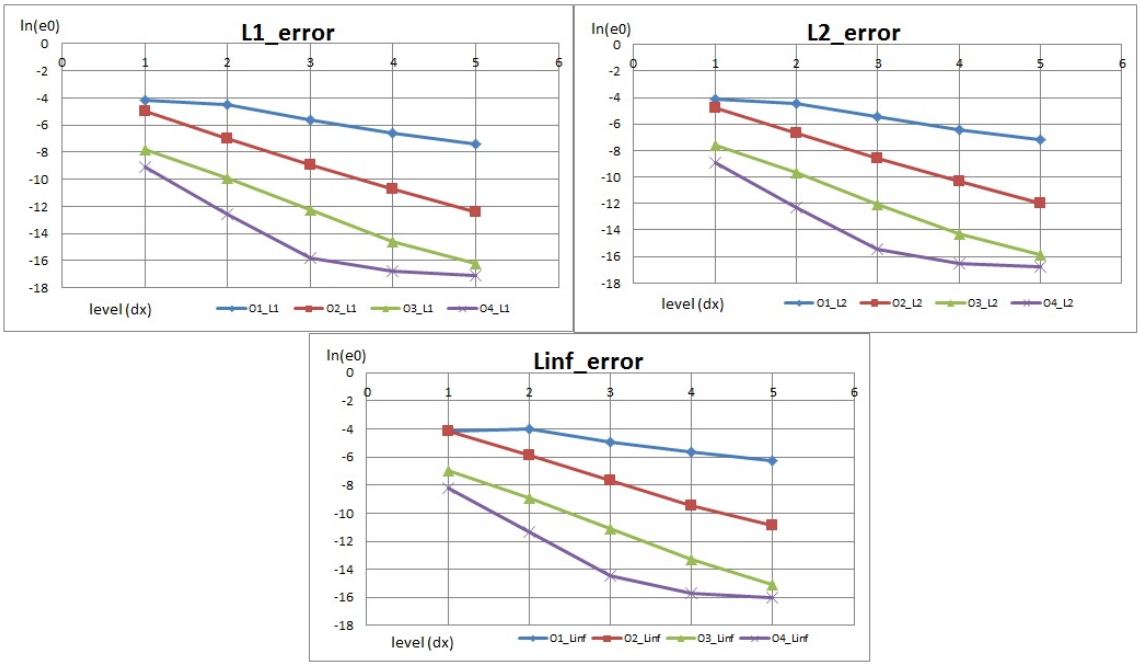

Figure 7.2

Errors for different orders of accuracy on progressively refined grids for inviscid scheme

Table 7.1

Data of errors for different order on refined grids for inviscid solver

L_1 error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.0159519	0.00679733	0.407498E-3	0.10979E-3
8($h/2$)	0.0108563	0.923066E-3	0.509214E-4	0.354651E-5
64($h/4$)	0.00351675	0.132768E-3	0.464611E-5	0.137527E-6
512($h/8$)	0.00132934	0.214681E-4	0.466729E-6	0.5056531E-6
4096($h/16$)	0.626341E-3	0.411676E-5	0.9213601E-7	0.3833126E-7
L_2 error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.0159519	0.00861329	0.50358E-3	0.136624E-3
8($h/2$)	0.0116613	0.00127075	0.639017E-4	0.469059E-5
64($h/4$)	0.00418671	0.19447E-3	0.601904E-5	0.188964E-6
512($h/8$)	0.00165381	0.316976E-4	0.607375E-6	0.691699E-7
4096($h/16$)	0.744522E-3	0.606541E-5	0.1281583E-6	0.5102246E-7
L_∞ error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.0159519	0.0153293	0.91298E-3	0.275167E-3
8($h/2$)	0.0182272	0.00276871	0.133802E-3	0.119633E-4
64($h/4$)	0.00740958	0.453117E-3	0.148286E-4	0.535359E-6
512($h/8$)	0.00359881	0.79526E-4	0.171607E-5	0.1475646E-6
4096($h/16$)	0.00187877	0.197088E-4	0.2740377E-6	0.1072611E-6

Table 7.2

Data of errors for different order on refined grids for viscous solver

L_1 error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.00635105	0.0145564	0.0017594	0.216325E-3
8($h/2$)	0.0404615	0.00244346	0.238298E-3	0.115155E-4
64($h/4$)	0.0201483	0.547681E-3	0.289034E-4	0.650491E-6
512($h/8$)	0.00990994	0.125677E-3	0.337695E-5	0.413268E-7
4096($h/16$)	0.00510534	0.30206E-4	0.399615E-6	0.141142E-7
L_2 error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.00635105	0.019344	0.002197	0.271683E-3
8($h/2$)	0.0517869	0.00315738	0.294413E-3	0.150965E-4
64($h/4$)	0.0267395	0.694506E-3	0.35796E-4	0.847449E-6
512($h/8$)	0.0133102	0.16215E-3	0.419567E-5	0.53518E-7
4096($h/16$)	0.00677791	0.39417E-4	0.491085E-6	0.176329E-7
L_∞ error (number of cells)	order=1	order=2	order=3	order=4
1(h)	0.00635105	0.0404802	0.00449059	0.953337E-3
8($h/2$)	0.109959	0.00885285	0.837904E-3	0.75661E-4
64($h/4$)	0.0791888	0.00211305	0.131493E-3	0.465556E-5
512($h/8$)	0.042071	0.553338E-3	0.189099E-4	0.273771E-6
4096($h/16$)	0.0218843	0.142224E-3	0.262084E-5	0.122007E-6

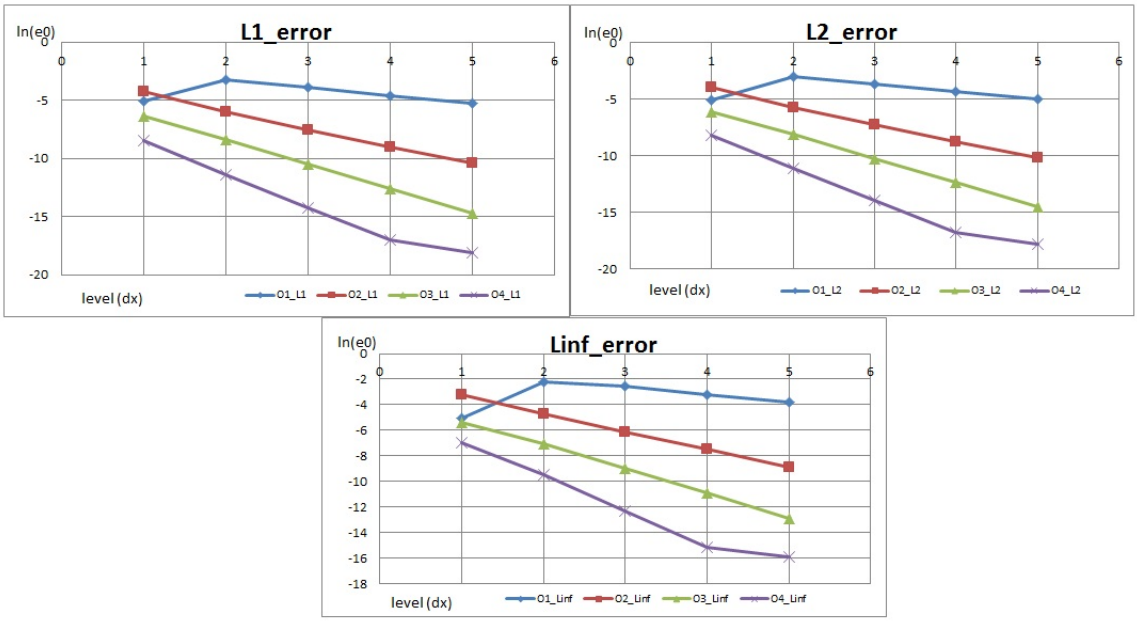

Figure 7.3

Errors for different orders of accuracy on progressively refined grids for viscous scheme

In these figures, the slope of the error is defined as:

$$slope = \frac{|\ln(e_0)^p - \ln(e_0)^{p-1}|}{|\ln(h)^p - \ln(h)^{p-1}|} = \frac{|\ln(e_0)^p - \ln(e_0)^{p-1}|}{\ln(2)} = const \tag{7.2}$$

From Figure 7.2 and Figure 7.3 we can see that the slope of the line is nearly constant for different orders of accuracy as expected. The order of accuracy on a single cell is not well maintained for first order, which is not significant since the errors are very large for only one cell. And we note that at fourth order of accuracy the errors for the finest grid do not decrease as expected since the computational truncation errors are of the same magnitude as the round-off errors, and the round-off errors are actually dominant in this situation so that the high order of accuracy can not be achieved. Except for these, other lines are smooth with constant slope which is equal to the order of accuracy. Since the grids have

been gradually refined, then theoretically the error is $\vartheta(h^p)$, and decreases (in a logarithm) in proportion to the change of grid length, thus the slope should be constant for different order of accuracy. The slopes from the numerical results are very close to the theoretical ones, and this effectively validates the numerical algorithm and proves that the high order DG scheme realizes our original purpose to achieve high order approximation without side effects.

7.2 Inviscid shock-tube problems

The Sod shock-tube problem [50] is a common test case to test the accuracy of approximate Riemann solvers. The unsteady Euler equations are solved in a time-accurate fashion for this problem. Explicit RK time integration is used with HLLC scheme to approximate the convective flux for the DG discretized governing equations. In the unsteady time marching process, three characteristics are formed, describing the propagation of waves. The rarefaction wave, the contact discontinuity, and the shock wave are resolved with appropriate numerical schemes. This test case provides us with some understanding of the basic discontinuities in CFD and ways on how they can be captured with commonly used numerical approximation schemes. This is a standard test case that has been widely studied, and the density, mach number, velocity, and pressure profiles in front of and behind the discontinuities can be compared with analytical solution.

The configuration of a shock tube is shown in Figure 7.4(a). This is a tube closed at both ends, with a diaphragm separating a region of high-pressure gas on the left from a region of low-pressure gas on the right. When the diaphragm is broken, a shock wave

propagates into section 1 and an expansion wave propagates into section 4. As the normal shock wave propagates to the right with velocity W, it increases the pressure of the gas behind it. The interface originally located between region 1 and 4 is called contact surface, which also moves with velocity u_p. The pressure and velocity are the same across the contact discontinuity. The expansion wave propagates to the left, smoothly and continuously decreasing the pressure in region 4 to the lower value behind the expansion wave. The flow field in the tube after the diaphragm is broken is shown in Figure 7.4(b), and is completely determined by the given conditions in region 1 and 4 before the diaphragm is broken.

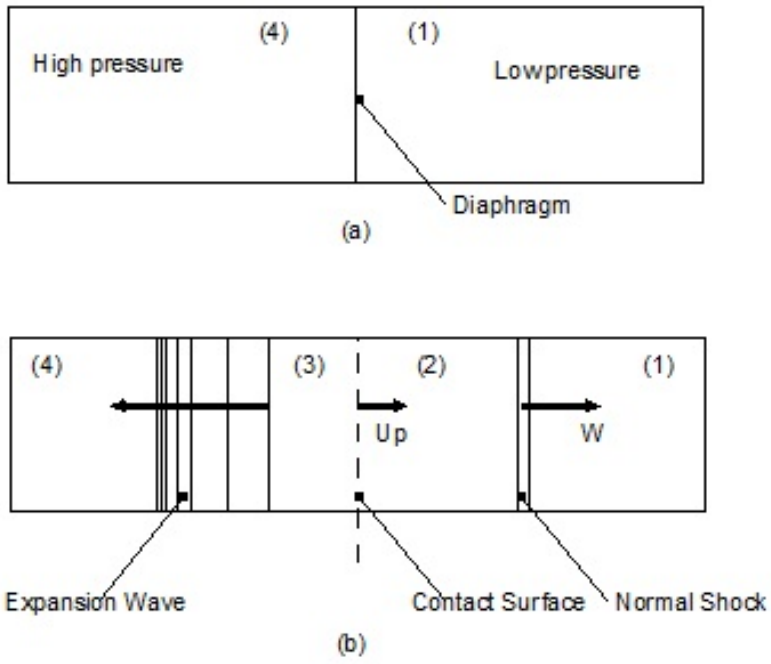

Figure 7.4

(a)Initital condition in a shock tube (b)Flow in a shock after the diaphragm is broken

In this shock-tube problem, two different gas species are employed in different regions to test the contact-surface-capturing method. Nitrogen is used in region 4 and oxygen is used in region 1. The basic conditions are given:

1. Geometry: x $\in$ [0,1] ft, number of grid points is 100.
2. CFL number: first order 0.01; second order 0.01; third order 0.01; fourth order 0.001.
3. Time: compare solution at $t = 1.0E-4s$.
4. Initial condition: P_4=68948Pa, T_4=289K, U_4=0.0, $\gamma_4 = 1.4$; P_1=6894.8Pa, T_1=231K, U_1=0.0, $\gamma_1 = 1.4$.

The inviscid flux are approximated by HLLC scheme. For higher order of accuracy, artificial viscosity diffusion based limiter and the newly proposed contact-surface-capturing method are employed. The numerical solutions are compared with analytical solutions from textbook [1]. The density, Mach number, x-direction velocity, pressure, and species mass fraction of nitrogen gas are shown in Figure 7.5, Figure 7.6, Figure 7.7, and Figure 7.8, Figure 7.9 respectively. In density and pressure figures, the contact surface and shock wave are captured by diffusion based limiter and closed to analytical solutions when higher order of accuracy is employed. From Mach number and x-velocity figures, we can see that the numerical oscillations are large for odd order of accuracy in the region between the shock wave and the expansion wave and the numerical results match to the analytical solutions smoothly for even solution order of accuracy. Moreover, the species mass fractions for nitrogen gas in the region close to the contact surface are displayed in the mass fraction figure. The species mass fraction from numerical results is very closed to the analytical solution for higher order of accuracy. From above figures, we can see that the numerical solution approximates to analytical solution well when the high-order scheme is

used. The shock discontinuities can be well captured with diffusion based limiter. Also the sharp contact surface can be captured effectively with mass diffusion based limiter, and the oscillation is much less. This comparison proves that the diffusion based limiter for shock wave and our new contact-surface-capturing method are robust and effective for simple cases. Also the comparison shows that our time integration is valid for high order accurate simulation.

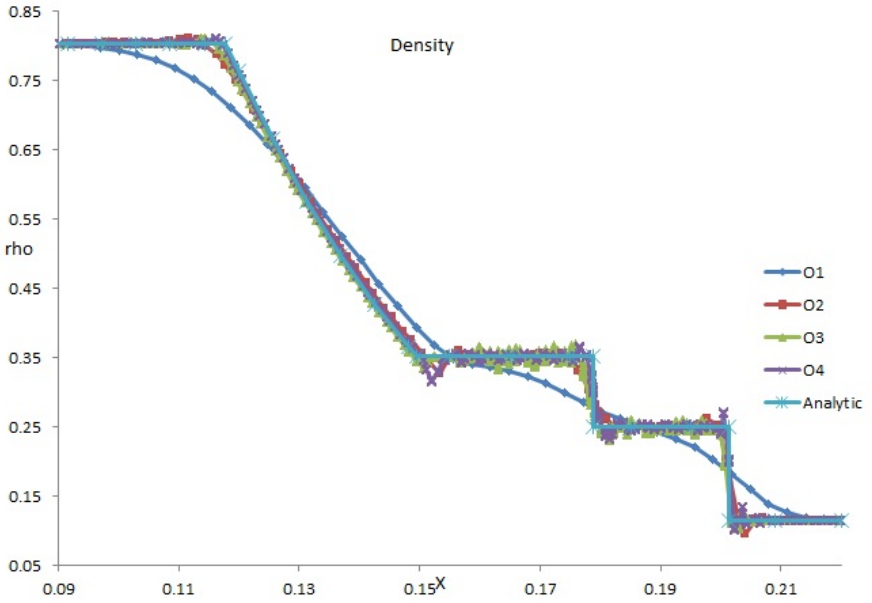

Figure 7.5

Density distribution for different order of accuracy

7.3 Steady supersonic ramp test cases

The high order DG method can be investigated as a candidate for resolving the shock wave and capturing the thin shock discontinuity within a single cell at higher dimensions.

