NUMERICAL SOLUTION OF MOMENT EQUATIONS USING THE DISCONTINUOUS-GALERKIN HANCOCK METHOD

A Thesis Submitted to

the Faculty of Graduate and Postdoctoral Studies

by

Seyedalireza Miri

In Partial Fulfillment of the Requirements for the Degree of

MASTERS OF APPLIED SCIENCE

in Mechanical Engineering

Department of Mechanical Engineering

University of Ottawa

Ottawa, Canada

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. First, I would like to express my sincere gratitude to my thesis supervisors, Professor James McDonald for continuous support of my Master studies and research. His deep insights and great ideas helped me at various stages of my research. Without his input, this thesis would not have been possible. I am also extremely grateful to University of Ottawa department of mechanical engineering for all the great support and help. I would also like to thank the members of my thesis examiners, Professor Catherine Mavriplis and Professor Henry M.J. Saari for their time, helpful advice and suggestions. I would like to thank my colleagues and friends in UOttawa, whose continuous help and support I have been receiving during my journey in finishing this Masters program. I will forever be thankful to all the people who made UOttawa a great place to work. There is not enough space to mention all of you. I would also like to thank the great UOttawa staff who has always been helpful. To my friends outside of UOttawa, thank you for your genuine friendship, your support, kindness and love. Finally, I would like to say a heartfelt thank you to my parents and my sister and also my aunt for their supports and help. Thank you for your unconditional love and care, for always believing in me and encouraging me to follow my dreams. I would not have made it this far without you. Thank you.

Abstract

Moment methods from the kinetic theory of gases exist as an alternative to the Navier-Stokes model. Models in this family are described by first-order hyperbolic PDEs with local relaxation. They provide a natural treatment for non-equilibrium effects and expand the regime for which the model is physically applicable past the Navier-Stokes level (when the continuum assumption breaks down).

Discontinuous-Galerkin (DG) methods are very well suited for distributed parallel solution of first-order PDEs. This is because the optimal locality of the method minimizes needed communication between computational processes. One highly efficient, coupled space-time DG method that achieves third-order accuracy in both space and time while using only linear elements is the discontinuous-Galerkin Hancock (DGH) scheme, which was specifically designed for the efficient solution of PDEs resulting from moment closures. Third-order accuracy is obtained through the use of a technique originally proposed by Hancock. The combination of moment methods with the DGH discretization leads to a very efficient numerical treatment for viscous compressible gas flows that is accurate both in and out of local thermodynamic equilibrium.

This thesis describe the first-ever implementation of this scheme for the solution of moment equations on large-scale distributed-memory computers. This implementation uses solution-directed automatic mesh refinement to increase accuracy while

reducing cost. A linear hyperbolic-relaxation equation is used to verify the order of accuracy of the scheme. Next a supersonic compressible Euler case is used to demonstrate the mesh refinement as well as the scheme's ability to capture sharp discontinuities. Third, a moment-closure is then used to compute a viscous mixing layer. This serves to demonstrate the ability of the first-order PDEs and the DG scheme to efficiently compute viscous solutions. A moment-closure is used to compute the solution for Stokes flow past a circular cylinder. This case reinforces the hyperbolic PDEs' ability to accurately predict viscous phenomena. As this case is very low speed, it also demonstrates the numerical technique's ability to accurately solve problems that are ill-conditioned due to the extremely low Mach number. Finally, the parallel efficiency of the scheme is evaluated on Canada's largest supercomputer.

It may be surprising to some that viscous flow behaviour can be accurately predicted by first-order PDEs. However, the applicability of hyperbolic moment methods to both continuum and non-equilibrium gas flows is now well established. Such a first-order treatment brings many physical and computational advantages to gas flow prediction.

Keywords: Moment Methods, Discontinuous Galerkin Scheme

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The compressible Navier-Stokes equations have been an immensely successful model for viscous gas-flow prediction. In fact, these equations can usually produce predictions that have such high accuracy that people can be forgiven for forgetting that they are only a model. However, in regimes where length scales of interest begin to approach the mean free path and the continuum assumption is violated, physically inaccurate results are obtained. The degree to which the continuum assumption is valid can be characterized by a non-dimensional number. This number is called the Knudsen number, which is defined as

$$\mathrm{Kn} = \frac{\lambda}{L}. \tag{1.1}$$

Here, λ is the mean free path of particles and L is a characteristic length. Basically, flows can be divided into four regimes, as listed in Table 1.1.

Table 1.1: Gas-Flow Regimes Definition

Kn< 0.01	continuum regime
0.01 <Kn< 0.1	slip-flow regime
0.1 <Kn< 10	transition regime
10 <Kn	free-molecular regime

A small Knudsen number means gas-particle collisions happen very frequently over the length of the problem. The result is that the individual contribution of each particle to the bulk is "averaged out" and the fluid can be treated as a continuum. These flows are said to exist in the continuum regime, where the Navier-Stokes equations provide an accurate model. As the Knudsen number increases, deviations from local equilibrium can be expected. Often these deviations first appear in the boundary layer. In this case, the Navier-Stokes equations can often still be used, but with modified "slip-flow" boundary conditions. Past this slip-flow regime, the Navier-Stokes equations cease to be a physically valid model.

In addition to the physical limitations of the Navier-Stokes equations, the partial differential equations (PDEs) that describe the model have mathematical properties that can make numerical solution inconvenient. For example, the second-order spacial derivatives present in the partial differential equations can cause solutions to be overly sensitive to grid quality [1].

When the Knudsen number becomes even larger, particle interactions occur more infrequently and derivations from local equilibrium become significant throughout the flow. When this happens, neither of the two classical models (Euler and Navier-Stokes) can make accurate predictions. Such non-equilibrium gas flows include micro-scale flows, highly rarefied flows, very high-speed flows and many other flows with extreme gradients in fluid properties.

For flows with a Knudsen number greater than approximately 10, the flow is said to be in the free-molecular regime, where, collisions are infrequent and can often be

ignored. The most widely used method in this non-equilibrium regime is the direct simulation Monte Carlo (DSMC) method established by Bird [2]. In this Model, one needs to make use of an extremely large number of representative particles to represent real molecules. The evolution of these representative particles is modelled in a statistical way that is expensive and has very slow error convergence. Although the Monte Carlo model is believed to be valid for all Knudsen numbers, its practical applicability is limited by its slow convergence rate, specially for flows with a high number of particles or for low-speed flows. The DSMC method is widely used for situations when the Knudsen numbers is above 10; the method tends to become prohibitively expensive for values much below that. Other methods that remain valid for all Knudsen numbers are based on direct numerical discretization of the high-dimensional Boltzmann equations [3]. These methods also suffer from computational cost caused by solving partial differential equations (PDEs) in a seven-dimensional phase space.

Moment methods from the kinetic theory of gases exist as another alternative to the Navier-Stokes model. Models in this family are described by first-order hyperbolic PDEs with local relaxation. Rather than including second-derivatives, the moment models include higher-order moments in the solution vector. This provides a natural treatment for non-equilibrium effects and expands the regime for which the model is physically applicable past the Navier-Stokes level. When combined with appropriate numerical techniques, solutions can be reliably and efficiently calculated on distributed-memory computers using meshes that may contain sharp irregularities.

Unfortunately, the PDEs describing the moment models bring their own numerical inconvenience: an extremely stiff local source term. Since the source term is parameterized by a relaxation time, which is of the order of the mean collision time, any standard explicit method has a severe time-step restriction with regard to both stability and accuracy, especially if one is only interested in evolution at the macroscopic temporal and spacial scale. There is a current lack of large-scale implementations of numerical methods for the efficient and accurate solution of these models.

1.2 Objective of the Current Study

Moment equations are in the form of hyperbolic-relaxation with stiff local source terms. The governing PDEs can be written as

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x_i}\boldsymbol{F}_i = \boldsymbol{S}\,. \tag{1.2}$$

Here, $\boldsymbol{U}$ is the solution vector containing an expanded number of velocity moments as compared to traditional methods, $\boldsymbol{F}_i$ is the flux dyad, and $\boldsymbol{S}$ is a local algebraic source term that can be quite stiff for dense-gas situations.

The main objective of this study is to implement, on a large scale, an existing numerical method specifically designed moment equations. This scheme has been previously developed mathematically, but has never been implemented for the solution of moment models. In a standard finite-volume method (FVM), solutions are defined as cell-averaged quantities over the computational domain, and the higher-order accuracy in space relies on local piecewise-polynomial reconstruction, which requires

extended stencils. Another class of methods, the Discontinuous-Galerkin (DG) methods, are very well suited for distributed parallel solution of first-order PDEs. This is because the optimal locality of the method minimizes needed communication between computational processes. The discontinues Galerkin (DG) method overcomes the issue of growing stencils by increasing the solution representation in each element, a solution in a cell is no longer piecewise constant, but a polynomial with each coefficient as an unknown. A highly efficient, coupled space-time DG method that achieves third-order accuracy in both space and time while using only linear elements is based on the upwind moment scheme, developed by Huynh [4], for hyperbolic conservation laws. The specific method used in this work has been developed by Suzuki and van Leer [5, 6] for hyperbolic-relaxation equations. Third-order accuracy is obtained through the use of a technique originally proposed by Hancock [7]. The two key characteristics of this method are:

- cell variables are first updated over a half time step without any interaction with neighbouring cells (Hancock's technique [7])
- the gradient of each flow variable evolves as described by an independent equation (DG representation).

The combination of moment methods with the discontinuous-Galerkin Hancock discretization leads to a very efficient numerical treatment for viscous compressible gas flows that is accurate both in and out of local thermodynamic equilibrium.

1.3 Scope of the Study

In Chapter 2 of this thesis, the background of moment methods is reviewed. After an introduction to the kinetic theory of gases and definition of the distribution function, the derivation of moment equations is shown. Moreover, the Boltzmann equations and the collision operator that is used in these equations are explained. Chapter 3 introduces the discontinuous-Galerkin-Hancock method. Following this, the exact form of the update equations for slopes and solution values within each element are presented. The advantages of this method are also explained. Chapter 4 explains the details of the current distributed adaptive-mesh-refinement implementation. The goal is to automatically refine and coarsen the mesh locally to increase accuracy and minimize cost while maintaining a good balance of work across CPUs. Furthermore, details of the techniques that are used to implement the DGH scheme in parallel are explained. Finally, Chapter 5 shows numerical results for several canonical problems. The accuracy and robustness of the scheme are demonstrated for several problems involving different models. The parallel efficiency and stability of the implementation are also assessed.