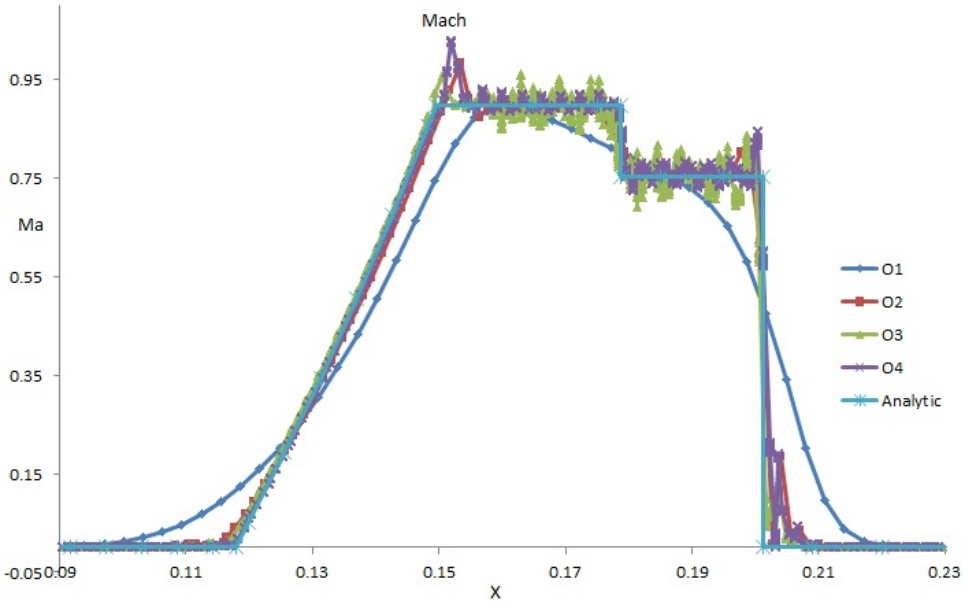

Figure 7.6

Mach number distribution for different order of accuracy

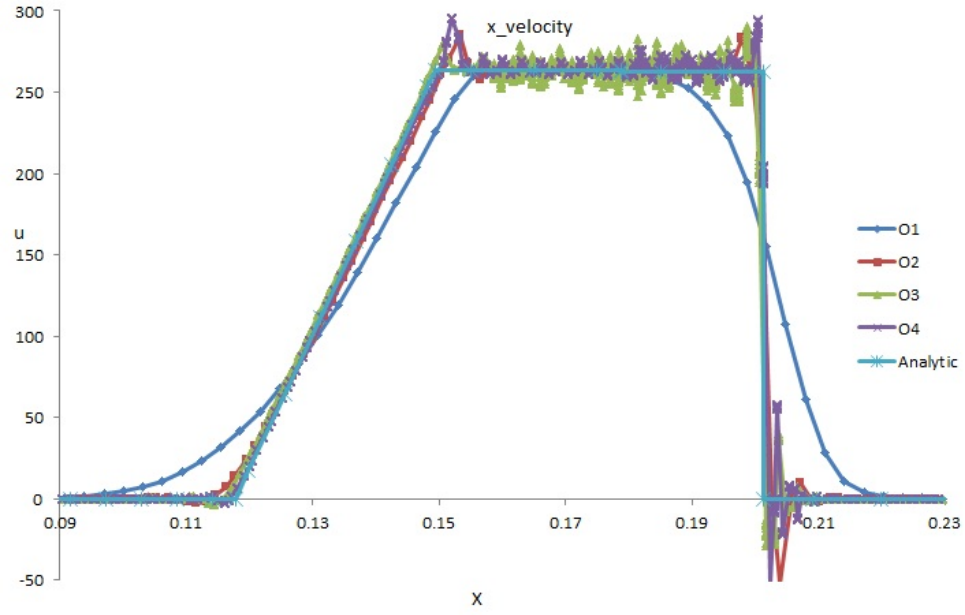

Figure 7.7

X-direction velocity distribution for different order of accuracy

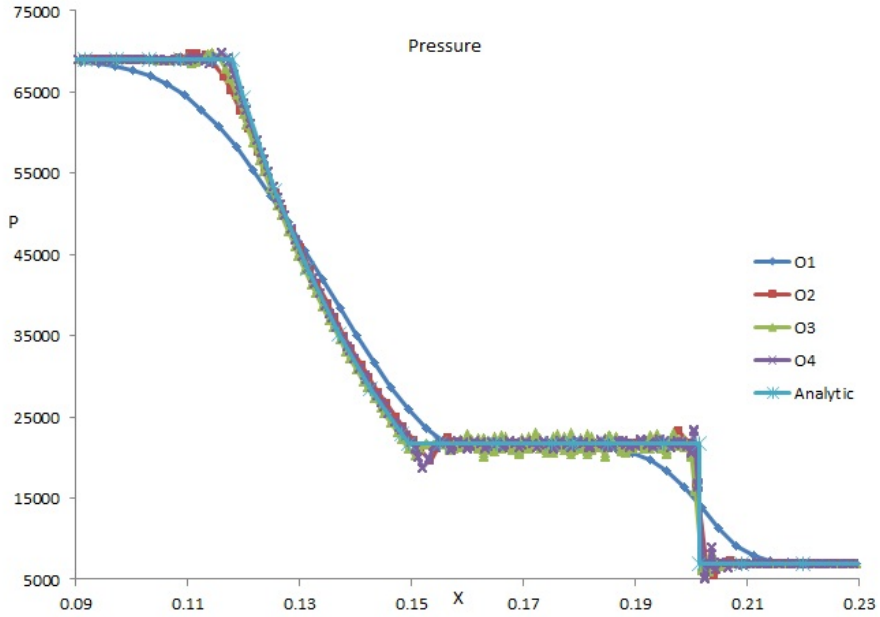

Figure 7.8

Pressure distribution for different order of accuracy

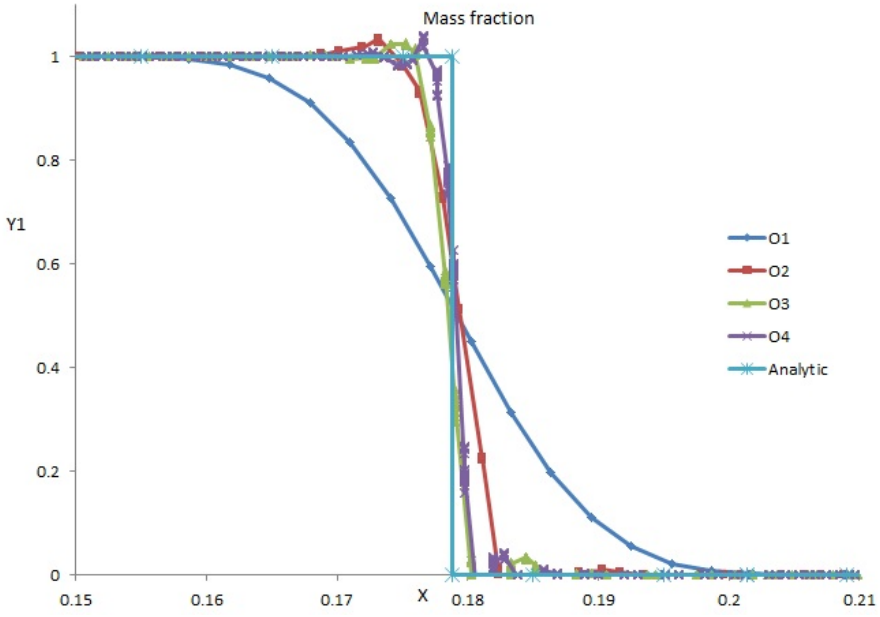

Figure 7.9

Nitrogen mass fraction distribution for different order of accuracy

The first test case is a supersonic air flow passing a simple ramp, and the mesh used for computation is shown in Figure 7.10. A supersonic flow with free stream Mach number 2.5, density 1.3 kg/m^3 and pressure 1atm can be used to test the shock-capturing capability of the solver. The DG discretized Euler equations for air are employed, with semi-implicit p-multigrid acceleration strategy for high order of accuracy, where the convective flux is computed using the Roe flux splitting scheme with entropy fix.

For first order of accuracy, whose pressure contour is shown in Figure 7.11, the DG solver can capture the shock wave across several cells, similarly to a solution using the finite volume method. The details of the pressure contour near the shock region are shown in Figure 7.12. The shock wave is smeared over three cells using the first order approximation shown in these figures. For second order of accuracy, whose pressure contour is shown in Figure 7.13, the DG solver can capture the shock wave in a single cell. The pressure contour in the shock region is magnified in Figure 7.14, and we clearly see that the shock wave is constrained in a cell even though there are some high frequency errors in front of and behind the shock wave (This is due to the fact that the figure displayed is from cell-node averaged results and more will be discussed in the next section). For third order of accuracy, a negative pressure would occur in some cells near shock wave if some kind of limiting procedure is not employed and this could cause the numerical scheme to become unstable. In this research, we take the diffusion based artificial viscosity method by Persson and Peraire to damp the high frequency terms so as to achieve a single-cell resolution with the desired order of accuracy. The third order solution with this limiter is shown in Figure 7.15. The pressure contour in the shock region is magnified in Figure 7.16. Not

only is the shock wave captured in a single cell, but the length scale of this shock is thinner than the one from the second order case. This result indicates that the shock-capturing scheme with artificial viscosity is working for supersonic flow, therefore this strategy can be applied to a more sophisticated case of multiple species gas flow in the following.

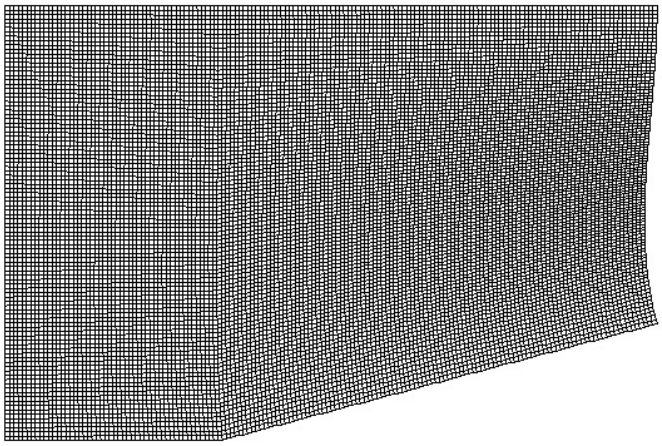

Figure 7.10

Geometry and meshes of supersonic flow passing a simple ramp

An important technique is introduced to display the results from high order DG method in high resolution mode. The figures we have seen so far are nodal averaged ones, for example the pressure contour just provided. For each node of the domain, values are computed from the several neighboring cells with DG approximation, then the arithmetically averaged output can be achieved. This way of displaying the flow field results can only provide roughly the mean of solution for each cell, and the high order component of the solution or the details of the polynomial approximation from the DG method can not be

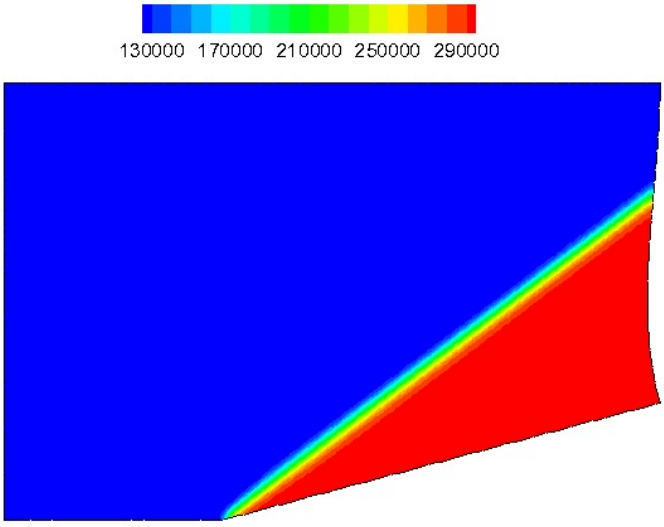

Figure 7.11

First order solution of pressure contour (cfl=5)

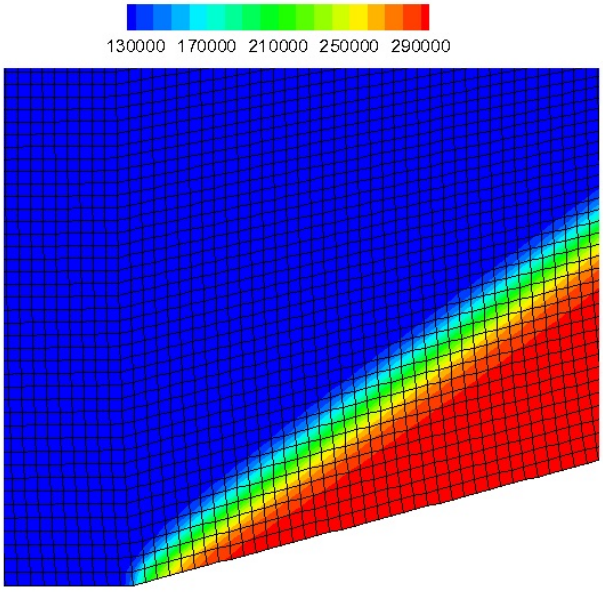

Figure 7.12

First order solution of pressure contour, magnified view near shock region with mesh displayed

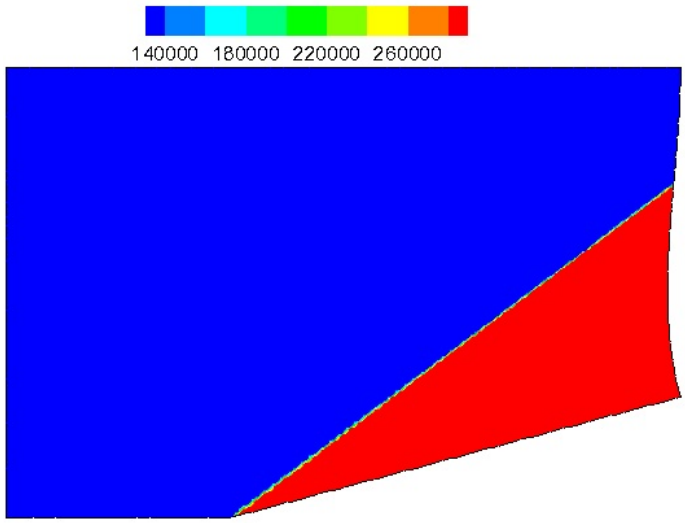

Figure 7.13

Second order solution of pressure contour (cfl=1)

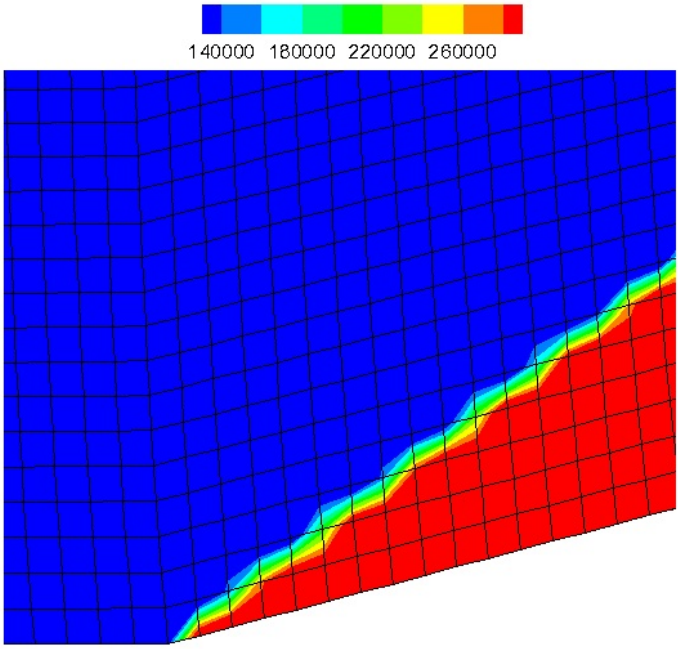

Figure 7.14

Second order solution of pressure contour, magnified view near shock region with mesh displayed

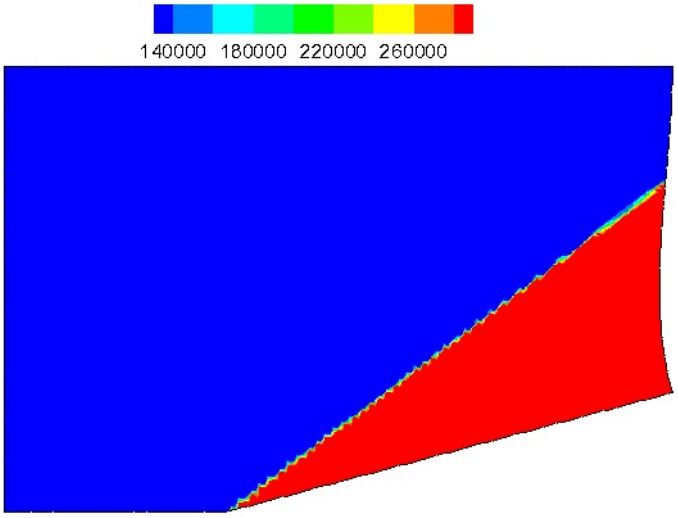

Figure 7.15

Third order solution of pressure contour (cfl=0.5)

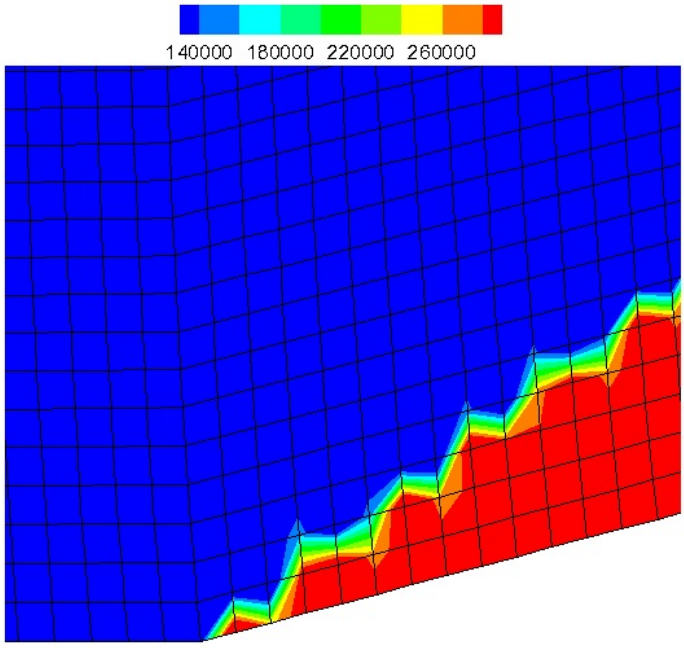

Figure 7.16

Third order solution of pressure contour, magnified view near shock region with mesh displayed

visualized in this simple way. Because the DG space discretization is implemented in this solver, then it is required that the visualization of the solution not only shows the basic information for each cell but also gives us more details about the high order components within the space inside each cell. Therefore we design a high resolution strategy and apply this to the updated solver to display the high order solution profile of each cell.

For the p-th order of accuracy obtained with DG method, each cell in the domain is divided into p^3 subcells for three-dimensional situations, and in this subdomain of each cell p^3 subcell nodes based solutions can be evaluated from the basis function approximations, according to the positions of these nodes. Finally we obtain the solution of the flow problems on the subcell nodal points, and from that the slope and changes of the polynomials can be viewed directly in the subdomain of each cell. This new strategy can help investigating the solution polynomial distribution inside each cell and understanding the formation of solution gradients in interesting regions, such as boundary layers.

The steady shock wave example studied above can be visualized again. This new technique of visualizing high resolution figures can be applied to the high order DG approximation. From the second order accurate high resolution pressure contour of Figure 7.17 and third order accurate high resolution pressure contour of Figure 7.18, we can clearly see that the shock profile is captured in a single cell and the width is decreasing as the order of accuracy becomes higher. The most important benefit is that the contour of the solution in each cell can be observed easily and the shock wave aligns properly with the cell interfaces, which shows the good directionality of this shock-capturing method.