Chapter 2

Moment Method

2.1 Kinetic Theory of Gases

Traditional fluid dynamics is based on the assumption that fluids are well approximated by a continuum. Which means they have very small Knudsen number and stay very close to local equilibrium. Changes in fluid properties such as mass density, pressure, velocity, and temperature are well described by either the compressible Euler or Navier-Stokes equations. However, this assumption is not always valid. Increases in the distance between particles make the Knudsen number larger and push a flow out of the continuum regime. For example, this situation can happen for flow around spacecraft, because the rarefied gas in the very high atmosphere has a long mean free path.

Rather than starting from the assumption that the fluid can be treated as a continuum, one can begin from a molecular treatment of a gas. In the kinetic theory of gases, this is done probabilistically. In this theory, gas particles are not treated individually—statistics of the ensemble are treated instead.

2.2 The Distribution Function

In the kinetic theory, a distribution function, $\mathcal{F}$, is defined that gives the phase-space density of particles at a point in space, x_i, with a particular velocity, v_i, at a given time, t [8]. The number of particles within a cube of phase space with volume $\mathrm{d}x_i\mathrm{d}v_i$ at location, x_i, with velocity, v_i, is

$$N_{x_i,v_i} = \mathcal{F}\left(x_i, v_i, t\right) \mathrm{d}x_i \mathrm{d}v_i. \tag{2.1}$$

Therefore, the number density of particles can be found by integrating Eq. (2.1) over velocity space,

$$n\left(x_i, t\right) = \int_V \mathcal{F}\left(x_i, v_i, t\right) \mathrm{d}v_i. \tag{2.2}$$

2.2.1 The Maxwell-Boltzmann Distribution

A simple argument by Maxwell allows one to find the phase-space distribution function that describes a gas in local thermal equilibrium [9]. Local equilibrium is the state to which any gas, left alone, will relax due to inter particle collisions. Maxwell's four basic assumptions used to find the equilibrium state are [10]:

- The gas particles are small hard particles. Their shape is not changed during a collision.
- The distribution of particles is independent of time when the gas is in equilibrium.

- The distribution function is isotropic in equilibrium.
- The orthogonal components of the molecular velocities are statistically independent in equilibrium.

In such a situation, the velocity distribution function, known as a Maxwell-Boltzmann distribution, $\mathcal{M}$, can be found and is uniquely described by its temperature, T, mass density, ρ, and average velocity, u_i, [1] as

$$\mathcal{M} = \mathcal{F}\left(x_i, v_i, t\right) = n\left(\frac{\beta(x_i,t)}{\pi}\right)^{\frac{3}{2}} \exp^{(-\beta(x_i,t)(v_i-u_i(x_i,t))^2)}. \tag{2.3}$$

Here, β is given by

$$\beta\left(x_i, t\right) = \frac{m}{2} kT\left(x_i, t\right), \tag{2.4}$$

where $k = 1.38053 \times 10^{-3}\text{J/K}$ is the Boltzmann constant, m is the particle mass and $T\left(x_i, t\right)$ is the temperature of the fluid at location, x_i, and time, t.

It has been proven that a gas at any non-equilibrium state will always move toward the Maxwell-Boltzmann distribution due to particle collisions and, after reaching that state, particle collisions no longer have any effect on the distribution function.

2.3 The Boltzmann Equation

In the absence of external acceleration fields, the evolution of the distribution function is given by the Boltzmann equation,

$$\frac{\partial \mathcal{F}}{\partial t} + v_i \frac{\partial \mathcal{F}}{\partial x_i} = \frac{\delta \mathcal{F}}{\delta t}. \tag{2.5}$$

This equation appears to be very simple, however it is very high dimensional (three space dimensions, three velocity dimensions, plus time). The numerical expense of a direct discretization is therefore extremely high. The left-hand side describes particles moving through space, while the right-hand side is known as the collision operator and models the effects of inter-particle collisions on the distribution. In general, this collision operator is a complex integral relation in five dimensions and is extremely difficult to evaluate accurately.

2.3.1 The Collision Operator

Particle collisions involve momentum and energy exchange between particles. These exchanges cause the distribution function, $\mathcal{F}$, to change with time. At each collision, one can imagine two particles with the pre-collision velocities being removed from $\mathcal{F}$ and two particles with the post-collision velocities being added. As a result, in order to solve the Boltzmann equation, detailed information is required regarding these particle interactions. In order to model these collisions, several assumption must be made. First, it is assumed that only binary collisions occur. It is also assumed that there is a large enough number of particles so that no single collision can change the distribution function appreciably and that the distribution function is constant in space over the range of interaction potential forces between particles. Finally, it is assumed that particle velocities are uncorrelated. Using these assumptions and the laws of classical mechanics, it can be shown that the operator has the form

$$\frac{\delta \mathcal{F}}{\delta t} = \int_V \int_0^{2\pi} \int_0^{\frac{\pi}{2}} \left(\mathcal{F}' \mathcal{F}'^1 - \mathcal{F} \mathcal{F}^1 \right) g \, \sigma \sin\theta \mathrm{d}\theta \mathrm{d}\epsilon \mathrm{d}v_i^1, \tag{2.6}$$

where,

$$\mathcal{F} = \mathcal{F}\left(x, v_i, t\right), \mathcal{F}^1 = \mathcal{F}\left(x, v_i^1, t\right), \mathcal{F}' = \mathcal{F}\left(x, v_i', t\right), \mathcal{F}'^1 = \mathcal{F}\left(x, v_i'^1, t\right).$$

Here, v_i and v_i^1 are the velocities of two independent particles before a collision. Similarly, v_i' and $v_i'^1$ are the corresponding velocities for those particles after an interaction. In Eq. (2.6), the relative speed of the particles prior to the collision is given by $g = v_i - v_i'$, σ is the differential collision cross section of the particle, θ is the deflection angle, and ϵ is a solid angle. More details about the derivation of the collision operator integral can be found in any kinetic theory textbook [11, 8].

In general the collision operator is high-dimensional, complicated, and difficult to handle in practice. Therefore, simplified models are widely used to model the effects of particle interactions. One of the most common models was suggested by Bhatnagar, Gross, and Krook (BGK) in 1954 [12]. In this model, it is assumed that particles corresponding to a non-equilibrium state are removed and equilibrium-state particles are added on prescribed time scales,

$$\frac{\delta \mathcal{F}}{\delta t} = -\frac{\mathcal{F}\left(x_i, v_i, t\right)}{\tau_{\mathcal{F}}\left(x_i, t\right)} + \frac{\mathcal{M}\left(x_i, v_i, t\right)}{\tau_{\mathcal{M}}\left(x_i, t\right)}. \tag{2.7}$$

Again, the equilibrium Maxwellian distribution is denoted by $\mathcal{M}\left(x_i, v_i, t\right)$. The time scale on which non-equilibrium particles are removed is $\tau_{\mathcal{F}}$ and the time scale on which equilibrium particles are added is $\tau_{\mathcal{M}}$. These times are taken to be equal to ensure conservation of particles. The relaxation time, τ, can be chosen to produce models that agree with macroscopic transport coefficients. For example, the equilibrium fluid viscosity, μ, can be recovered by choosing $\mu = \tau p$. This model is also correct in the

limit of free-molecular flow, as $\tau \to \infty$. However, in the transition regime, its physical accuracy is less certain.

2.4 Moments of the Distribution Function

Fortunately, the huge amount of information given by $\mathcal{F}$ is rarely needed. Instead, macroscopic "observable" quantities are more often desired. Traditional variables are related to $\mathcal{F}$ through velocity moments. This means multiplying the distribution function by an appropriate velocity weight, $M(v)$, and integrating over all velocity space. For instance, the mass density of the gas, can be calculated as

$$\rho = \iiint_{\infty} m\mathcal{F}\,\mathrm{d}v_i = \langle m\mathcal{F} \rangle \ . \tag{2.8}$$

The short-hand rotation, $\langle \cdot \rangle$, denotes integration over all possible particle velocities. Similarity, the momentum density of the gas can also be calculated as

$$\rho u_i = \langle m v_i \mathcal{F} \rangle \ . \tag{2.9}$$

Using the definitions of mass density and momentum density, one can define the bulk velocity as

$$u_i = \frac{\langle m v_i \mathcal{F} \rangle}{\langle m \mathcal{F} \rangle}. \tag{2.10}$$

The bulk velocity can be used to define the random particle velocity as

$$c_i = v_i - u_i \, . \tag{2.11}$$

Using the random velocity, higher-order moments, which define some important physical properties such as pressure and heat flux, can be computed. For example, P_{ij} is

the second-order pressure tensor defined by

$$P_{ij} = m \iiint_{\infty} c_i c_j \mathcal{F} \, \mathrm{d}v_i = \langle m c_i c_j \mathcal{F} \rangle \ . \tag{2.12}$$

Here, for a monatomic gas, it is related to the traditional thermodynamic pressure as

$$p = P_{ii}/3, \tag{2.13}$$

and to the deviatoric (viscous) stress as

$$\tau_{ij} = \delta_{ij} p - P_{ij} \, . \tag{2.14}$$

A generalized heat-flux tensor can also be defined as

$$Q_{ijk} = m \iiint_{\infty} c_i c_j c_k \mathcal{F} \, \mathrm{d}v_i = \langle m c_i c_j c_k \mathcal{F} \rangle \ . \tag{2.15}$$

The third-order tensor, Q_{ijk}, is related to the traditional heat-flux vector as

$$q_i = \frac{1}{2} Q_{ijj} = \frac{1}{2} \langle m c_i c_j c_j \mathrm{F} \rangle \, . \tag{2.16}$$

2.5 Moment Closure

There are several different methods to solve or approximate the Boltzmann or BGK equation. The first approach is a direct treatment of each gas particle, which means solving for the molecular dynamics directly. This method obviously maximizes accuracy, however, the computational cost is enormous. Another option is the direct discretization of the Boltzmann equations in seven-dimensional phase space. This is also prohibitively expensive for all but the simplest problems [3]. One method that is affordable in some practical situations is the direct simulation Monte Carlo method

(DSMC) [2]. This method randomly chooses a subset of particles, and evolves them using statistical approximations. This method is simple to implement but converges very slowly and is excessively expensive for low-speed flows due to statistical noise. The Chapman Enskog method approximates the Boltzmann equation by extending the Navier-Stokes equations by adding terms with higher-order derivatives, but all models past the Navier-Stokes level are linearly unstable to high frequencies [11]. As a result, a simpler and more practical method is desired.