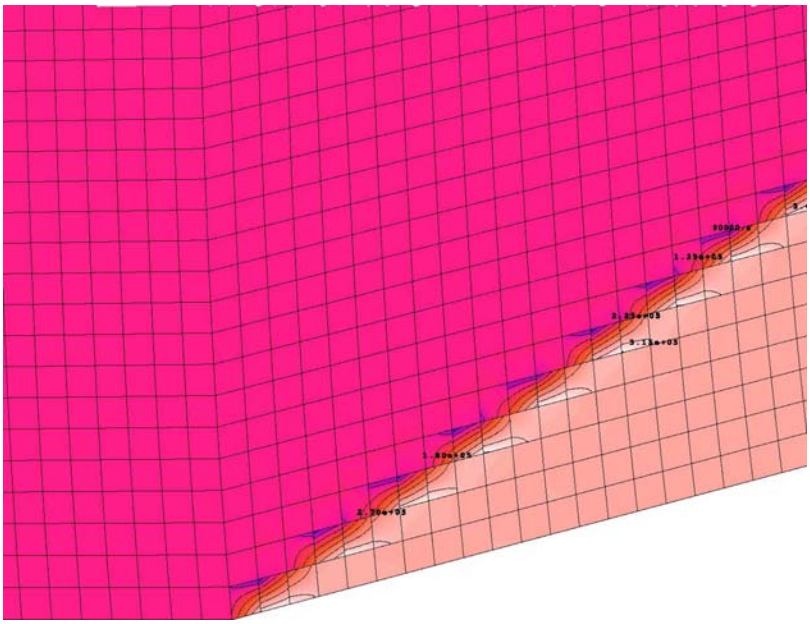

Figure 7.17

Second order accurate high resolution pressure profile

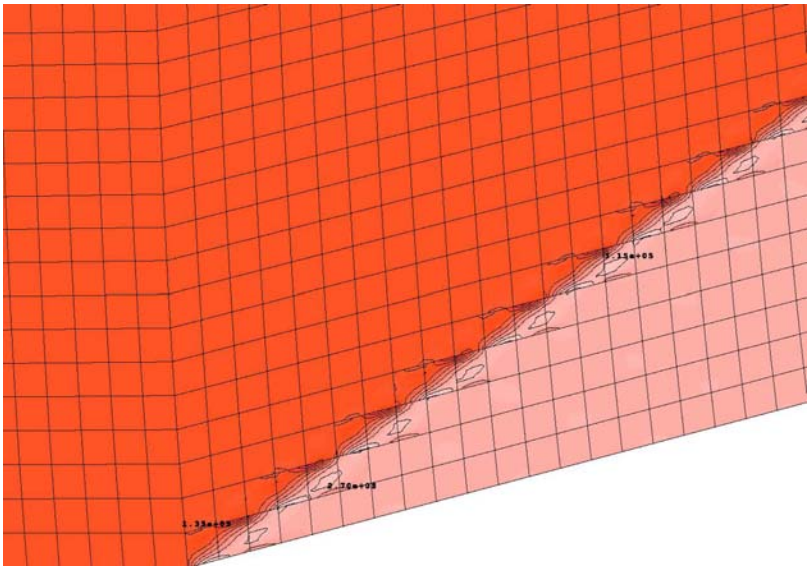

Figure 7.18

Third order accurate high resolution pressure profile

7.4 Supersonic shock capturing test cases

A typical unsteady shock wave problem is the shock-wedge reflection test case. The Euler equations are solved in time with the unsteady Runge-Kutta method, and Roe approximate Riemann scheme is used to compute convection flux with diffusion based limiter added to capture the shock wave. The test problem of shock-wedge reflection is described here by an incident shock wave ($Ms = 1.2$), initially perpendicular to the x-direction, encountering a wedge at an angle of $30\,^{\circ}$. A structured mesh is utilized for this implementation. Without using a limiter, the first order solution of Figure 7.19 and second order solution of Figure 7.20 are shown at some instant when shock wave arrives at the wedge. Moreover, the diffusion based limiter has been added to the third and fourth order solutions, and the results are shown in Figure 7.21 and Figure 7.22 respectively.

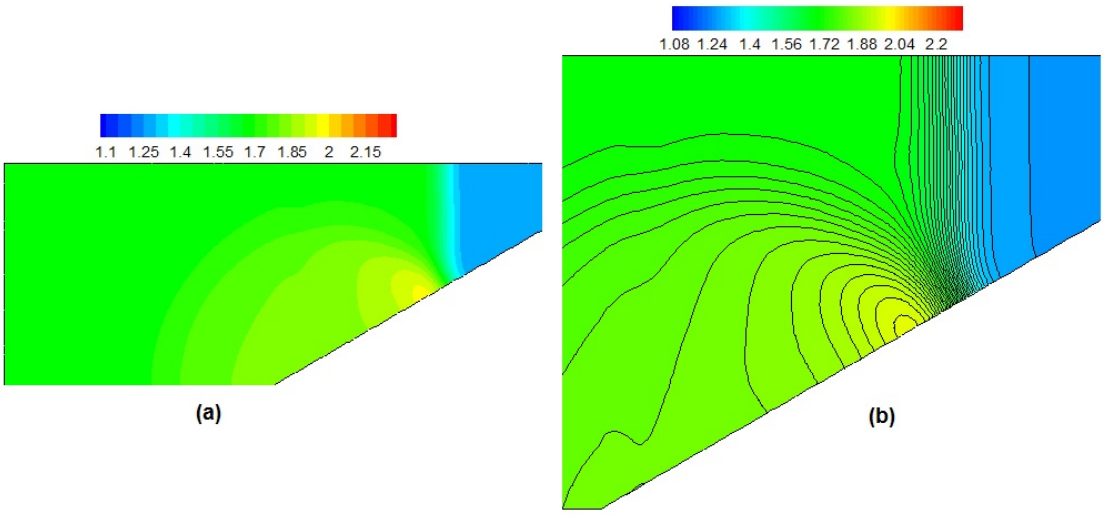

Figure 7.19

First order density contour of shock-wedge reflection (cfl=0.8)

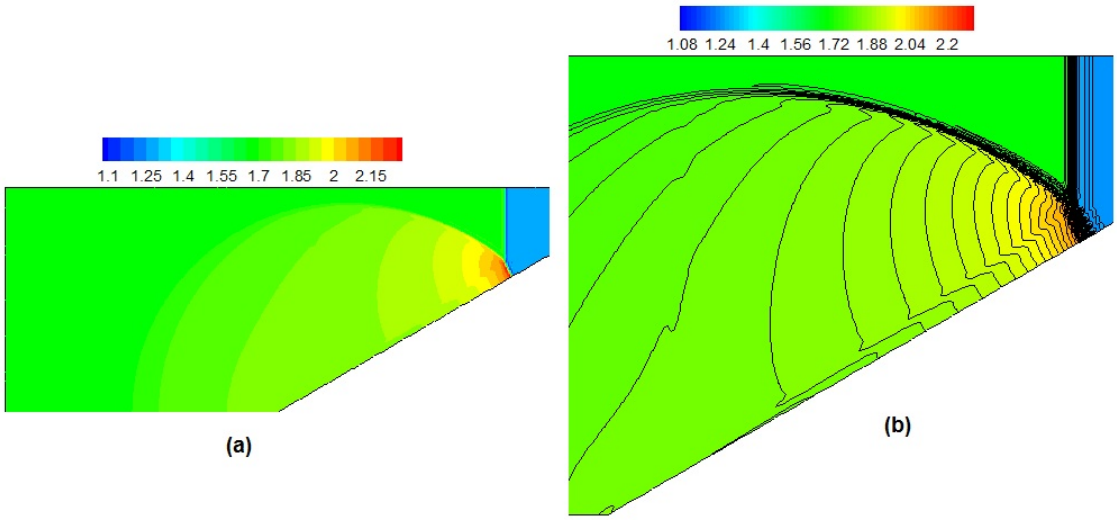

Figure 7.20

Second order density contour of shock-wedge reflection (cfl=0.5)

The solution corresponds to a single Mach reflection and the three shock waves (incident, reflected, and Mach stem) meet at the triple point. These numerical results can be compared to the results from [3]. A contact surface emerges from the triple point that joins the wedge surface at a sharp angle. The present results were found to be in very good agreement with other numerical results for high order approximations.

7.5 Laminar flow over a plate test cases

Solving the diffusion flux in the governing equations is an important task of this research. Because the diffusion based limiter is dependent on the implementation of the DG discretization of the diffusion flux, it is necessary to verify the second order derivative diffusion flux with the DG method. Laminar flow over a flat plate is a fundamental problem of fluid mechanics. The boundary layer is defined as the thin layer of the transition zone from

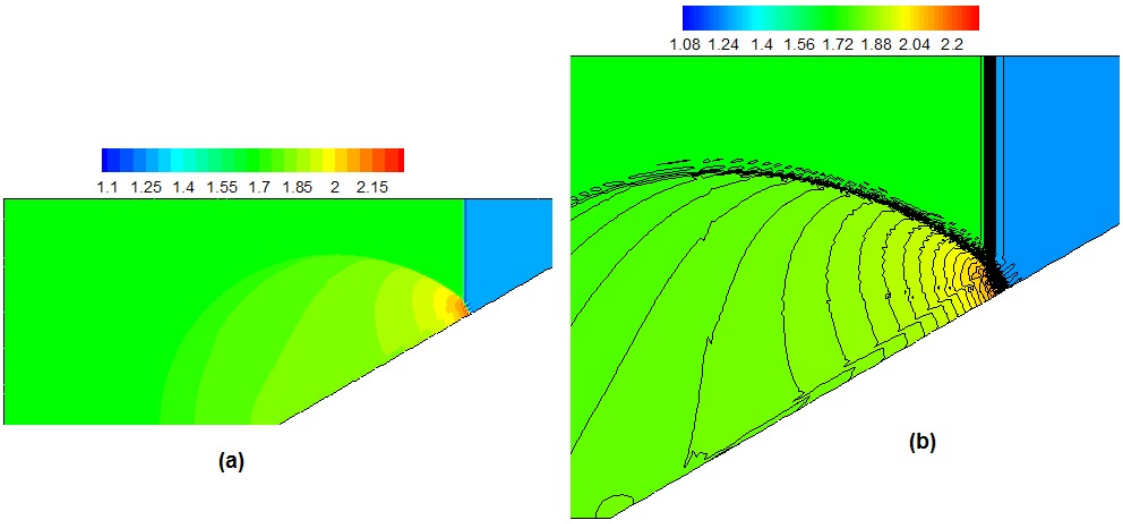

Figure 7.21

Third order density contour of shock-wedge reflection (cfl=0.1)

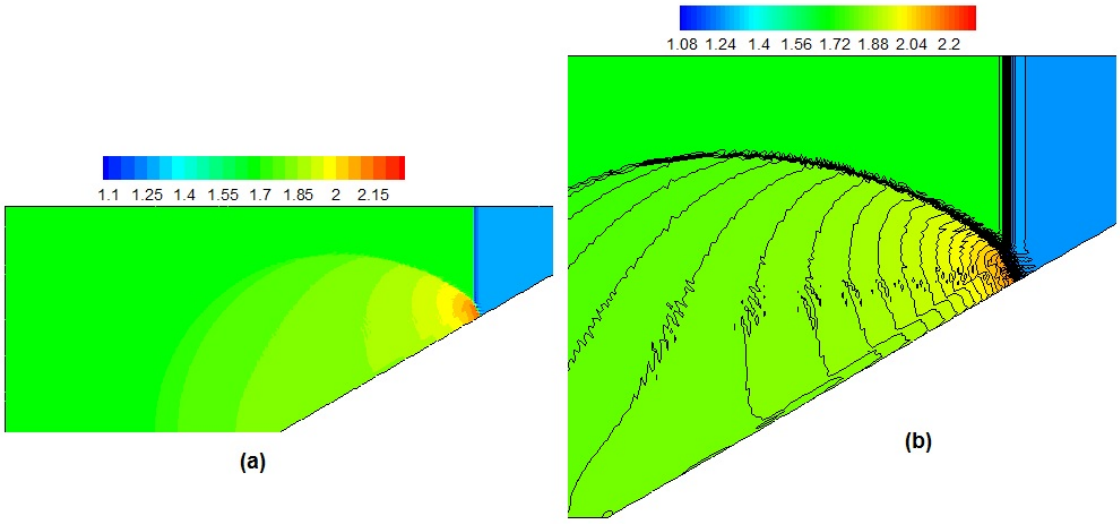

Figure 7.22

Fourth order density contour of shock-wedge reflection (cfl=0.1)

zero velocity at the flat plat to full small viscosity or large Reynolds number. The simplified Navier-Stokes equations yield approximate solutions of viscosity at high Reynolds number. In this case, the Navier-Stokes equations are solved for laminar flow, then two choices can be taken to discretize the diffusion flux component, one of Bassi and Rebay's BR2 method and another of Cockburn and Shu's LDG method. The numerical results are compared to the analytical solution from [58].

This study examines laminar flow over a plate. The incoming free stream is uniform with a Mach number of 0.1 and a Reynolds number of 200000. The free stream pressure is 41368.8Pa and temperature is 388.9K. A laminar boundary layer is formed at some point over the plate, and numerical solution with current solver are compared to the analytical Blasius [58] solution for some dimensionless similarity variables. The solutions of at most third order of accuracy are compared to Blasius solutions for x-direction velocity and wall friction coefficient in Figure 7.23 and Figure 7.24.

From above figures, we can see that the laminar flow problems including the diffusion contribution can be solved by BR2 or LDG methods, which verified the solver through this example. The second order derivative can be solved with both methods in DG framework, and the comparison to analytical solutions shows that the diffusion flux approximation is valid and can be applied to discretize the artificial diffusion in diffusion based limiter.

7.6 Multiple species test cases

As an important verification for the newly designed contact-surface-capturing scheme, the classical unsteady air-helium shock bubble problem is investigated with the high or-

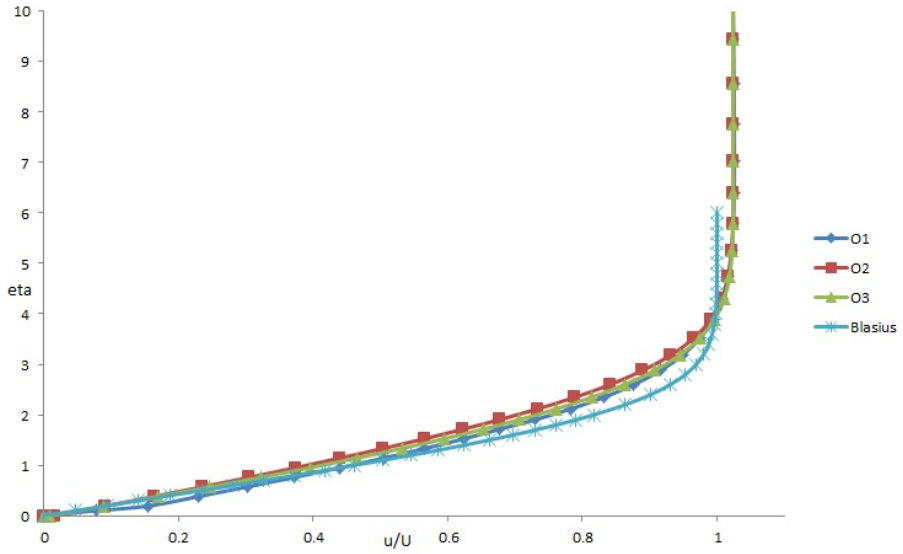

Figure 7.23

Comparison of the x-velocity at the end of the flat plate for different order of accuracy to analytical solution

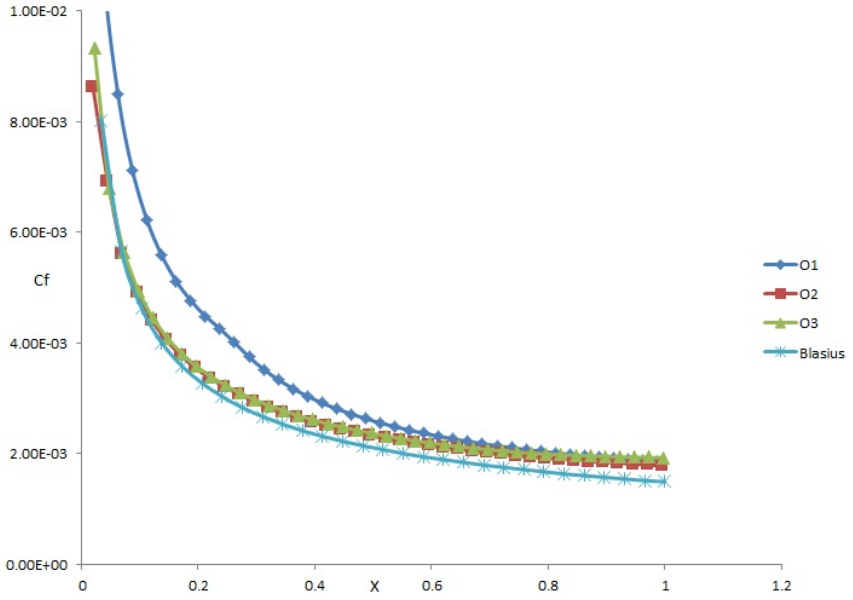

Figure 7.24

Comparison of the skin friction along the flat plate for different order of accuracy to analytical solution

der DG approximation for gas dynamics. The high order accurate numerical results are compared to experimental results in the following.

7.6.1 Problem description

From [3], we know that when two motionless gases are brought into contact, they mix by diffusion at a rate proportional to the product of the contact area and the concentration gradient at the interface. If the gases are set into motion by a passing shock wave, the mixing rate will increase because the motion produces an increase in the contact area. The bubble may be lighter or denser than the surrounding medium (we choose lighter bubble in the following). The shock-bubble interaction results in refraction, diffraction and reflection phenomena. Haas and Sturtevant [31] proposed the interaction of a planar shock wave with a bubble as a model problem for studying vorticity and turbulence generation in compressible flows with shock waves.

In the present study, a planar shock wave propagates through air from left to right of a tube and impinges on a cylindrical bubble. The cylindrical bubble has a radius of 25 mm and the vertical dimension of the shock tube is 89 mm. The horizontal dimension of the shock tube is 10 times the radius of the bubble. The incident shock is moving with a Mach number $Ms = 1.22$. Figure 7.25 is the schematic of the computational domain. Helium (molecular weight is 4) gas is used inside the bubble and air (molecular weight is 28.97) is used outside. In the computation, both structured and unstructured grids have been employed, but only the results from structured grids will be displayed here, which are similar to those from unstructured grids by comparison. In the following, several test cases

are carried out based on this configuration for different schemes, types of grids and orders of accuracy.