The method of moment closures, derived from the Boltzmann equation, promises an alternative technique to describe a monatomic gas's behaviour as compared to traditional techniques, such as the Euler or Navier-Stokes model. The theory of moment closures comes from the realization that the detailed information regarding the exact microscopic state of a gas is usually not important in the end. Rather, it is the macroscopic moments of the gas that are generally of interest. In this method, evolution equations for macroscopic moments can be obtained by taking a chosen set of moments of the Boltzmann equation directly [13]. For example, for an arbitrary velocity weight, M, an evolution equation for the corresponding moment can be found as

$$\frac{\partial}{\partial t}\left\langle mM\mathcal{F}\right\rangle + \frac{\partial}{\partial x_i}\left\langle mMv_i\mathcal{F}\right\rangle = \Delta(M)\,. \tag{2.17}$$

Here, $\Delta(M)$ represents the effect of collisions on the moment of interest. Normally, one is interested in the evolution of more than one moment. Therefore, a new notations is defined such that

$$\boldsymbol{M} = \left[M_0, M_1, ..., M_N\right]^T, \tag{2.18}$$

where $\boldsymbol{M}$ is a column vector including different weights corresponding to the moments of interest. Here, $N+1$ is the number of the entries in the vector. Using this vector, moment relations can be written in the vector form as

$$\boldsymbol{U} = \langle m\boldsymbol{M}\mathcal{F} \rangle \,. \tag{2.19}$$

Here, $\boldsymbol{U}$, is a solution vector containing an expanded number of velocity moments as compared to traditional methods. One can rewrite Eq. (2.17) in vector form as

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x_i} \langle m\boldsymbol{M}v_i\mathcal{F} \rangle = \Delta(\boldsymbol{M}) \,. \tag{2.20}$$

Introducing the flux dyad, $\boldsymbol{F}_i = \langle m\boldsymbol{M}v_i\mathcal{F} \rangle$, and $\Delta(\boldsymbol{M}) = \boldsymbol{S}$ as a local algebraic source term that is the effect of microscopic collisions on the macroscopic moments, a system of PDEs can be obtained that has the form

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x_i}\boldsymbol{F}_i = \boldsymbol{S} \,. \tag{2.21}$$

The first-order balance-law form of Eq. (2.21) is very appealing. Unfortunately, things are not so simple. In general, the flux dyad and source vector are not known functions of the solution vector. The system is not closed. This problem stems from the fact that the distribution function cannot be uniquely determined from the moment of the solution vector.

Fortunately, for many practical situations, simplified collision operators, such as BGK model, can be used. These allow the right-hand side of Eq. (2.21) to be determined as a function of the solution vector, $\boldsymbol{U}$. The flux dyad, however, is more difficult. As can be observed from Eq. (2.17), the flux will always contain moments that are one order higher in terms of velocity than the entry in $\boldsymbol{U}$.

In order to close the system, usually restrictions are placed on the form of allowable distribution functions, $\mathcal{F}$. This assumed form of $\mathcal{F}$ is chosen such that it contains the same number of free parameters as there are moments in $\boldsymbol{U}$. The value of these parameters is set such that the moment relations, such as those shown in Eqs. (2.8), (2.9), (2.12) and (2.15), are satisfied. Any moment needed for the flux dyad can then simply be integrated, as the assumed expression for $\mathcal{F}$ is then known. This technique is known as a moment closure.

The original moment-closure technique is due to Grad [13]. In this method, the distribution function is expanded around the local equilibrium Maxwellian in terms of polynomials of the particle velocity

$$\mathcal{F}(x_i, v_i, t) = \mathcal{M}(x_i, v_i, t)\ \left(\boldsymbol{\alpha}^T \boldsymbol{M}\right)\ . \tag{2.22}$$

Here, $\boldsymbol{M}$ is the vector that includes the generating weights and generally contains monomials of the particle velocity, v_i. The vector of coefficients of the distribution function, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_n]^T$, are chosen such that the moments in the solution vector, $\boldsymbol{U}$, satisfy Eq. (2.19). Although this technique gives a closed set of transport equations for a finite set of velocity moments, the resulting distribution function in a Grad-type moment closures can predict a negative number of particles at some velocities, which is not realistic. The other problem is that, even for small deviations of local equilibrium, Grad's moment closures lose hyperbolicity and yields moment equations that are ill-posed for initial value problems. This happens when the Jacobian of the flux vector develops complex eigenvalues.