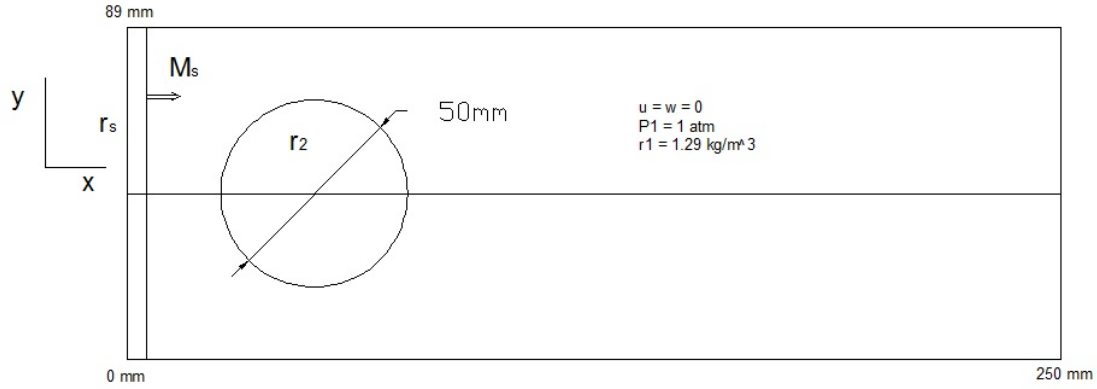

Figure 7.25

Schematic of the computational domain for the shock-bubble interaction problem

7.6.2 Case for comparing inviscid flux

First of all, a comparison of different inviscid numerical schemes for the air-helium shock bubble problem with the mass diffusion based limiter is carried out to find the best scheme for further study. Starting from the same initial condition at $Ms = 1.22$, we can compare the iso-density contour for each inviscid flux approximation scheme at some time after the shock interacts with bubble using second order of accuracy. At dimensionless time t=0.9 in Figure 7.26, the incident shock is reflected on the surface of the bubble as well as transmitted (refracted) through the bubble. At time t=3.8 in Figure 7.27, a kidney-shaped bubble is formed. At time t=7.0 in Figure 7.28, the VL-FVS scheme exhibits ripples at the

bubble interface because of instability, while other schemes can capture the bubble interface smoothly. At time t=10.2 in Figure 7.29, two separate vortices are formed when the jet vanished. From these comparisons, we find that different schemes have individual diffusive capability, and the large gradient at the interface could cause a Richtmyer-Meshkov [29] instability during the numerical computation process. The HLLC numerical scheme seems to capture the contact surface smoothly with better solution profiles. In the following implementation with the diffusion based limiter, we will use this method.

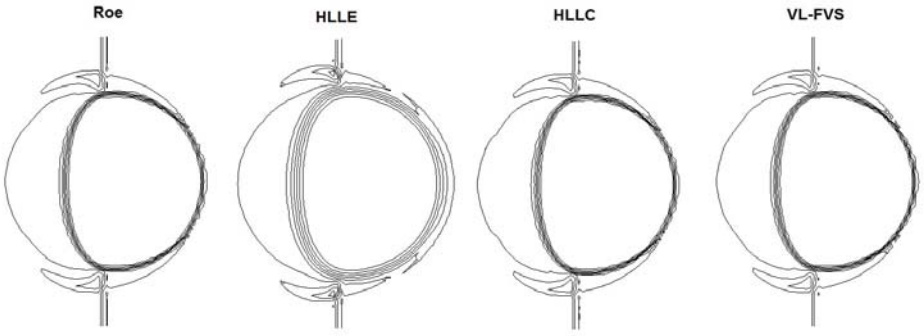

Figure 7.26

Iso-density contours at t=0.9 for different inviscid flux schemes

7.6.3 Testing discontinuity capturing case

The mass diffusion based limiter has been applied to the case of an unsteady shock wave passing a helium bubble. The explicit RK time integration and HLLC method for inviscid flux approximation are used, also the LDG method is used to compute the mass diffusion. The diffusion based limiter is employed for second, third, and fourth order accu-

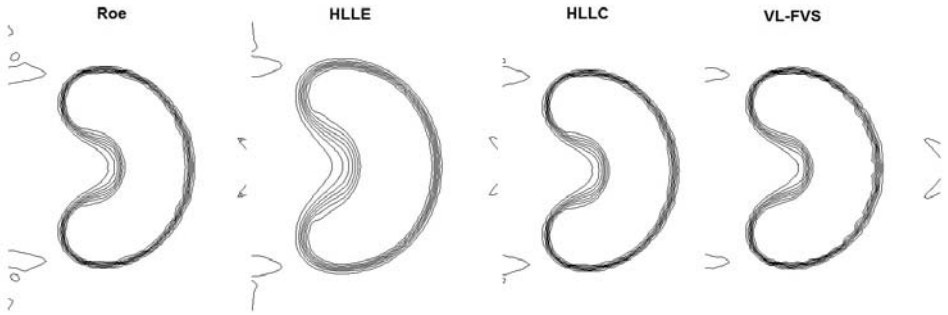

Figure 7.27

Iso-density contours at t=3.8 for different inviscid flux schemes

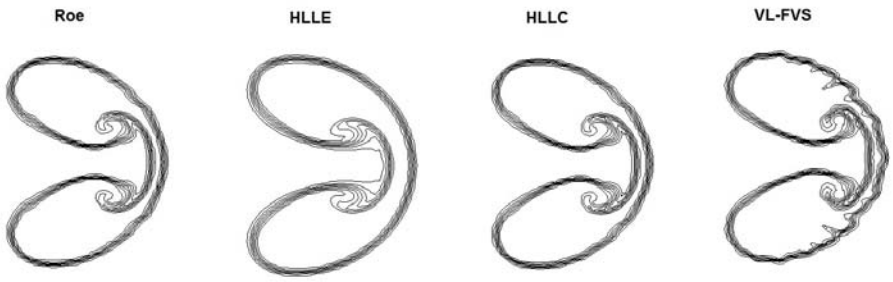

Figure 7.28

Iso-density contours at t=7.0 for different inviscid flux schemes

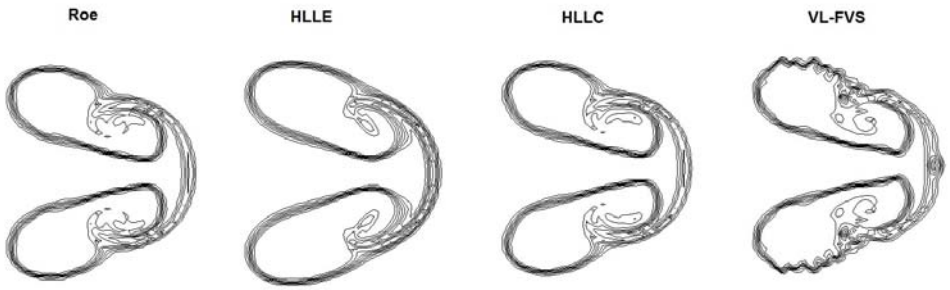

Figure 7.29

Iso-density contours at t=10.2 for different inviscid flux schemes

rate simulations. In order to enhance the comparison, we use the experimental results from Haas and Sturtevant [31]. In Figure 7.30 at t=32μs, experimental results show refracted wave (right) and reflected wave (left) and the motion of the upstream interface. In the numerical results, the waves can be seen for all situations and the interface is sharp for high order of accuracy. In Figure 7.31 at t=52μs, experimental results show a running refracted wave, a transmitted wave and a reflected wave. In the numerical results, the refracted wave is not very clear for second order of accuracy and is shown for higher order accurate cases. The reflected and transmitted waves are clear in the figure. In Figure 7.32 at t=62μs, experimental results show an entirely emerged transmitted wave from interface and shock-on-shock interaction. In the numerical results, these two phenomena can be clearly seen, especially at high order accurate results. In Figure 7.33 at t=72μs, experimental results show secondary transmitted waves. In numerical results, these can be seen for high order accurate cases. In Figure 7.34 at t=82μs, experimental results show two branches of the

secondary transmitted waves intersecting on the centreline. In numerical results, this can be seen clearly on all cases. In Figure 7.35 at t=102μs, experimental results show distortion and motion of the helium volume, and this is also easy seen from the numerical results. Afterwards, the incident shock is passing over the body, and the upstream face continues to deform so that a kidney-shaped volume is formed both in experimental and numerical results, as seen in Figure 7.36 at t=245μs. In Figure 7.37 at t=427μs, Figure 7.38 at t=674μs, and Figure 7.39 at t=983μs, a re-entrant jet forms both in experimental and numerical results. When the head of jet impinges the downstream interface, it spreads out and eventually forms a pair of vortical structures. From these figures, we can conclude that the scheme with high-order approximation can bring accurate numerical results with the diffusion based limiter.

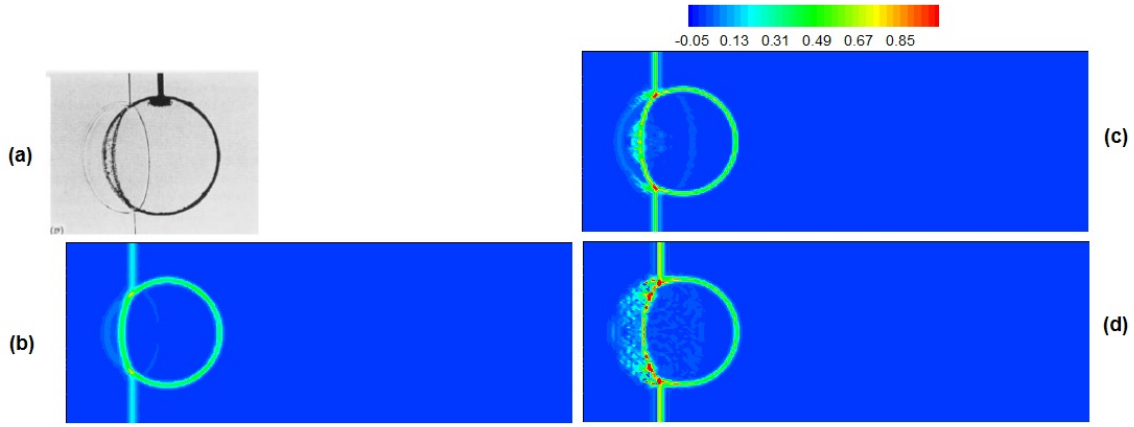

Figure 7.30

Comparison of density gradient at t=32μs:(a)experimental data (b)second order (c)third order (d)fourth order

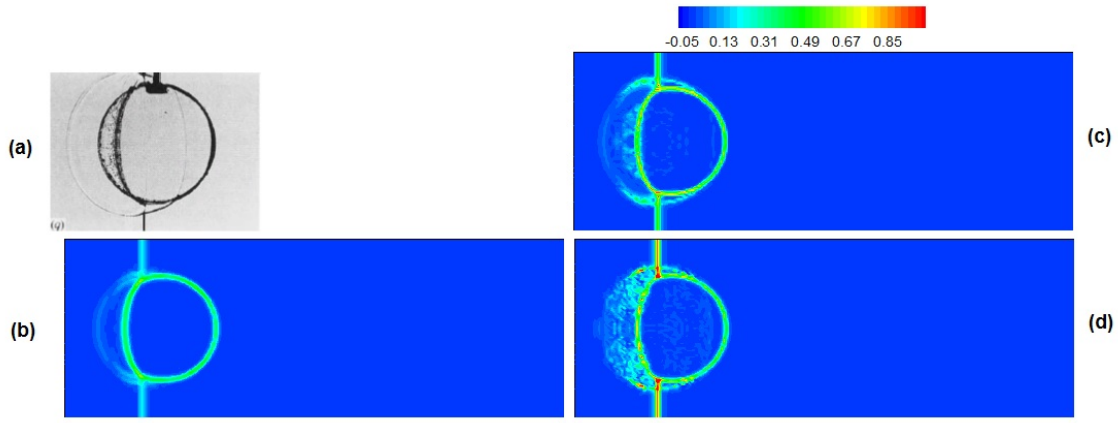

Figure 7.31

Comparison of density gradient at t=52μs:(a)experimental data (b)second order (c)third order (d)fourth order

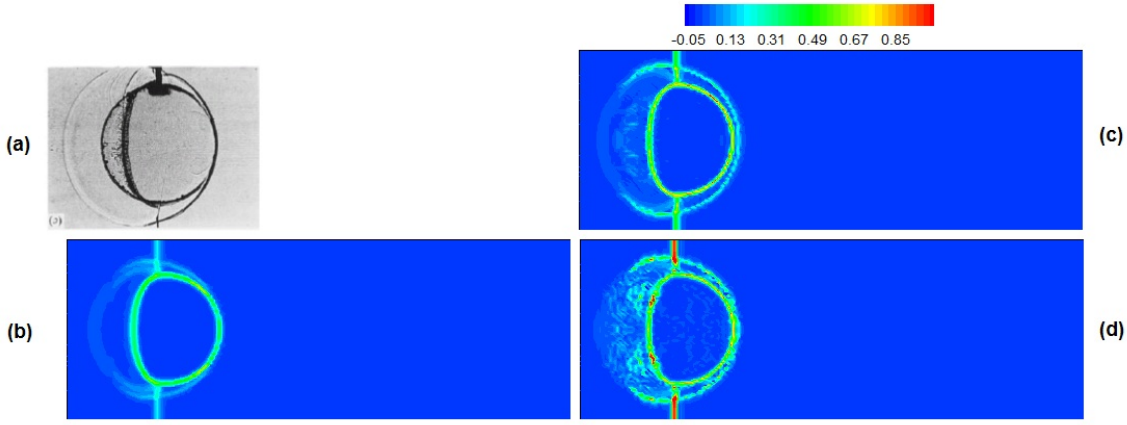

Figure 7.32

Comparison of density gradient at t=62μs:(a)experimental data (b)second order (c)third order (d)fourth order

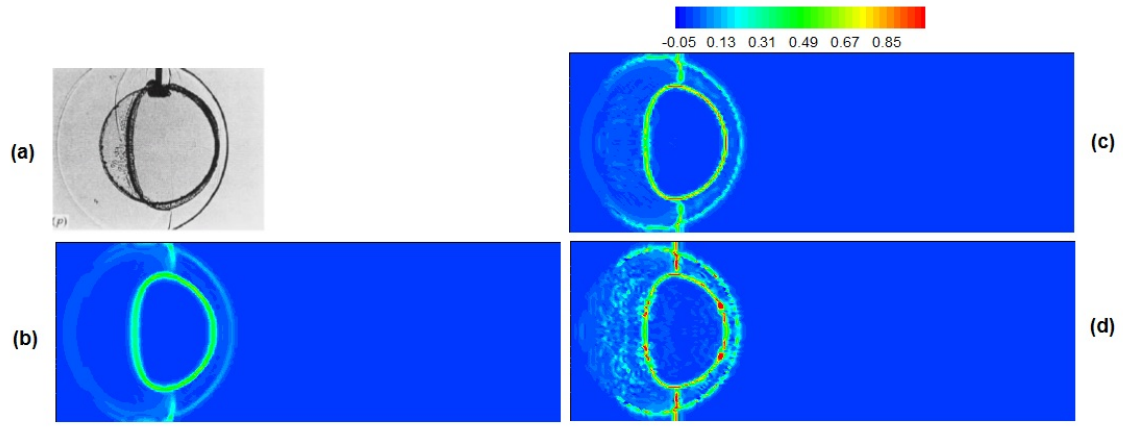

Figure 7.33

Comparison of density gradient at t=72μs:(a)experimental data (b)second order (c)third order (d)fourth order

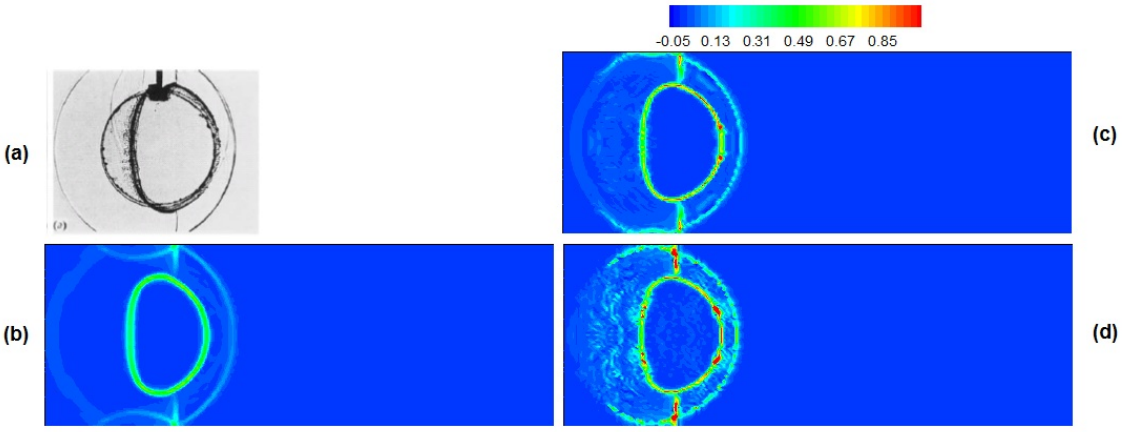

Figure 7.34

Comparison of density gradient at t=82μs:(a)experimental data (b)second order (c)third order (d)fourth order

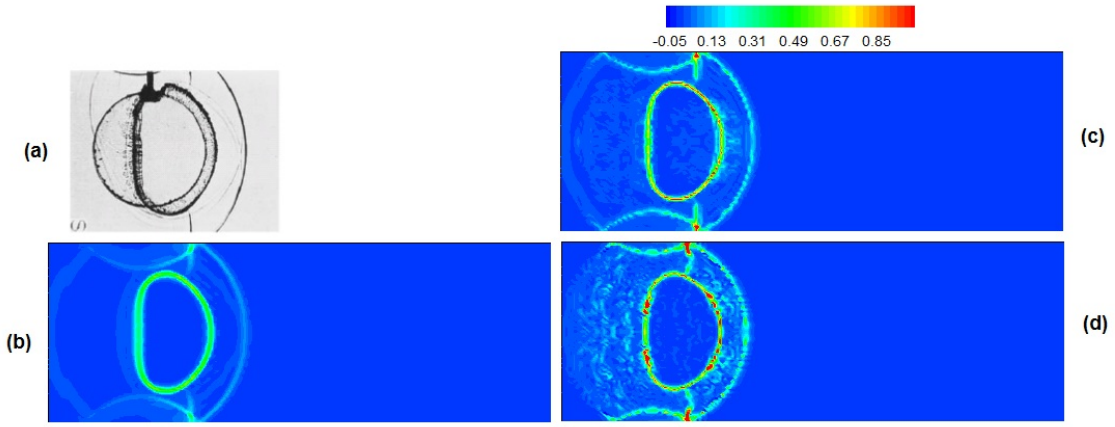

Figure 7.35

Comparison of density gradient at t=102μs:(a)experimental data (b)second order (c)third order (d)fourth order

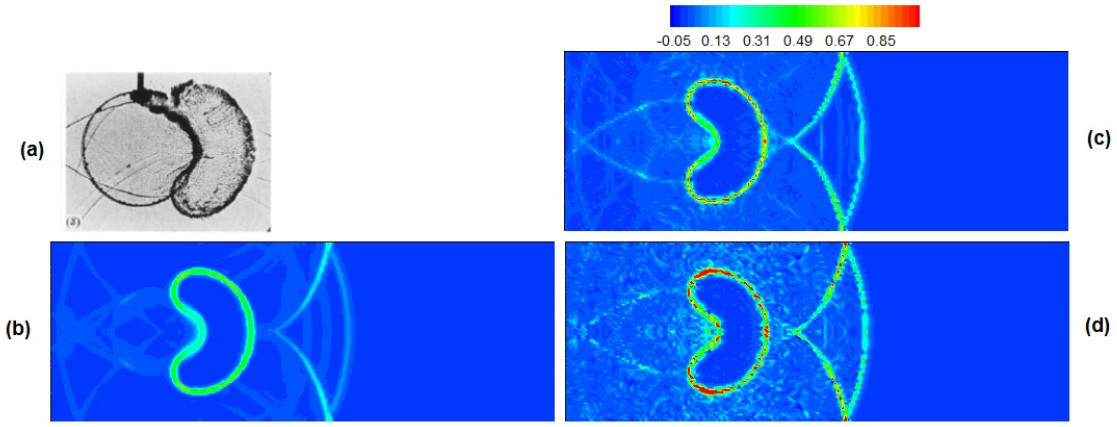

Figure 7.36

Comparison of density gradient at t=245μs:(a)experimental data (b)second order (c)third order (d)fourth order

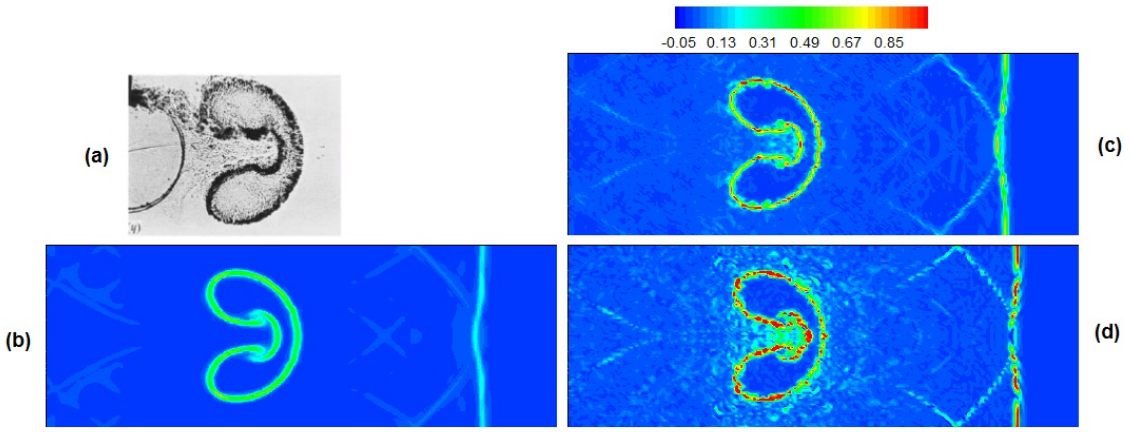

Figure 7.37

Comparison of density gradient at t=427μs:(a)experimental data (b)second order (c)third order (d)fourth order

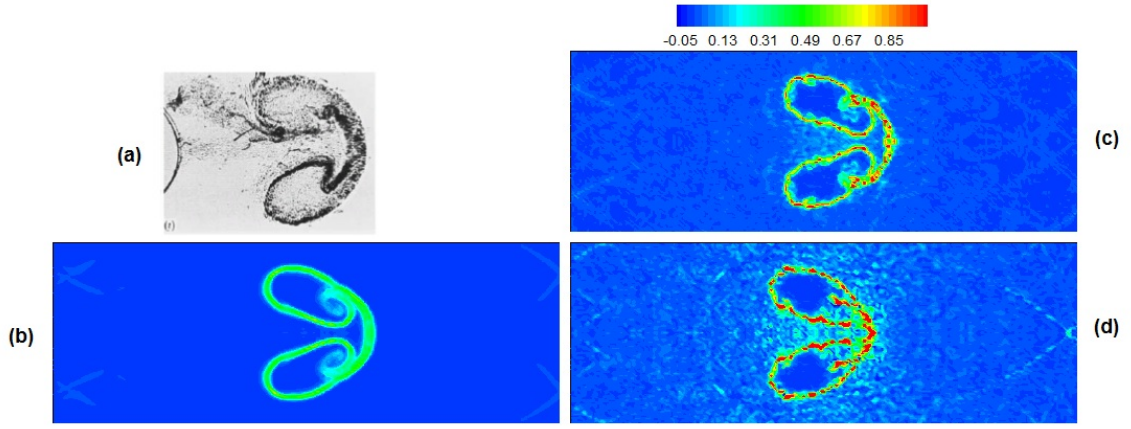

Figure 7.38

Comparison of density gradient at t=674μs:(a)experimental data (b)second order (c)third order (d)fourth order

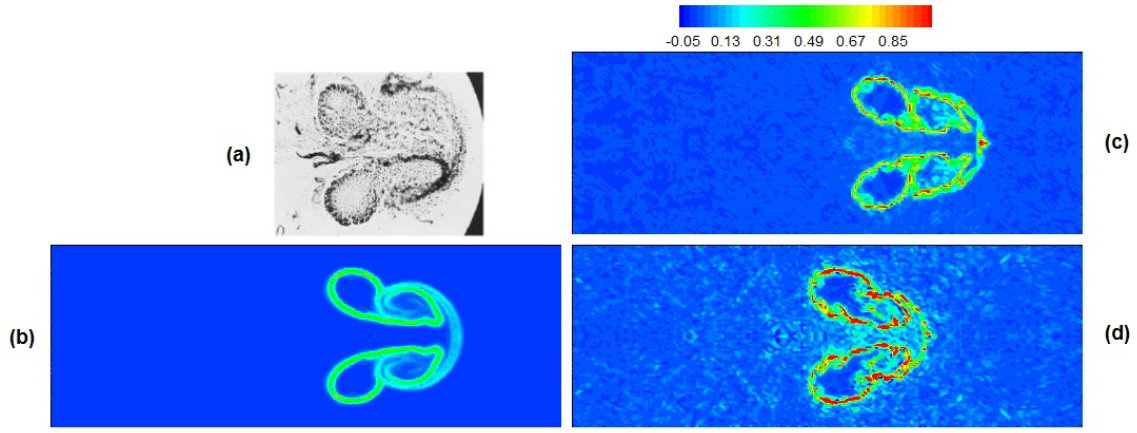

Figure 7.39

Comparison of density gradient at t=983μs:(a)experimental data (b)second order (c)third order (d)fourth order

7.6.4 Diffusion based limiter on high order accurate cases

Increasing the order of approximation, the contact discontinuity can be captured within only one cell. This can be verified from the numerical results of density for second order and third order of accuracy with the DG method in Figure 7.40 at t=245μs. From this figure, we can clearly find out that the species interface is sharp for the third order approximation. This is one of the benefits of the DG scheme with high order of accuracy. Additionally, one important point needs to be stated. Without the mass diffusion based limiter, the high-order scheme (for example, third order approximation) has large negative species mass fractions at interface with the results shown in Figure 7.41(a). In this plot, we notice that there are large negative mass fractions at the species interface, which are numerically accelerating the mixing process so as to cause instability of the numerical solver. On the other hand, when the proposed mass diffusion based limiter is considered in the

third order scheme with the same condition shown in Figure 7.41(b), the mass fraction of air is always maintained positive and the mass fraction interface is smooth. In this way, other gas properties are reasonable and the numerical algorithm is robust.

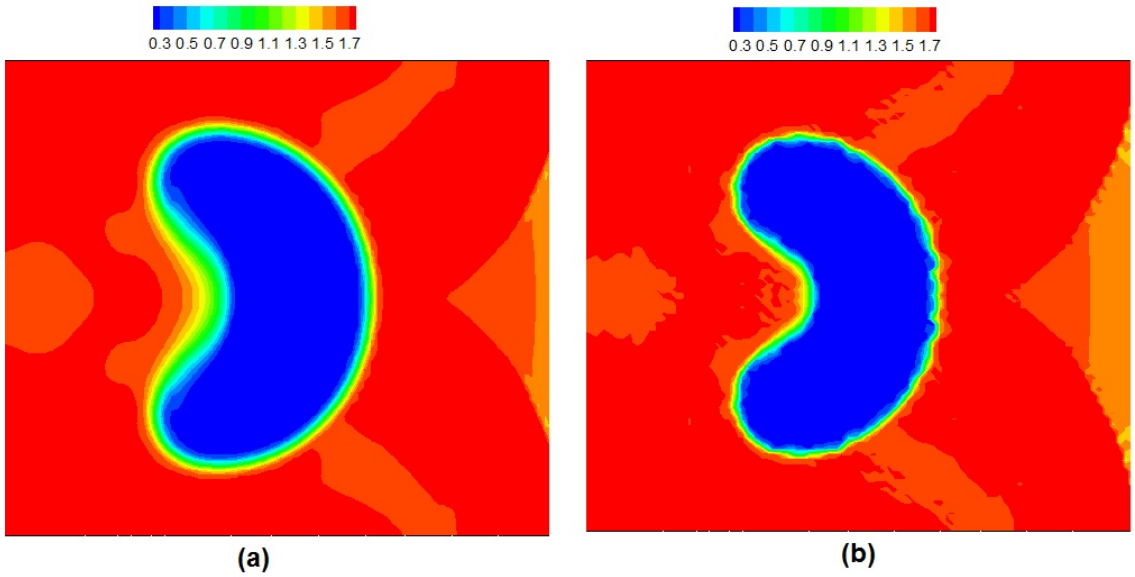

Figure 7.40

Comparison of density contour for high order of accuracy (a)second order (b)third order

The examples in this chapter proved that the proposed mass diffusion based limiter strategy can control the non-physical mass fraction so as to make the numerical scheme stable and robust.

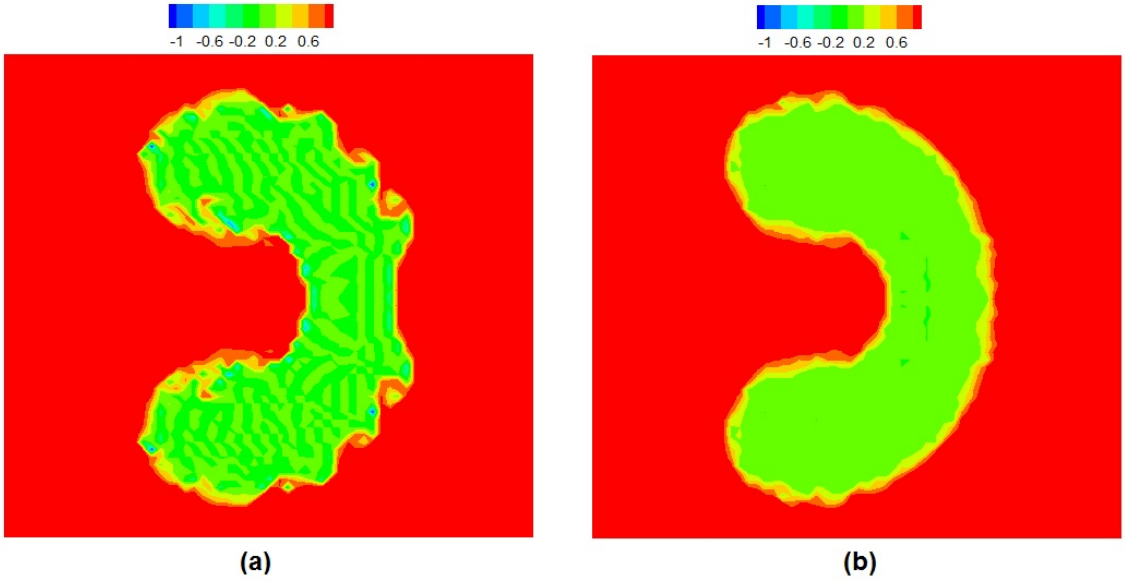

Figure 7.41

Mass diffusion based limiter effect from comparison of air negative mass fraction(Y_1)
(a)not using limiter (b)using proposed limiter

CHAPTER 8

CONCLUSIONS

The objective of this research has been achieved and the results are provided in this document. A high order DG solver has been set up for computations based on the Loci framework for simulations of three-dimensional flow problems. Several numerical methods have been verified. However, there is still some additional work needed to improve the solver. In the following, some conclusions are obtained and some long-term goals are presented.

8.1 Conclusions

A few conclusions can be drawn here, based on the work presented in this study.

8.1.1 Governing equations

The basic Euler governing equations are mainly used and RANS equations with turbulence model are implemented as well. The nondimensional governing equations are supplemented with some physical models. A two-equations turbulence model and a thermodynamic model for multiple species are introduced. Explicit time integration and semi-implicit p-multigrid algorithm are employed. Computing the Jacobian matrix in the fully

implicit method is too expensive, and so the p-multigrid method has the benefit of being relatively less expensive in solving the Jacobian matrix.

8.1.2 Discontinuous Galerkin method

The most important task of this research was building the numerical solver based on the DG method, which has actually been fulfilled. This solver provides a basic framework to do simulations of flow problems with high order of accuracy, and from the numerical results we conclude that the foundation of DG method is robust in real applications. Additionally, it is straightforward to increase the order of accuracy by providing more basis functions in quadrature rules. The DG method has been applied to both structured and unstructured grids successfully, but the computations on very complicated geometry with this solver should be further investigated. As the DG method becomes more popular, more functions and techniques could be developed based on this framework in future research.

8.1.3 Multiple species gas

A mixture of multiple species has been used as a simple model for general gas properties. For simplicity, a mixture of two thermally perfect gases is considered and the properties of each species are well maintained in the shock bubble test case since the species mass fraction is always positive to guarantee the gas properties to be physically meaningful. The extension of this model to a mixture of multiple species is straightforward, but considering thermo-chemical non-equilibrium process in a gas is still a challenging topic for further efforts.

8.1.4 Diffusion based limiter

The diffusion based limiter with high order DG method is successfully implemented for multiple species. When multiple species are separated by a contact surface, the solution gradient will not be large and the transition is smooth for species thermodynamic properties with high order DG approximation. From the numerical results, we observed that the new scheme is robust and can capture the contact discontinuity in a single cell with high order of accuracy. This is probably the most important contribution of the present study.

8.2 Future work

There are some research issues that can be further studied as an extension of this work. When computing the Jacobian matrix for the implicit time integration, the memory cost is very large and optimization of the algorithm is needed to reduce the memory requirements. The RANS equations with turbulence model have not been fully investigated in this research, and it would be interesting to concentrate on this topic. Currently the adaptive strategy has not been implemented in the mass diffusion based limiter, and the computational expense is relatively large as a result. Another long-term goal is to achieve the simulations of thermo-chemical non-equilibrium gas mixtures, for high temperature and high Mach number conditions with high order DG approximation, which could produce more accurate results based on the preliminary work shown in this document.

REFERENCES

[1] J. D. Anderson, *Modern compressible flow: with historical perspective*, McGraw-Hill, Inc., 1990.

[2] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, "Unified analysis of discontinuous Galerkin methods for elliptic problems," *SIAM J. Numer. Anal.*, vol. 39, no. 5, 2001, pp. 1749–1779.

[3] A. Bagabir and D. Drikakis, "Numerical experiments using high-resolution schemes for unsteady, inviscid, compressible flows," *Comp. Meth. in Appl. Mech. and Eng*, vol. 193, 2004, pp. 4675–4705.

[4] S. Balay, J. Brown, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, "PETSc Web page," 2012, http://www.mcs.anl.gov/petsc.

[5] G. E. Barter and D. L. Darmofal, "Shock capturing with higher-order, PDE-based artificial viscosity," *AIAA Paper 2007-3823*, 2007.

[6] F. Bassi, L. Botti, A. Colombo, A. Crivellini, N. Franchina, A. Ghidoni, and S. Rebay, "Very high-order accurate discontinuous Galerkin computation of transonic turbulent flows on aeronautical configurations," *ADIGMA A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications,*, N.Kroll, ed., NNFM 113, Springer, 2010, pp. 25–38.

[7] F. Bassi, A. Crivellini, A. Ghidoni, and S. Rebay, "High-order discontinuous Galerkin discretization of transonic turbulent flows," *AIAA Paper 2009-180*, Jan. 2009.

[8] F. Bassi, A. Crivellini, S. Rebay, and M. Savini, "Discontinuous Galerkin solution of the Reynolds-averaged Navier-Stokes and $\kappa - \omega$ turbulence model equations," *Computers and Fluids*, vol. 34, no. 4–5, 2005, pp. 507–540.

[9] F. Bassi, A. Ghidoni, S. Rebay, and P. Tesini, "High-order accurate p-multigrid discontinuous Galerkin solution of the Euler equations," *Int. J. Numer. Methods Fluids*, vol. 60, 2009, pp. 847–865.

[10] F. Bassi and S. Rebay, "High-order accurate discontinuous finite element solution of the 2D Euler equations," *J. Comput. Phys*, vol. 138, 1997, pp. 251–285.

[11] P. Batten, N. Clarke, C. Lambert, and D. M. Causon, "On the choice of wavespeeds for the HLLC Riemann solver," *SIAM J. Sci. Comput.*, vol. 18, no. 6, 1997, pp. 1553–1570.

[12] C. E. Baumann and J. T. Oden, "A discontinuous hp finite element method for the Euler and Navier-Stokes equations," *Int. J. Numer. Methods Fluids*, vol. 31, no. 1, 1999, pp. 79–95.

[13] R. A. Baurle, "Modeling of High Speed Reacting Flows: Established Practices and Future Challenges," *AIAA Paper 2004–0267*, Jan. 2004.

[14] J. A. Boles, J.-I. Choi, J. R. Edwards, and R. A. Baurle, "Simulations of High-Speed Internal Flows using LES/RANS Models," *AIAA Paper 2009–1324*, Jan. 2009.

[15] A. Burbeau, P. Sagaut, and C.-H. Bruneau, "A problem-independent limiter for high-order Runge-Kutta discontinuous Galerkin methods," *J. Comput. Phys.*, vol. 169, 2001, pp. 111–150.

[16] G. V. Candler, D. J. Mavriplis, and L. Trevino, "Current Status and Future Prospects for the Numerical Simulation of Hypersonic Flows," *AIAA Paper 2009-153*, Jan. 2009.

[17] J. Casper and H. L. Atkins, "A finite-volume high-order ENO scheme for two-dimensional hyperbolic systems," *J. Comput. Phys.*, vol. 106, 1993, pp. 62–76.

[18] B. Cockburn, "Discontinuous Galerkin methods," *ZAMM. Z. Angew. Math. Mech.*, vol. 83, no. 11, 2003, pp. 731–754.

[19] B. Cockburn, S. Hou, and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case," *Math. Comp.*, vol. 54, 1990, pp. 545–581.

[20] B. Cockburn, S. Y. Lin, and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems," *J. Comput. Phys.*, vol. 84, 1989, pp. 90–113.

[21] B. Cockburn and C.-W. Shu, "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws II: General framework," *Math. Comp.*, vol. 52, 1989, pp. 411–435.