2.5.1 Maximum-Entropy Moment Closures

More recently, a much more elegant choice for $\mathcal{F}$ has been proposed [14, 15, 16]. In this technique, the distribution function is assumed to have the form that is consistent with the moments present in $\boldsymbol{U}$, while maximizing entropy. Based on the Boltzmann H-theorem, the entropy density for a general non-equilibrium gas is

$$S = \langle \mathcal{F} \ln \mathcal{F} \rangle \,. \tag{2.23}$$

Using the method of Lagrange multipliers, one can show that the resulting form of $\mathcal{F}$ that maximizes this entropy subject to moment constraints is

$$\mathcal{F} = e^{\boldsymbol{\alpha}^T M} \,, \tag{2.24}$$

where $\boldsymbol{\alpha}^T M$ is a polynomial of the particle velocity. In contrast to the Grad method, this assumed form is guaranteed to be positive and can be shown to lead to hyperbolic PDEs whenever a distribution function of the form given in Eq. (2.24) can be found that corresponds to the moments in $\boldsymbol{U}$. Unfortunately, for high-order moment methods, there are valid physical gas states for which a corresponding distribution function of this form does not exist [17]. Initially this seemed to be a devastating blow to the maximum-entropy idea, however more recent progress has shown that these areas of non-definition can be handled in practice [18, 19]. In any event, the questions of existence of high-order moment closures is not the subject of the current work. All models used in the present study are of low enough order that well-posedness is guaranteed.

The maximum-entropy distribution is also pleasing due to the fact that particle

collisions are known to always cause the entropy to increase. Therefore, high entropy distribution are the most likely. The lowest-order member of this hierarchy is a 5-moment closure, which leads to the familiar Euler equations. It is found by taking the generating weights to be $\boldsymbol{M} = [m, mv_i, mv_iv_i]^T$. The Gaussian closure is the second lowest-order closure and is derived from the maximum-entropy theory by taking $\boldsymbol{M} = [m, mc_i, mc_ic_j]^T$ as weights. It consists of 10 moment equations and yield a strictly hyperbolic treatment for a compressible, viscous, adiabatic gas.

Moment models, derived through the maximum-entropy theory, bring many advantages as compared to the traditional Navier-Stokes equations. Firstly, the fact that they are derived from a more fundamental physical description of a gas means that they can produce physically accurate solutions in situations when the Navier-Stokes equations are an incorrect model [1, 19]. Typically, the solution vector in moment methods contain ten or more unknowns. These include the classical mass, momentum, and energy density, but also extra degrees of freedom for higher-order statistics of gas particle velocities. This brings added modelling fidelity in extreme situations and allows the model to maintain physical accuracy well out of the traditional continuum regime.

In addition to physical advantages, moment methods bring many numerical advantages as compared to the Navier-Stokes models. The fact that all derivatives are first order means that an extra order of spatial accuracy is often possible for a given stencil or polynomial representation of the approximate solution. For explicit methods, the time-step restriction is proportional to Δx, rather than $(\Delta x)^2$, which can lead to more

efficient time integration. Also, it has been demonstrated that the first-order nature of moment methods makes them far less sensitive to grid irregularities [1]. Lower-quality grids are often unavoidable for practical situations or when adaptive mesh refinement (AMR) is used. The use of models that maintain high-quality solutions on low quality meshes is very desirable.

Chapter 3

The Discontinuous-Galerkin-Hancock Method

3.1 Introduction

Moment methods hold the promise of new models for gas flow that possess many physical and mathematical advantages over traditional methods. Their first-order nature makes them well suited for solution on large parallel computers. Particularly, discontinuous-Galerkin (DG) methods seem like good candidates for their numerical solution. These DG methods have many properties that have lead to an increase in their popularity. The use of a DG discretization in space allows high-order accuracy, flexibility in handling complicated geometries, relatively easy application of boundary conditions, and the potential for high parallel efficiency.

The original discontinuous-Galerkin finite-element method was introduced by Reed and Hill [20] for solving the neutron transport equation. LeSaint and Raviart [21] made the first analysis of this method and proved a rate of convergence for a polynomial representation of degree k is $(\Delta x)^k$ for general triangulations and of $(\Delta x)^{k+1}$ for Cartesian grids. Later, Johnson and Pikaränta [22] proved a rate of convergence of $(\Delta x)^{k+\frac{1}{2}}$ for general triangulations and Peterson [23] confirmed this rate to be optimal. Richter [24] obtained the optimal rate of convergence of $(\Delta x)^{k+1}$ for some

structured two-dimensional non-Cartesian grids.

After success of this method for linear equations, Chavent and Salzano [25] constructed an explicit version of the DG method for one-dimensional scalar nonlinear hyperbolic conservation laws. To do this, they discretized in space by using the DG method with piecewise linear elements and then discretized in time using the simple implicit Euler method. Chavent and Cockburn [26] improved the stability of the scheme by introducing a suitably defined slope limiter following the ideas of van Leer [27]. The first Runge-Kutta discontinuous Galerkin (RKDG) method was introduced by Cockburn and Shu [28]. In this method, a piecewise linear DG method was used for the spacial discretization. This was combined with a special explicit total variation diminishing (TVD) second-order Runge-Kutta type discretization in time. This time integration scheme was developed by Shu and Osher in a finite difference framework [29, 30]. In the multidimensional case, the complicated geometry of the domain in practical applications can be handled by a DG space discretization and the TVD time discretization is also applicable, however defining a robust slope limiter is a main challenge.

For RKDG methods, a DG discretization is adopted only in space. Test functions are therefore just functions of space. A semi-discrete method-of-lines (MOL) discretization is used. These methods have reduced stability and more severe time-step restrictions than typical finite-volume methods. To overcome the stability restriction and reduced efficiency of the MOL approach, fully discrete methods can be considered. In these approaches, spatial and temporal operators are discretized in a similar

manner.

The target equations in this study are hyperbolic equations with a stiff relaxation source term that are derived from a moment closure approach. The numerical method presented in this work is a two-dimensional finite-element scheme that was originally proposed by Huynh [4] based on the upwind moment scheme for hyperbolic conservation laws. The specific scheme implemented here was developed by Suzuki and van Leer [5, 6] to solve hyperbolic-relaxation equations. In other words, it was created specifically to allow for the efficient solutions of hyperbolic PDEs that result from moment methods. Important aspects of the scheme are summarized here. The interested reader should consult Suzuki [6] for a detailed explanation and analysis of all aspects of the technique.

3.2 Discontinuous-Galerkin Hancock Method for Two-Dimensional Equations

3.2.1 Weak Formulation

Galerkin methods are highly successful finite-element methods. The DGH method used here is a coupled space-time method. That is to say, weak solutions of the governing PDEs,

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x_i}\boldsymbol{F}_i = \boldsymbol{S}, \tag{3.1}$$

are sought such that the product of the PDE with a test function, integrated through space and time should be satisfied,

$$\iint\limits_{\Omega\times T} \nu \frac{\partial}{\partial t}\boldsymbol{U}\,\mathrm{d}x_j\,\mathrm{d}t = -\iint\limits_{\Omega\times T} \nu \frac{\partial}{\partial x_i}\boldsymbol{F}_i\,\mathrm{d}x_j\,\mathrm{d}t + \iint\limits_{\Omega\times T} \nu\,\boldsymbol{S}\,\mathrm{d}x_j\,\mathrm{d}t\,. \tag{3.2}$$

Here, Ω is the domain made up of the space of the problem. The time difference between one instant and a future time is T. The Hancock method is adopted for time discretization for the time interval from time step n to $n+1$, and ν is an arbitrary test function. In order to obtain a practical numerical method, a finite number of test functions, ν, are selected.

Test functions are chosen to be non-zero in only one cell. The spatial domain of integration in Eq (3.2) can be reduced to that of one cell, Ω_k. Integration by parts is then used to remove derivatives from the solution and its flux,

$$\int\limits_{T} \nu \frac{\partial}{\partial t}\boldsymbol{U}\,\mathrm{d}t = \left(\boldsymbol{U}^{n+1}\nu - \boldsymbol{U}^{n}\nu\right) - \int\limits_{T} \boldsymbol{U}\frac{\partial \nu}{\partial t}\,\mathrm{d}t\,, \tag{3.3}$$

$$\int\limits_{\Omega_k} \nu \frac{\partial}{\partial x_i}\boldsymbol{F}_i\,\mathrm{d}x_j = \int\limits_{\Gamma_k} \nu \boldsymbol{F}_i \hat{n}_i\,\mathrm{d}\Gamma - \int\limits_{\Omega_k} \boldsymbol{F}_i \frac{\partial \nu}{\partial x_i}\,\mathrm{d}x_j\,. \tag{3.4}$$

Here, Γ_k is the boundary of the two-dimensional cell, and $\hat{n}$ is the outward unit vector normal on this boundary. The DG method relies on integration by parts. This is because the integration by parts transfers a differential operator acting on nonlinear functions to test functions. In both Eq. (3.3) and Eq. (3.4), derivatives of entries from the PDE are transferred to basis functions.

The test functions are chosen to be constant in time, by substituting Eq. (3.3) and Eq. (3.4) into Eq. (3.2) and applying Fubini's theorem (which allows to alternate the order of integration in space and time) the weak formulation simplifies to

$$\int_{\Omega_k} \nu \left(\boldsymbol{U}^{n+1} - \boldsymbol{U}^{n}\right) \mathrm{d}x_j = -\iint_{\Gamma_k \times T} \nu \boldsymbol{F}_i \hat{n}_i \,\mathrm{d}\Gamma \,\mathrm{d}t + \iint_{\Omega_k \times T} \boldsymbol{F}_i \frac{\partial \nu}{\partial x_i} \mathrm{d}x_j \,\mathrm{d}t + \iint_{\Omega_k \times T} \nu \boldsymbol{S} \,\mathrm{d}x_j \,\mathrm{d}t \,. \tag{3.5}$$

Eq. (3.5) is still exact in the weak form. Furthermore, the shape of the element, Ω_k, is not specified yet, any chosen polygon element satisfies Eq. (3.5).

The solution, $\boldsymbol{U}_\kappa$, in element Ω_κ is assumed to be a polynomial function. The last thing needed for an actual discretization method from the weak formulation Eq. (3.5) is choosing appropriate test functions. In this study the piecewise linear functions are taken as the space of polynomial functions.