[22] B. Cockburn and C.-W. Shu, "The local discontinuous Galerkin method for time-dependent convection-diffusion systems," *SIAM J. Numer. Anal*, vol. 35, 1998, pp. 2440–2463.

[23] B. Cockburn and C.-W. Shu, "Runge-Kutta discontinuous Galerkin methods for convection-dominated problems," *J. Sci. Comput.*, vol. 16, no. 3, 2001, pp. 173–261.

[24] M. G. Dunn and S. W. Kang, *Theoretical and Experimental Studies of Reentry Plasmas*, Tech. Rep., NASA CR-2232, 1973.

[25] B. Einfeldt, "On Godunov-type methods for gas dynamics," *SIAM J. Numer. Anal.*, vol. 25, no. 2, 1988, pp. 294–318.

[26] K. J. Fidkowski, T. A. Oliver, J. Lu, and D. L. Darmofal, "p-multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations," *J. Comput. Phys.*, vol. 207, 2005, pp. 92–113.

[27] B. G. Galerkin, "Rods and plates. Series in some problems of elastic equilibrium of rods and plates," *Vestn. Inzh. Tech.(USSR)*, 1915.

[28] V. K. Garg and A. A. Ameri, "Two-equation turbulence models for prediction of heat transfer on a transonic turbine blade," *Int. J. Heat and Fluid Flow*, vol. 22, 2001, pp. 593–602.

[29] J. Giordano and Y. Burtschell, "Richtmyer-Meshkov instability induced by shock-bubble interaction: Numerical and analytical studies with experimental validation," *Phys. Fluids*, vol. 18, no. 036102, 2006.

[30] B. Grossman and P. Cinnella, "The computation of non-equilibrium, chemically-reacting flows," *Computers & Structures*, vol. 30, no. 1/2, 1988, pp. 79–93.

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

[32] A. Harten and S. Osher, "Uniformly high-order accurate nonoscillatory schemes. I," *SIAM J. Numer. Anal.*, vol. 24, no. 2, 1987, pp. 279–309.

[33] R. Hartmann, "Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations," *Int. J. Numer. Methods Fluids*, vol. 51, 2006, pp. 1131–1156.

[34] C. Hirsch, *Numerical computation of internal and external flows. Vol. 2, Computational methods for inviscid and viscous flows*, John Wiley and Sons, Chichester, England and New York, NY, 1990.

[35] T. J. R. Hughes, M. Mallet, and A. Mizukami, "A new finite element formulation for computational fluid dynamics: II. Beyond SUPG," *Comput. Methods Appl. Mech. Engrg.*, vol. 54, no. 3, 1986, pp. 341–355.

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

[37] P. LeSaint and P. A. Raviart, *On a finite element method for solving the neutron transport equation*, In de Boor, C.(ed.), Mathematical Aspects of Finite Elements in Partial Differential Equations, Academic Press, New York, 1974, 89–145.

[38] Y. Liu, M. Vinokur, and Z. J. Wang, "Spectral difference method for unstructured grids I: Basic Formulation," *J. Comput. Physics*, vol. 216, 2006, pp. 780–801.

[39] I. Lomtev and G. E. Karniadakis, "A discontinuous Galerkin method for the Navier-Stokes equations," *Int. J. Numer. Meth. Fluids*, vol. 29, 1999, pp. 587–603.

[40] E. A. Luke, X. L. Tong, J. X. Wu, L. Tang, and P. Cinnella, "A Chemically Reacting Flow Solver for Generalized Grids," *AIAA J*, 2003.

[41] H. Luo, J. D. Baum, and R. Lohner, "A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids," *J. Comput. Phys.*, vol. 225, 2007, pp. 686–713.

[42] F. R. Menter, "Zonal two equation κ-ω turbulence models for aerodynamic flows," *AIAA Paper*, 1993, 2906.

[43] W. L. Oberkampf, M. N. Sindar, and A. T. Conlisk, *Guide for the verification and validation of computational fluid dynamics simulations*, Tech. Rep., American Institute of Aeronautics and Astronautics, 1998.

[44] P. Persson and J. Peraire, "Sub-cell shock capturing for discontinuous Galerkin methods," *AIAA Paper 2006-0112*, Jan. 2006.

[45] L. A. Piegl and W. Tiller, *The Nurbs Book*, Springer-Verlag, New York, NY, 1997.

[46] W. H. Reed and T. R. Hill, *Triangular mesh methods for the neutron transport equation*, Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[47] P. L. Roe, "Approximate Riemann solvers, parameter vectors and difference schemes," *J. Comput. Physics*, vol. 43, 1981, pp. 357–372.

[48] J. S. Shang, S. T. Surzhikov, and H. Yan, "Simulate Hypersonic Nonequilibrium Flow Using Kinetic Models," *AIAA Paper 2009-386*, Jan. 2009.

[49] C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes," *J. Comput. Phys.*, vol. 77, no. 2, 1988, pp. 439–471.

[50] G. A. Sod, "A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws," *J. Comput. Phys.*, vol. 27, no. 1, 1978, pp. 1–31.

[51] E. F. Toro, *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*, Springer-Verlag, Berlin, Heidelberg, 1997.

[52] E. F. Toro, M. Spruce, and W. Speares, "Restoration of the contact surface in the HLL-Riemann solver," *Shock Waves J.*, vol. 4, 1994, pp. 25–34.

[53] B. van Leer, "Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme," *J. Comput. Phys.*, vol. 14, no. 4, 1974, pp. 361–370.

[54] B. van Leer, "Flux Vector Splitting for the Euler Equations," *Proceedings: 8th International Conference on Numerical Methods in Fluid Dynamics*, Aachen, 1982, Springer-Verlag Lecture Notes in Physics, pp. 507–512, edited by E. Krause.

[55] J. VonNeumann and R. D. Richtmyer, "A method for the numerical calculation of hydrodynamic shocks," *J. Appl. Phys.*, vol. 21, 1950, pp. 232–237.

[56] R. W. Walters, P. Cinnella, D. C. Slack, and D. Halt, "Characteristic-Based Algorithms for Flows in Thermochemical Nonequilibrium," *AIAA J*, vol. 30, no. 5, May 1992, pp. 1304–1313.

[57] T. C. Warburton, I. Lomtev, Y. Du, S. J. Sherwin, and G. E. Karniadakis, "Galerkin and Discontinuous Galerkin Spectral/hp Methods," *Comput. Methods Appl. Mech. Engrg.*, vol. 175, 1999, pp. 343–359.

[58] F. F. White, *Viscous Fluid Flow*, McGraw-Hill, Inc., Boston, 1991.

[59] D. C. Wilcox, *Turbulence modelling for CFD*, DCW industries Inc., La Canada, CA91011, USA, 1993.

[60] O. C. Zienkiewicz, R. L. Taylor, S. J. Sherwin, and J. Peiro, "On discontinuous Galerkin methods," *Int. J. Numer. Meth. Engng*, vol. 58, 2003, pp. 1119–1148.

APPENDIX A

BASIS FUNCTIONS

A.1 Basis functions

In this section the basis functions of hexahedral element for each degree of freedom in the implementation are listed for clarity. These symbolic formulae are the results of Equation (3.2) and Equation (3.3) with the help of the Maple mathematic software package.

Table A.1

Basis functions for reference hexahedra (DOF: degree of freedom)

DOF/Order	1	2	3	4
0	1	1	1	1
1		ξ_1	ξ_1	ξ_1
2		ξ_2	ξ_2	ξ_2
3		ξ_3	ξ_3	ξ_3
4		$\xi_1\xi_2$	$\xi_1\xi_2$	$\xi_1\xi_2$
5		$\xi_1\xi_3$	$\xi_1\xi_3$	$\xi_1\xi_3$
6		$\xi_2\xi_3$	$\xi_2\xi_3$	$\xi_2\xi_3$
7		$\xi_1\xi_2\xi_3$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2\xi_3$
8			$-\frac{1}{2}+\frac{3}{2}\xi_3^2$	$-\frac{1}{2}+\frac{3}{2}\xi_3^2$
9			$-\frac{1}{2}\xi_2+\frac{3}{2}\xi_2\xi_3^2$	$-\frac{1}{2}\xi_2+\frac{3}{2}\xi_2\xi_3^2$
10			$-\frac{1}{2}+\frac{3}{2}\xi_2^2$	$-\frac{1}{2}+\frac{3}{2}\xi_2^2$
11			$\frac{3}{2}\xi_3\xi_2^2-\frac{1}{2}\xi_3$	$\frac{3}{2}\xi_3\xi_2^2-\frac{1}{2}\xi_3$
12			$-\frac{3}{4}\xi_2^2+\frac{9}{4}\xi_2^2\xi_3^2+$ $\frac{1}{4}-\frac{3}{4}\xi_3^2$	$-\frac{3}{4}\xi_2^2+\frac{9}{4}\xi_2^2\xi_3^2+$ $\frac{1}{4}-\frac{3}{4}\xi_3^2$
13			$-\frac{1}{2}\xi_1+\frac{3}{2}\xi_1\xi_3^2$	$-\frac{1}{2}\xi_1+\frac{3}{2}\xi_1\xi_3^2$
14			$-\frac{1}{2}\xi_1\xi_2+\frac{3}{2}\xi_1\xi_2\xi_3^2$	$-\frac{1}{2}\xi_1\xi_2+\frac{3}{2}\xi_1\xi_2\xi_3^2$
15			$-\frac{1}{2}\xi_1+\frac{3}{2}\xi_1\xi_2^2$	$-\frac{1}{2}\xi_1+\frac{3}{2}\xi_1\xi_2^2$

Table A.1

(continued)

DOF/Order	1	2	3	4
16			$\frac{3}{2}\xi_1\xi_3\xi_2^2 - \frac{1}{2}\xi_1\xi_3$	$\frac{3}{2}\xi_1\xi_3\xi_2^2 - \frac{1}{2}\xi_1\xi_3$
17			$-\frac{3}{4}\xi_1\xi_2^2 + \frac{9}{4}\xi_1\xi_2^2\xi_3^2 +$ $\frac{1}{4}\xi_1 - \frac{3}{4}\xi_1\xi_3^2$	$-\frac{3}{4}\xi_1\xi_2^2 + \frac{9}{4}\xi_1\xi_2^2\xi_3^2 +$ $\frac{1}{4}\xi_1 - \frac{3}{4}\xi_1\xi_3^2$
18			$-\frac{1}{2} + \frac{3}{2}\xi_1^2$	$-\frac{1}{2} + \frac{3}{2}\xi_1^2$
19			$\frac{3}{2}\xi_3\xi_1^2 - \frac{1}{2}\xi_3$	$\frac{3}{2}\xi_3\xi_1^2 - \frac{1}{2}\xi_3$
20			$-\frac{3}{4}\xi_1^2 + \frac{9}{4}\xi_1^2\xi_3^2 +$ $\frac{1}{4} - \frac{3}{4}\xi_3^2$	$-\frac{3}{4}\xi_1^2 + \frac{9}{4}\xi_1^2\xi_3^2 +$ $\frac{1}{4} - \frac{3}{4}\xi_3^2$
21			$\frac{3}{2}\xi_2\xi_1^2 - \frac{1}{2}\xi_2$	$\frac{3}{2}\xi_2\xi_1^2 - \frac{1}{2}\xi_2$
22			$\frac{3}{2}\xi_2\xi_3\xi_1^2 - \frac{1}{2}\xi_2\xi_3$	$\frac{3}{2}\xi_2\xi_3\xi_1^2 - \frac{1}{2}\xi_2\xi_3$
23			$-\frac{3}{4}\xi_2\xi_1^2 + \frac{9}{4}\xi_2\xi_1^2\xi_3^2 +$ $\frac{1}{4}\xi_2 - \frac{3}{4}\xi_2\xi_3^2$	$-\frac{3}{4}\xi_2\xi_1^2 + \frac{9}{4}\xi_2\xi_1^2\xi_3^2 +$ $\frac{1}{4}\xi_2 - \frac{3}{4}\xi_2\xi_3^2$
24			$-\frac{3}{4}\xi_1^2 + \frac{9}{4}\xi_1^2\xi_2^2 +$ $\frac{1}{4} - \frac{3}{4}\xi_2^2$	$-\frac{3}{4}\xi_1^2 + \frac{9}{4}\xi_1^2\xi_2^2 +$ $\frac{1}{4} - \frac{3}{4}\xi_2^2$
25			$\frac{9}{4}\xi_1^2\xi_3\xi_2^2 - \frac{3}{4}\xi_1^2\xi_3 -$ $\frac{3}{4}\xi_3\xi_2^2 + \frac{1}{4}\xi_3$	$\frac{9}{4}\xi_1^2\xi_3\xi_2^2 - \frac{3}{4}\xi_1^2\xi_3 -$ $\frac{3}{4}\xi_3\xi_2^2 + \frac{1}{4}\xi_3$
26			$-\frac{9}{8}\xi_1^2\xi_2^2 + \frac{27}{8}\xi_1^2\xi_2^2\xi_3^2 +$ $\frac{3}{8}\xi_1^2 - \frac{9}{8}\xi_1^2\xi_3^2 + \frac{3}{8}\xi_2^2$ $-\frac{9}{8}\xi_2^2\xi_3^2 - \frac{1}{8} + \frac{3}{8}\xi_3^2$	$-\frac{9}{8}\xi_1^2\xi_2^2 + \frac{27}{8}\xi_1^2\xi_2^2\xi_3^2 +$ $\frac{3}{8}\xi_1^2 - \frac{9}{8}\xi_1^2\xi_3^2 + \frac{3}{8}\xi_2^2$ $-\frac{9}{8}\xi_2^2\xi_3^2 - \frac{1}{8} + \frac{3}{8}\xi_3^2$

Table A.1

(continued)

DOF/Order	1	2	3	4
27				$-\frac{3}{2}\xi_3+\frac{5}{2}\xi_3^3$
28				$-\frac{3}{2}\xi_2\xi_3+\frac{5}{2}\xi_2\xi_3^3$
29				$\frac{15}{4}\xi_2^2\xi_3^3-\frac{9}{4}\xi_3\xi_2^2-\frac{5}{4}\xi_3^3+\frac{3}{4}\xi_3$
30				$-\frac{3}{2}\xi_2+\frac{5}{2}\xi_2^3$
31				$\frac{5}{2}\xi_3\xi_2^3-\frac{3}{2}\xi_2\xi_3$
32				$-\frac{5}{4}\xi_2^3+\frac{15}{4}\xi_2^3\xi_3^2+\frac{3}{4}\xi_2-\frac{9}{4}\xi_2\xi_3^2$
33				$\frac{25}{4}\xi_2^3\xi_3^3-\frac{15}{4}\xi_2^3\xi_3-\frac{15}{4}\xi_2\xi_3^3+\frac{9}{4}\xi_2\xi_3$
34				$-\frac{3}{2}\xi_1\xi_3+\frac{5}{2}\xi_1\xi_3^3$
35				$-\frac{3}{2}\xi_1\xi_2\xi_3+\frac{5}{2}\xi_1\xi_2\xi_3^3$
36				$\frac{15}{4}\xi_1\xi_2^2\xi_3^3-\frac{9}{4}\xi_1\xi_3\xi_2^2-\frac{5}{4}\xi_1\xi_3^3+\frac{3}{4}\xi_1\xi_3$
37				$-\frac{3}{2}\xi_1\xi_2+\frac{5}{2}\xi_1\xi_2^3$
38				$\frac{5}{2}\xi_1\xi_3\xi_2^3-\frac{3}{2}\xi_1\xi_3\xi_2$
39				$-\frac{5}{4}\xi_1\xi_2^3+\frac{15}{4}\xi_1\xi_2^3\xi_3^2+\frac{3}{4}\xi_1\xi_2-\frac{9}{4}\xi_1\xi_2\xi_3^2$
40				$\frac{25}{4}\xi_1\xi_2^3\xi_3^3-\frac{15}{4}\xi_1\xi_3\xi_2^3-\frac{15}{4}\xi_1\xi_2\xi_3^3+\frac{9}{4}\xi_1\xi_3\xi_2$
41				$\frac{15}{4}\xi_1^2\xi_3^3-\frac{9}{4}\xi_3\xi_1^2-\frac{5}{4}\xi_3^3+\frac{3}{4}\xi_3$
42				$\frac{15}{4}\xi_2\xi_1^2\xi_3^3-\frac{9}{4}\xi_2\xi_1^2\xi_3-\frac{5}{4}\xi_2\xi_3^3+\frac{3}{4}\xi_2\xi_3$
43				$\frac{45}{8}\xi_1^2\xi_2^2\xi_3^3-\frac{27}{8}\xi_1^2\xi_3\xi_2^2-\frac{15}{8}\xi_1^2\xi_3^3+\frac{9}{8}\xi_1^2\xi_3$ $-\frac{15}{8}\xi_2^2\xi_3^3+\frac{9}{8}\xi_3\xi_2^2+\frac{5}{8}\xi_3^3-\frac{3}{8}\xi_3$
44				$\frac{15}{4}\xi_1^2\xi_2^3-\frac{9}{4}\xi_1^2\xi_2-\frac{5}{4}\xi_2^3+\frac{3}{4}\xi_2$
45				$\frac{15}{4}\xi_1^2\xi_3\xi_2^3-\frac{9}{4}\xi_1^2\xi_3\xi_2-\frac{5}{4}\xi_3\xi_2^3+\frac{3}{4}\xi_3\xi_2$
46				$-\frac{15}{8}\xi_1^2\xi_2^3+\frac{45}{8}\xi_1^2\xi_2^3\xi_3^2+\frac{9}{8}\xi_1^2\xi_2-\frac{27}{8}\xi_1^2\xi_2\xi_3^2$ $+\frac{5}{8}\xi_2^3-\frac{15}{8}\xi_2^3\xi_3^2-\frac{3}{8}\xi_2+\frac{9}{8}\xi_2\xi_3^2$
47				$\frac{75}{8}\xi_1^2\xi_2^3\xi_3^3-\frac{45}{8}\xi_1^2\xi_3\xi_2^3-\frac{45}{8}\xi_1^2\xi_2\xi_3^3+\frac{27}{8}\xi_1^2\xi_3\xi_2$ $-\frac{25}{8}\xi_2^3\xi_3^3+\frac{15}{8}\xi_3\xi_2^3+\frac{15}{8}\xi_2\xi_3^3-\frac{9}{8}\xi_3\xi_2$

Table A.1

(continued)

DOF/Order	1	2	3	4
48				$-\frac{3}{2}\xi_1+\frac{5}{2}\xi_1^3$
49				$\frac{5}{2}\xi_3\xi_1^3-\frac{3}{2}\xi_3\xi_1$
50				$-\frac{5}{4}\xi_1^3+\frac{15}{4}\xi_1^3\xi_3^2+\frac{3}{4}\xi_1-\frac{9}{4}\xi_1\xi_3^2$
51				$\frac{25}{4}\xi_1^3\xi_3^3-\frac{15}{4}\xi_3\xi_1^3-\frac{15}{4}\xi_1\xi_3^3+\frac{9}{4}\xi_3\xi_1$
52				$\frac{5}{2}\xi_2\xi_1^3-\frac{3}{2}\xi_2\xi_1$
53				$\frac{5}{2}\xi_2\xi_3\xi_1^3-\frac{3}{2}\xi_2\xi_3\xi_1$
54				$-\frac{5}{4}\xi_2\xi_1^3+\frac{15}{4}\xi_2\xi_1^3\xi_3^2+\frac{3}{4}\xi_2\xi_1-\frac{9}{4}\xi_2\xi_1\xi_3^2$
55				$\frac{25}{4}\xi_2\xi_1^3\xi_3^3-\frac{15}{4}\xi_2\xi_1^3\xi_3-\frac{15}{4}\xi_2\xi_1\xi_3^3+\frac{9}{4}\xi_2\xi_1\xi_3$
56				$-\frac{5}{4}\xi_1^3+\frac{15}{4}\xi_1^3\xi_2^2+\frac{3}{4}\xi_1-\frac{9}{4}\xi_1\xi_2^2$
57				$\frac{15}{4}\xi_1^3\xi_3\xi_2^2-\frac{5}{4}\xi_1^3\xi_3-\frac{9}{4}\xi_1\xi_3\xi_2^2+\frac{3}{4}\xi_1\xi_3$
58				$-\frac{15}{8}\xi_1^3\xi_2^2+\frac{45}{8}\xi_1^3\xi_2^2\xi_3^2+\frac{5}{8}\xi_1^3-\frac{15}{8}\xi_1^3\xi_3^2$ $+\frac{9}{8}\xi_1\xi_2^2-\frac{27}{8}\xi_1\xi_2^2\xi_3^2-\frac{3}{8}\xi_1+\frac{9}{8}\xi_1\xi_3^2$
59				$\frac{75}{8}\xi_1^3\xi_2^2\xi_3^3-\frac{45}{8}\xi_1^3\xi_3\xi_2^2-\frac{25}{8}\xi_1^3\xi_3^3+\frac{15}{8}\xi_1^3\xi_3$ $-\frac{45}{8}\xi_1\xi_2^2\xi_3^3+\frac{27}{8}\xi_1\xi_3\xi_2^2+\frac{15}{8}\xi_1\xi_3^3-\frac{9}{8}\xi_1\xi_3$
60				$\frac{25}{4}\xi_1^3\xi_2^3-\frac{15}{4}\xi_1^3\xi_2-\frac{15}{4}\xi_1\xi_2^3+\frac{9}{4}\xi_1\xi_2$
61				$\frac{25}{4}\xi_1^3\xi_3\xi_2^3-\frac{15}{4}\xi_1^3\xi_3\xi_2-\frac{15}{4}\xi_1\xi_3\xi_2^3+\frac{9}{4}\xi_1\xi_3\xi_2$
62				$-\frac{25}{8}\xi_1^3\xi_2^3+\frac{75}{8}\xi_1^3\xi_2^3\xi_3^2+\frac{15}{8}\xi_1^3\xi_2-\frac{45}{8}\xi_1^3\xi_2\xi_3^2$ $+\frac{15}{8}\xi_1\xi_2^3-\frac{45}{8}\xi_1\xi_2^3\xi_3^2-\frac{9}{8}\xi_1\xi_2+\frac{27}{8}\xi_1\xi_2\xi_3^2$
63				$\frac{125}{8}\xi_1^3\xi_2^3\xi_3^3-\frac{75}{8}\xi_1^3\xi_2^3\xi_3-\frac{75}{8}\xi_1^3\xi_2\xi_3^3+\frac{45}{8}\xi_1^3\xi_2\xi_3$ $-\frac{75}{8}\xi_1\xi_2^3\xi_3^3+\frac{45}{8}\xi_1\xi_2^3\xi_3+\frac{45}{8}\xi_1\xi_2\xi_3^3-\frac{27}{8}\xi_1\xi_2\xi_3$

APPENDIX B

QUADRATURE POINTS

The Gauss quadrature rules are applied to the implementation. The space of polynomial functions of degree at most k is:

$$P^k = \{p(x) \,|\, p(x) \text{ is polynomial of degree} \leq k\} \tag{B.1}$$

B.1 One-dimensional case

For 1D case, an approximation to the integral of polynomial function $p(x)$ is obtained by sampling function at the space abscissas x_i with the corresponding weights ω_i:

$$\int_{-1}^{1} p(x)\mathrm{d}x = \sum_{i=1}^{n} \omega_i p(x_i) \tag{B.2}$$

where n is the number of sampling points. Three choices of quadrature rules can be found from mathematic formulae: Gauss-Legendre quadrature, Gauss-Lobatto quadrature, and Gauss-Radau quadrature. For the current implementation, at most fourth order of accuracy is considered, and the sampling points and weights are given in the following tables, Table B.1 to Table B.5 (note: in order to minimize the effect of machine errors, very high precision numbers are shown here).

B.2 Two-dimensional case

For 2D case, an approximation to the integral of polynomial function $p(x, y)$ is obtained by sampling function at the space abscissas x_i, y_i with the corresponding weights ω_i:

$$\int_{-1}^{1}\int_{-1}^{1} p(x, y)\mathrm{d}x\mathrm{d}y = \sum_{i=1}^{n} \omega_i p(x_i, y_i) \tag{B.3}$$

For the case of reference quadrilateral domain, the numerical integration can be computed as the tensor product of the two 1D integrations, for example, the integral is computed

Table B.1

Gauss-Legendre quadrature points

Order	Integration points x_i
1	x_1= 0.0000000000000000000000000000000000000000000000
2	x_1= -0.57735026918962576450914878050195745564 76 x_2= 0.57735026918962576450914878050195745564 76
3	x_1= -0.77459666924148337703585307995647992216 66 x_2= 0.0000000000000000000000000000000000000000000000 x_3= 0.77459666924148337703585307995647992216 66
4	x_1= -0.86113631159405257522394648889280950509 57 x_2= -0.33998104358485626480266575910324468720 06 x_3= 0.33998104358485626480266575910324468720 06 x_4= 0.86113631159405257522394648889280950509 57
5	x_1= -0.90617984593866399279762687829939296512 57 x_2= -0.53846931010568309103631442070020880496 73 x_3= 0.0000000000000000000000000000000000000000000000 x_4= 0.53846931010568309103631442070020880496 73 x_5= 0.90617984593866399279762687829939296512 57

Table B.2

Gauss-Legendre quadrature weights

Order	Integration weights ω_i
1	ω_1= 2.0000000000000000000000000000000000000000000000
2	ω_1= 1.0000000000000000000000000000000000000000000000 ω_2= 1.0000000000000000000000000000000000000000000000
3	ω_1= 0.5555555555555555555555555555555555555555555556 ω_2= 0.8888888888888888888888888888888888888888888889 ω_3= 0.5555555555555555555555555555555555555555555556
4	ω_1= 0.3478548451374538573730639492219994072353 ω_2= 0.6521451548625461426269360507780005927647 ω_3= 0.6521451548625461426269360507780005927647 ω_4= 0.3478548451374538573730639492219994072353
5	ω_1= 0.2369268850561890875142640407199173626433 ω_2= 0.4786286704993664680412915148356381929123 ω_3= 0.5688888888888888888888888888888888888888888889 ω_4= 0.4786286704993664680412915148356381929123 ω_5= 0.2369268850561890875142640407199173626433

Table B.3

Gauss-Radau quadrature points

Order	Integration points x_i
1	x_1= -1.0
	x_2= 1/3
2	x_1= -1.0000000000000000000000000000000000
	x_2= -0.28989794855663561963945681494118
	x_3= 0.68989794855663561963945681494118
3	x_1= -1.0000000000000000000000000000000000
	x_2= -0.57531892352169411205048377975200
	x_3= 0.18106627111853057827014749586234
	x_4= 0.82282408097459210520890771246109
4	x_1= -1.0000000000000000000000000000000000
	x_2= -0.72048027131243889569582583775024
	x_3= -0.16718086473783364011339533732583
	x_4= 0.44631397272375234463990800462875
	x_5= 0.88579160777096463561375761489177
5	x_1= -1.0000000000000000000000000000000000
	x_2= -0.80292982840234714775300220422449
	x_3= -0.39092854670727218902922964744233
	x_4= 0.12405037950522771198997495998954
	x_5= 0.60397316425278365492841572640941
	x_6= 0.92038028589706251531838661981333

Table B.4

Gauss-Radau quadrature weights

Order	Integration weights ω_i
1	ω_1= 0.5
	ω_2= 1.5
2	ω_1= 2/9
	ω_2= 1.02497165237684322767762689303920
	ω_3= 0.75280612540093455010015088473858
3	ω_1= 2/16
	ω_2= 0.65768863996011948788857844214558
	ω_3= 0.77638693768634376156046461378002
	ω_4= 0.44092442235353675055095694407455
4	ω_1= 2/25
	ω_2= 0.44620780216714148880512043645701
	ω_3= 0.62365304595148250816370982315323
	ω_4= 0.56271203029892412038434530068129
	ω_5= 0.28742712158245188264682443970824
5	ω_1= 2/36
	ω_2= 0.31964075322051096654577998379628
	ω_3= 0.48538718846896991615982791558685
	ω_4= 0.52092678318957498257022940656957
	ω_5= 0.41690133431190773895940638274197
	ω_6= 0.20158838525348084020920075574912

Table B.5

Gauss-Lobatto quadrature points and weights

Order	Integration points x_i	Integration weights ω_i
1	x_1= -1.0	ω_1= 1.0
	x_2= 1.0	ω_2= 1.0
2	x_1= -1.0	ω_1= 1/3
	x_2= 0.0	ω_2= 4/3
	x_3= 1.0	ω_3= 1/3
3	x_1= -1.0	ω_1= 1/6
	x_2= -$\sqrt{5}/5$	ω_2= 5/6
	x_3= $\sqrt{5}/5$	ω_3= 5/6
	x_4= 1.0	ω_4= 1/6
4	x_1= -1.0	ω_1= 0.1
	x_2= -$\sqrt{21}/7$	ω_2= 49/90
	x_3= 0.0	ω_3= 32/45
	x_4= $\sqrt{21}/7$	ω_4= 49/90
	x_5= 1.0	ω_5= 0.1
5	x_1= -1.0	ω_1= 1/15
	x_2= -$\sqrt{7+2\sqrt{7}}/\sqrt{21}$	ω_2= $(14-\sqrt{7})/30$
	x_3= -$\sqrt{7-2\sqrt{7}}/\sqrt{21}$	ω_3= $(14+\sqrt{7})/30$
	x_4= $\sqrt{7-2\sqrt{7}}/\sqrt{21}$	ω_4= $(14+\sqrt{7})/30$
	x_5= $\sqrt{7+2\sqrt{7}}/\sqrt{21}$	ω_5= $(14-\sqrt{7})/30$
	x_6= 1.0	ω_6= 1/15

from two independent Gauss-Legendre quadrature integrations as shown on the left of Figure B.1.

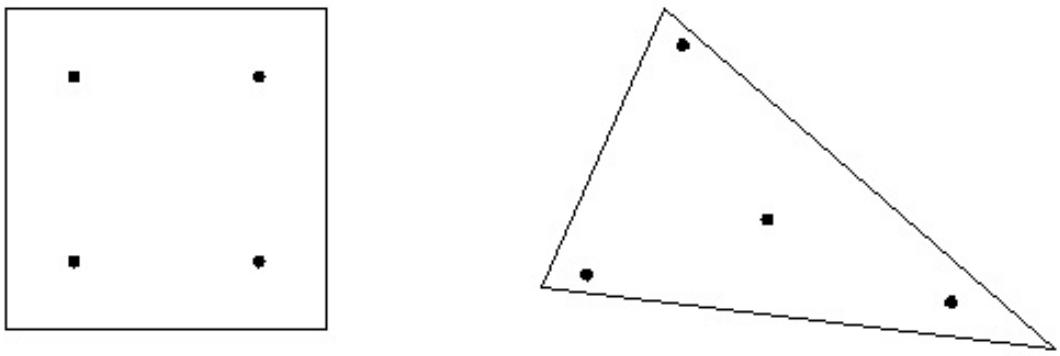

Figure B.1

Second order quadrature points in quadrilateral domain and triangular domain

For the case of reference triangular domain, the numerical integration can be computed on quadrature points as shown on the right of Figure B.1. For the current implementation, at most fourth order of accuracy is used, then the data of sampling points and weights are given in Table B.6 and Table B.7.

B.3 Three-dimensional case

For 3D case, an approximation to the integral of polynomial function $p(x, y, z)$ is obtained by sampling function at the space abscissas x_i, y_i, z_i with the corresponding weights ω_i:

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} p(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z = \sum_{i=1}^{n} \omega_i p(x_i, y_i, z_i) \tag{B.4}$$

Table B.6

Quadrature points in triangular domain

Order	First vertex T_1	Second vertex T_2	Third vertex T_3
1	x_1= 1/3	x_1= 1/3	x_1= 1/3
2	x_1= 1/3	x_1= 1/3	x_1= 1/3
	x_2= 0.6	x_2= 0.2	x_2= 0.2
	x_3= 0.2	x_3= 0.6	x_3= 0.2
	x_4= 0.2	x_4= 0.2	x_4= 0.6
3	x_1= 1/3	x_1= 1/3	x_1= 1/3
	x_2= α_1	x_2= β_1	x_2= β_1
	x_3= β_1	x_3= α_1	x_3= β_1
	x_4= β_1	x_4= β_1	x_4= α_1
	x_5= α_2	x_5= β_2	x_5= β_2
	x_6= β_2	x_6= α_2	x_6= β_2
	x_7= β_2	x_7= β_2	x_7= α_2
4	x_1= ξ_1	x_1= η_1	x_1= $1-\xi_1-\eta_1$
	x_2= η_1	x_2= $1-\xi_1-\eta_1$	x_2= ξ_1
	x_3= $1-\xi_1-\eta_1$	x_3= ξ_1	x_3= η_1
	x_4= ξ_2	x_4= η_2	x_4= $1-\xi_2-\eta_2$
	x_5= η_2	x_5= $1-\xi_2-\eta_2$	x_5= ξ_2
	x_6= $1-\xi_2-\eta_2$	x_6= ξ_2	x_6= η_2
	x_7= ξ_3	x_7= η_3	x_7= $1-\xi_3-\eta_3$
	x_8= η_3	x_8= $1-\xi_3-\eta_3$	x_8= ξ_3
	x_9= $1-\xi_3-\eta_3$	x_9= ξ_3	x_9= η_3
	x_{10}= ξ_4	x_{10}= η_4	x_{10}= $1-\xi_4-\eta_4$
	x_{11}= η_4	x_{11}= $1-\xi_4-\eta_4$	x_{11}= ξ_4
	x_{12}= $1-\xi_4-\eta_4$	x_{12}= ξ_4	x_{12}= η_4
	α_1 = 0.059715871789769820459117580973106		
	β_1 = 0.470142064105115089770441209513447		
	α_2 = 0.797426985353087322398025276169754		
	β_2 = 0.101286507323456338800987361915123		
	ξ_1 = 0.062382265094402118173683000996349 9		
	η_1 = 0.067517867073916085442557131050868 5		
	ξ_2 = 0.055225456656926611737479190275644 9		
	η_2 = 0.321502493851981822666307849199202		
	ξ_3 = 0.034324302945097146469630642483937 6		
	η_3 = 0.660949196186735657611980310197799		
	ξ_4 = 0.515842334353591779257463386826430		
	η_4 = 0.277716166976391782569581871393723		

Table B.7

Quadrature weights in triangular domain

Order	Integration weights ω_i
1	ω_1= 1.0
2	ω_1= -27/48 ω_2= 25/48 ω_3= 25/48 ω_4= 25/48
3	ω_1= 0.22500000000000000000000000000000 ω_2= 0.13239415278850618073764938783315 18 ω_3= 0.13239415278850618073764938783315 18 ω_4= 0.13239415278850618073764938783315 18 ω_5= 0.12593918054482715259568394550018 12 ω_6= 0.12593918054482715259568394550018 12 ω_7= 0.12593918054482715259568394550018 12
4	ω_1= 0.026517028157436251428754180460739 1$*$2.0 ω_2= 0.026517028157436251428754180460739 1$*$2.0 ω_3= 0.026517028157436251428754180460739 1$*$2.0 ω_4= 0.043881408714446055036769903139287 5$*$2.0 ω_5= 0.043881408714446055036769903139287 5$*$2.0 ω_6= 0.043881408714446055036769903139287 5$*$2.0 ω_7= 0.028775042784981585738445496900218 5$*$2.0 ω_8= 0.028775042784981585738445496900218 5$*$2.0 ω_9= 0.028775042784981585738445496900218 5$*$2.0 ω_{10}= 0.067493187009802774462697086166421 4$*$2.0 ω_{11}= 0.067493187009802774462697086166421 4$*$2.0 ω_{12}= 0.067493187009802774462697086166421 4$*$2.0

The integration strategies for different type of hybrid elements are not the same. For the case of reference hexahedral domain, the numerical integration can be computed as the tensor product of the three 1D integrations. For example, the integral is computed from three independent Gauss-Legendre quadrature integrations. For the case of reference prism domain, the numerical integration can be computed with triangle integration on collapsed direction in triangular face and Gauss-Legendre integration on the third direction. For the case of reference pyramid domain, the numerical integration can be computed with Gauss-Radau integration on collapsed direction and two independent Gauss-Legendre integrations on other two directions in quadrilateral face. For the case of reference tetrahedral domain, the numerical integration can be performed on quadrature points, as shown on Figure B.2. For the current implementation, at most fourth order of accuracy is used, then the data of sampling points and weights are given in Table B.8 and Table B.9.

Table B.8

Quadrature points in tetrahedral domain

Order	First vertex T_1	Second vertex T_2	Third vertex T_3	Fourth vertex T_4
1	x_1= 1/4	x_1= 1/4	x_1= 1/4	x_1= 1/4
2	x_1= 1/4	x_1= 1/4	x_1= 1/4	x_1= 1/4
	x_2= 1/2	x_2= 1/6	x_2= 1/6	x_2= 1/6
	x_3= 1/6	x_3= 1/2	x_3= 1/6	x_3= 1/6
	x_4= 1/6	x_4= 1/6	x_4= 1/2	x_4= 1/6
	x_5= 1/6	x_5= 1/6	x_5= 1/6	x_5= 1/2
3	x_1= α_{31}	x_1= α_{31}	x_1= α_{31}	x_1= β_{31}
	x_2= β_{31}	x_2= α_{31}	x_2= α_{31}	x_2= α_{31}
	x_3= α_{31}	x_3= β_{31}	x_3= α_{31}	x_3= α_{31}
	x_4= α_{31}	x_4= α_{31}	x_4= β_{31}	x_4= α_{31}
	x_5= α_{32}	x_5= α_{32}	x_5= α_{32}	x_5= β_{32}
	x_6= β_{32}	x_6= α_{32}	x_6= α_{32}	x_6= α_{32}
	x_7= α_{32}	x_7= β_{32}	x_7= α_{32}	x_7= α_{32}
	x_8= α_{32}	x_8= α_{32}	x_8= β_{32}	x_8= α_{32}
	x_9= α_{33}	x_9= α_{33}	x_9= α_{33}	x_9= β_{33}
	x_{10}= β_{33}	x_{10}= α_{33}	x_{10}= α_{33}	x_{10}= α_{33}
	x_{11}= α_{33}	x_{11}= β_{33}	x_{11}= α_{33}	x_{11}= α_{33}
	x_{12}= α_{33}	x_{12}= α_{33}	x_{12}= β_{33}	x_{12}= α_{33}
	x_{13}= α_{34}	x_{13}= α_{34}	x_{13}= β_{34}	x_{13}= γ_{34}
	x_{14}= α_{34}	x_{14}= α_{34}	x_{14}= γ_{34}	x_{14}= β_{34}
	x_{15}= α_{34}	x_{15}= β_{34}	x_{15}= α_{34}	x_{15}= γ_{34}
	x_{16}= α_{34}	x_{16}= β_{34}	x_{16}= γ_{34}	x_{16}= α_{34}
	x_{17}= α_{34}	x_{17}= γ_{34}	x_{17}= α_{34}	x_{17}= β_{34}
	x_{18}= α_{34}	x_{18}= γ_{34}	x_{18}= β_{34}	x_{18}= α_{34}
	x_{19}= β_{34}	x_{19}= α_{34}	x_{19}= α_{34}	x_{19}= γ_{34}
	x_{20}= β_{34}	x_{20}= α_{34}	x_{20}= γ_{34}	x_{20}= α_{34}
	x_{21}= β_{34}	x_{21}= γ_{34}	x_{21}= α_{34}	x_{21}= α_{34}
	x_{22}= γ_{34}	x_{22}= α_{34}	x_{22}= α_{34}	x_{22}= β_{34}
	x_{23}= γ_{34}	x_{23}= α_{34}	x_{23}= β_{34}	x_{23}= α_{34}
	x_{24}= γ_{34}	x_{24}= β_{34}	x_{24}= α_{34}	x_{24}= α_{34}

Table B.8

(continued)

Order	First vertex T_1	Second vertex T_2	Third vertex T_3	Fourth vertex T_4
4	x_1= α_{41}	x_1= α_{41}	x_1= α_{41}	x_1= α_{41}
	x_2= α_{42}	x_2= α_{42}	x_2= α_{42}	x_2= β_{42}
	x_3= β_{42}	x_3= α_{42}	x_3= α_{42}	x_3= α_{42}
	x_4= α_{42}	x_4= β_{42}	x_4= α_{42}	x_4= α_{42}
	x_5= α_{42}	x_5= α_{42}	x_5= β_{42}	x_5= α_{42}
	x_6= α_{43}	x_6= α_{43}	x_6= α_{43}	x_6= β_{43}
	x_7= β_{43}	x_7= α_{43}	x_7= α_{43}	x_7= α_{43}
	x_8= α_{43}	x_8= β_{43}	x_8= α_{43}	x_8= α_{43}
	x_9= α_{43}	x_9= α_{43}	x_9= β_{43}	x_9= α_{43}
	x_{10}= α_{44}	x_{10}= α_{44}	x_{10}= α_{44}	x_{10}= β_{44}
	x_{11}= β_{44}	x_{11}= α_{44}	x_{11}= α_{44}	x_{11}= α_{44}
	x_{12}= α_{44}	x_{12}= β_{44}	x_{12}= α_{44}	x_{12}= α_{44}
	x_{13}= α_{44}	x_{13}= α_{44}	x_{13}= β_{44}	x_{13}= α_{44}
	x_{14}= α_{45}	x_{14}= α_{45}	x_{14}= β_{45}	x_{14}= β_{45}
	x_{15}= α_{45}	x_{15}= β_{45}	x_{15}= α_{45}	x_{15}= β_{45}
	x_{16}= α_{45}	x_{16}= β_{45}	x_{16}= β_{45}	x_{16}= α_{45}
	x_{17}= β_{45}	x_{17}= α_{45}	x_{17}= α_{45}	x_{17}= β_{45}
	x_{18}= β_{45}	x_{18}= α_{45}	x_{18}= β_{45}	x_{18}= α_{45}
	x_{19}= β_{45}	x_{19}= β_{45}	x_{19}= α_{45}	x_{19}= α_{45}
	x_{20}= α_{46}	x_{20}= α_{46}	x_{20}= β_{46}	x_{20}= γ_{46}
	x_{21}= α_{46}	x_{21}= α_{46}	x_{21}= γ_{46}	x_{21}= β_{46}
	x_{22}= α_{46}	x_{22}= β_{46}	x_{22}= α_{46}	x_{22}= γ_{46}
	x_{23}= α_{46}	x_{23}= β_{46}	x_{23}= γ_{46}	x_{23}= α_{46}
	x_{24}= α_{46}	x_{24}= γ_{46}	x_{24}= α_{46}	x_{24}= β_{46}
	x_{25}= α_{46}	x_{25}= γ_{46}	x_{25}= β_{46}	x_{25}= α_{46}
	x_{26}= β_{46}	x_{26}= α_{46}	x_{26}= α_{46}	x_{26}= γ_{46}
	x_{27}= β_{46}	x_{27}= α_{46}	x_{27}= γ_{46}	x_{27}= α_{46}
	x_{28}= β_{46}	x_{28}= γ_{46}	x_{28}= α_{46}	x_{28}= α_{46}
	x_{29}= γ_{46}	x_{29}= α_{46}	x_{29}= α_{46}	x_{29}= β_{46}
	x_{30}= γ_{46}	x_{30}= α_{46}	x_{30}= β_{46}	x_{30}= α_{46}
	x_{31}= γ_{46}	x_{31}= β_{46}	x_{31}= α_{46}	x_{31}= α_{46}
	x_{32}= α_{47}	x_{32}= α_{47}	x_{32}= β_{47}	x_{32}= γ_{47}
	x_{33}= α_{47}	x_{33}= α_{47}	x_{33}= γ_{47}	x_{33}= β_{47}
	x_{34}= α_{47}	x_{34}= β_{47}	x_{34}= α_{47}	x_{34}= γ_{47}
	x_{35}= α_{47}	x_{35}= β_{47}	x_{35}= γ_{47}	x_{35}= α_{47}

Table B.8

(continued)

Order	First vertex T_1	Second vertex T_2	Third vertex T_3	Fourth vertex T_4
	x_{36}= α_{47}	x_{36}= γ_{47}	x_{36}= α_{47}	x_{36}= β_{47}
	x_{37}= α_{47}	x_{37}= γ_{47}	x_{37}= β_{47}	x_{37}= α_{47}
	x_{38}= β_{47}	x_{38}= α_{47}	x_{38}= α_{47}	x_{38}= γ_{47}
	x_{39}= β_{47}	x_{39}= α_{47}	x_{39}= γ_{47}	x_{39}= α_{47}
	x_{40}= β_{47}	x_{40}= γ_{47}	x_{40}= α_{47}	x_{40}= α_{47}
	x_{41}= γ_{47}	x_{41}= α_{47}	x_{41}= α_{47}	x_{41}= β_{47}
	x_{42}= γ_{47}	x_{42}= α_{47}	x_{42}= β_{47}	x_{42}= α_{47}
	x_{43}= γ_{47}	x_{43}= β_{47}	x_{43}= α_{47}	x_{43}= α_{47}
	α_{31} = 0.21460287125915202928839219386284			
	β_{31} = 1.0 - 3.0α_{31}			
	α_{32} = 0.04067395853461135311557944895641 00			
	β_{32} = 1.0 - 3.0α_{32}			
	α_{33} = 0.32233789014227551034399447076249 2			
	β_{33} = 1.0 - 3.0α_{33}			
	α_{34} = 0.06366100187501752529923552760572 69			
	β_{34} = 0.26967233145831580803409780572760 6			
	γ_{34} = 1.0 - 2.0α_{34} - β_{34}			
	α_{41} = 1/4			
	α_{42} = 0.20682993161067320408398090002496 1			
	β_{42} = 1.0 - 3.0α_{42}			
	α_{43} = 0.08210358831054672309060580787142 15			
	β_{43} = 1.0 - 3.0α_{43}			
	α_{44} = 0.00578195050519799725317663886414 270			
	β_{44} = 1.0 - 3.0α_{44}			
	α_{45} = 0.05053274001889422442562452855790 71			
	β_{45} = 0.44946725998110577557437547144209 2			
	α_{46} = 0.22906653611681113960040885455475 3			
	β_{46} = 0.03563958278853404371691739695061 14			
	γ_{46} = 1.0 - 2.0α_{46} - β_{46}			
	α_{47} = 0.03660774955319742367877385463271 04			
	β_{47} = 0.19048604193463345569943328531509 9			
	γ_{47} = 1.0 - 2.0α_{47} - β_{47}			

Table B.9

Quadrature weights in tetrahedral domain

Order	Integration weights ω_i
1	ω_1= 4/3
2	ω_1= -4/5$*$(4/3) ω_2= 9/20$*$(4/3) ω_3= 9/20$*$(4/3) ω_4= 9/20$*$(4/3) ω_5= 9/20$*$(4/3)
3	ω_1= 0.006653791709694582016615104592913 32$*$8.0 ω_2= 0.006653791709694582016615104592913 32$*$8.0 ω_3= 0.006653791709694582016615104592913 32$*$8.0 ω_4= 0.006653791709694582016615104592913 32$*$8.0 ω_5= 0.001679535175886773824668872907656 14$*$8.0 ω_6= 0.001679535175886773824668872907656 14$*$8.0 ω_7= 0.001679535175886773824668872907656 14$*$8.0 ω_8= 0.001679535175886773824668872907656 14$*$8.0 ω_9= 0.009226196923942453682525546308954 33$*$8.0 ω_{10}= 0.009226196923942453682525546308954 33$*$8.0 ω_{11}= 0.009226196923942453682525546308954 33$*$8.0 ω_{12}= 0.009226196923942453682525546308954 33$*$8.0 ω_{13}= 0.008035714285714285714285714285714 28$*$8.0 ω_{14}= 0.008035714285714285714285714285714 28$*$8.0 ω_{15}= 0.008035714285714285714285714285714 28$*$8.0 ω_{16}= 0.008035714285714285714285714285714 28$*$8.0 ω_{17}= 0.008035714285714285714285714285714 28$*$8.0 ω_{18}= 0.008035714285714285714285714285714 28$*$8.0 ω_{19}= 0.008035714285714285714285714285714 28$*$8.0 ω_{20}= 0.008035714285714285714285714285714 28$*$8.0 ω_{21}= 0.008035714285714285714285714285714 28$*$8.0 ω_{22}= 0.008035714285714285714285714285714 28$*$8.0 ω_{23}= 0.008035714285714285714285714285714 28$*$8.0 ω_{24}= 0.008035714285714285714285714285714 28$*$8.0

Table B.9

(continued)

Order	Integration weights ω_i
4	ω_1= -0.020500188658639915840586517764294 1$*$8.0
	ω_2= 0.014250305822866901248439741535870 4$*$8.0
	ω_3= 0.014250305822866901248439741535870 4$*$8.0
	ω_4= 0.014250305822866901248439741535870 4$*$8.0
	ω_5= 0.014250305822866901248439741535870 4$*$8.0
	ω_6= 0.001967033313133900987562803424454 66$*$8.0
	ω_7= 0.001967033313133900987562803424454 66$*$8.0
	ω_8= 0.001967033313133900987562803424454 66$*$8.0
	ω_9= 0.001967033313133900987562803424454 66$*$8.0
	ω_{10}= 0.000169834109092887379837744566704016$*$8.0
	ω_{11}= 0.000169834109092887379837744566704016$*$8.0
	ω_{12}= 0.000169834109092887379837744566704016$*$8.0
	ω_{13}= 0.000169834109092887379837744566704016$*$8.0
	ω_{14}= 0.004579683824467281800743514462976 $*$8.0
	ω_{15}= 0.004579683824467281800743514462976 $*$8.0
	ω_{16}= 0.004579683824467281800743514462976 $*$8.0
	ω_{17}= 0.004579683824467281800743514462976 $*$8.0
	ω_{18}= 0.004579683824467281800743514462976 $*$8.0
	ω_{19}= 0.004579683824467281800743514462976 $*$8.0
	ω_{20}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{21}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{22}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{23}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{24}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{25}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{26}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{27}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{28}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{29}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{30}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{31}= 0.005704485808681918506802558627830 40$*$8.0
	ω_{32}= 0.002140519141162092596483353000920 23$*$8.0
	ω_{33}= 0.002140519141162092596483353000920 23$*$8.0
	ω_{34}= 0.002140519141162092596483353000920 23$*$8.0
	ω_{35}= 0.002140519141162092596483353000920 23$*$8.0
	ω_{36}= 0.002140519141162092596483353000920 23$*$8.0

Table B.9

(continued)

Order	Integration weights ω_i
4	ω_{37}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{38}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{39}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{40}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{41}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{42}= 0.0021405191411620925964833530009 2023∗8.0
	ω_{43}= 0.0021405191411620925964833530009 2023∗8.0

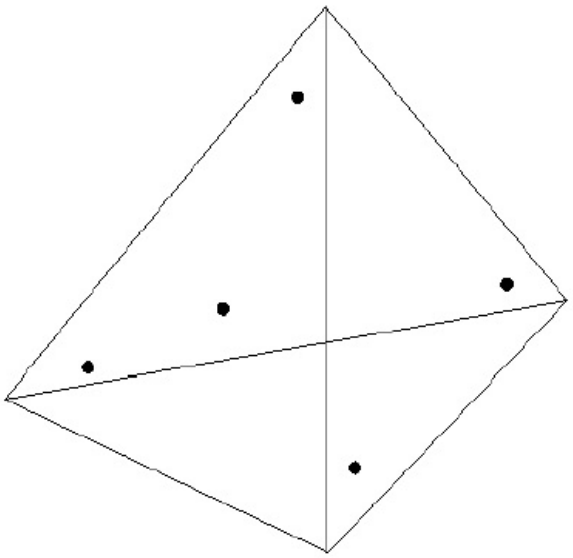

Figure B.2

Second order quadrature points in tetrahedral domain

APPENDIX C

JACOBIAN MATRIX

The computation of the Jacobian matrix for the Euler equations in Equation (4.5) is discussed now. The computation of the Jacobian matrix for the RANS equations in Equation (4.65) can be taken in a similar manner depending on the diffusive flux approximation method chosen (BR2 or LDG method). The difference between these two matrices is in computing the derivatives of diffusive flux and source term, which can be obtained analytically using mathematic software (Maple) or numerically from finite difference approximation. Since the derivation is very long and complicated, it is not shown here. Whenever needed, they can be derived in a similar procedure as shown next for the Euler equations.

The residual vector from the Euler governing equations in Equation (4.5) is:

$$\mathbf{R}(U) = (A) - (B) \tag{C.1}$$

based on this general form, the full Jacobian matrix $\partial\mathbf{R}(U)/\partial U$ is computed for implicit time integration. Taking the derivative of the above equations, we can explicitly write out the Jacobian matrix as:

$$\frac{\partial\mathbf{R}(U)}{\partial U} = \frac{\partial(A)}{\partial U} - \frac{\partial(B)}{\partial U} \tag{C.2}$$

C.1 Compute $\partial(A)/\partial U$

From Equation (4.6) we know the cell convective flux contribution is:

$$(A) = \int_{\Omega_h} \mathbf{F}(U_h) \cdot \nabla\psi_h \mathrm{d}\Omega \tag{C.3}$$

hence the derivative is:

$$\frac{\partial(A)}{\partial U_h} = \int_{\Omega_h} \frac{\partial\mathbf{F}(U_h)}{\partial U_h} \cdot \nabla\psi_h \mathrm{d}\Omega \tag{C.4}$$

where the convective flux is given explicitly as a function of conservative variables, so we can directly calculate this convective flux Jacobian matrix and integrate over domain to get the $\partial(A)/\partial U$ term.

C.2 Compute $\partial(B)/\partial U$

From Equation (4.7) we find the face convective flux contribution is:

$$(B) = \int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)(\psi_h^- - \psi_h^+)\mathrm{d}\sigma \tag{C.5}$$

hence the derivatives are computed on faces with respect to solution vector from left and right side of this face are:

$$\begin{aligned}
\frac{\partial(B)}{\partial U_h} &= \frac{\partial}{\partial U_h}\Big[\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)(\psi_h^- - \psi_h^+)\mathrm{d}\sigma\Big] \\
&= \frac{\partial}{\partial U_h^-}\Big[\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)\psi_h^-\mathrm{d}\sigma\Big] - \frac{\partial}{\partial U_h^+}\Big[\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)\psi_h^+\mathrm{d}\sigma\Big] \\
&+ \frac{\partial}{\partial U_h^+}\Big[\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)\psi_h^-\mathrm{d}\sigma\Big] - \frac{\partial}{\partial U_h^-}\Big[\int_{\Sigma_h} \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)\psi_h^+\mathrm{d}\sigma\Big] \\
&= \int_{\Sigma_h} \frac{\partial \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)}{\partial U_h^-}\psi_h^-\mathrm{d}\sigma - \int_{\Sigma_h} \frac{\partial \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)}{\partial U_h^+}\psi_h^+\mathrm{d}\sigma \\
&+ \int_{\Sigma_h} \frac{\partial \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)}{\partial U_h^+}\psi_h^-\mathrm{d}\sigma - \int_{\Sigma_h} \frac{\partial \widehat{\mathbf{F}}(U_h^-, U_h^+; \mathbf{n}^-)}{\partial U_h^-}\psi_h^+\mathrm{d}\sigma
\end{aligned} \tag{C.6}$$

The first and second terms in above equations are the contributions to the diagonal parts of the convective flux Jacobian matrix, whereas the third and fourth terms are off-diagonal parts of the matrix. Since the face convective flux is given explicitly as a function of conservative variables by means of the convective flux approximation method (for example the Roe scheme), we can directly compute the derivative of this convective flux approximation with respect to the solution vector (U_h^- or U_h^+) and integrate over the domain to get the $\partial(B)/\partial U$ term.