For the present method, three test functions are chosen for each cell in a computational mesh. These test functions are only non-zero within one cell, in which they have the form

$$\nu_1 = 1 \,,\ \nu_2 = x - \tilde{x}_\kappa \,,\ \nu_3 = y - \tilde{y}_\kappa \,,$$

where $(\tilde{x_\kappa}, \tilde{y_\kappa})$ are the coordinates of the area centroid of the cell of interest. The constants $\tilde{x}_\kappa$ and $\tilde{y}_\kappa$ in the test function of the element K are defined by

$$\tilde{x}_\kappa = \frac{\iint\limits_{\Omega_K} x \,\mathrm{d}x\mathrm{d}y}{\iint\limits_{\Omega_K} \mathrm{d}x\mathrm{d}y}, \quad \tilde{y}_\kappa = \frac{\iint\limits_{\Omega_K} y \,\mathrm{d}x\mathrm{d}y}{\iint\limits_{\Omega_K} \mathrm{d}x\mathrm{d}y}. \tag{3.6}$$

The numerical solution variables are taken to be the cell average value, $\overline{\boldsymbol{U}_k}$, and average value of the x and y components of the solution gradient within each cell, $(\overline{\partial_x \boldsymbol{U}_k})$ and $(\overline{\partial_y \boldsymbol{U}_k})$. The assumed solution in the k^{th} cell is therefore

$$\boldsymbol{U}_\kappa(x,y) = \overline{\boldsymbol{U}}_\kappa + (\overline{\partial_x \boldsymbol{U}})_\kappa (x - \tilde{x}_\kappa) + (\overline{\partial_y \boldsymbol{U}})_\kappa (y - \tilde{y}_\kappa)\,. \tag{3.7}$$

Insertion of the assumed form of the solution into Eq. (3.5) and integration using each test function yields update formulas for the degrees of freedom as

$$\overline{\boldsymbol{U}}_\kappa^{n+1} = \overline{\boldsymbol{U}}_\kappa^{n} + \frac{1}{A_\kappa}\left(- \iint\limits_{\partial\Gamma_\kappa \times T} \boldsymbol{F}_i \hat{n}_i \,\mathrm{d}\Gamma \mathrm{d}t + \iint\limits_{\Omega_\kappa \times T} \boldsymbol{S} \,\mathrm{d}x_j \,\mathrm{d}t \right), \tag{3.8}$$

and

$$\begin{aligned} \begin{bmatrix} \overline{\partial_x \boldsymbol{U}}_\kappa^{n+1} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n+1} \end{bmatrix} = \begin{bmatrix} \overline{\partial_x \boldsymbol{U}}_\kappa^{n} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n} \end{bmatrix} + \overline{\boldsymbol{K}}_\kappa \Bigg(& - \iint\limits_{\partial\Gamma_\kappa \times T} \begin{bmatrix} x - \tilde{x}_\kappa \\ y - \tilde{y}_\kappa \end{bmatrix} \boldsymbol{F}_i \,\hat{n}_i \,\mathrm{d}\Gamma \,\mathrm{d}t \\ & + \iint\limits_{\Omega_\kappa \times T} \begin{bmatrix} \boldsymbol{F}_x \\ \boldsymbol{F}_y \end{bmatrix} \mathrm{d}x_j \,\mathrm{d}t + \iint\limits_{\Omega_\kappa \times T} \begin{bmatrix} x - \tilde{x}_\kappa \\ y - \tilde{y}_\kappa \end{bmatrix} \boldsymbol{S} \,\mathrm{d}x_j \,\mathrm{d}t \Bigg)\,. \end{aligned} \tag{3.9}$$

Here, A_κ is the area of cell κ and $\overline{\boldsymbol{K}}_\kappa$ is the matrix given by

$$\overline{\boldsymbol{K}}_\kappa = \begin{bmatrix} K_{xx} & K_{xy} \\ K_{xy} & K_{yy} \end{bmatrix}^{-1}, \tag{3.10}$$

with

$$K_{xx} = \iint_{\Omega_\kappa} (x - \tilde{x}_\kappa)^2 \, \mathrm{d}x_j \,, \quad K_{xy} = \iint_{\Omega_\kappa} (x - \tilde{x}_\kappa)(y - \tilde{y}_\kappa) \, \mathrm{d}x_j \,, \quad K_{yy} = \iint_{\Omega_\kappa} (y - \tilde{y}_\kappa)^2 \, \mathrm{d}x_j, \tag{3.11}$$

being the area moments of inertia of the cell. In this study only quadrilateral elements are used.

3.2.2 Cell-Average Update

As stated several times, the PDEs resulting from moment methods contain very stiff local source terms. To alleviate this problem, it is desirable to treat the source term implicitly. This does not generally add an excessive numerical expense, as the source terms are local and do not couple between cells. For the present method, Suzuki chose to use the Radau IIA method for the source-term time integration [6]. The Radau IIA method is a third-order accurate implicit two-step time-marching algorithm given by

$$y^{(n+\frac{1}{3})} = y^n + \frac{1}{3}\Delta t \left[\frac{5}{4} \left(\frac{\mathrm{d}y}{\mathrm{d}t} \right)^{(n+\frac{1}{3})} - \frac{1}{4} \left(\frac{\mathrm{d}y}{\mathrm{d}t} \right)^{(n+1)} \right], \tag{3.12}$$

$$y^{(n+1)} = y^n + \Delta t \left[\frac{3}{4} \left(\frac{\mathrm{d}y}{\mathrm{d}t} \right)^{(n+\frac{1}{3})} + \frac{1}{4} \left(\frac{\mathrm{d}y}{\mathrm{d}t} \right)^{(n+1)} \right]. \tag{3.13}$$

This method requires that an intermediate solution state be added at time level $n+\frac{1}{3}$.

The surface flux integrals in Eq. (3.8) are evaluated using the midpoint rule in the time dimension and two-point Gaussian quadrature on each edge of the cell for the space dimension, as shown in Fig 3.1. It shows that a quadrilateral element requires

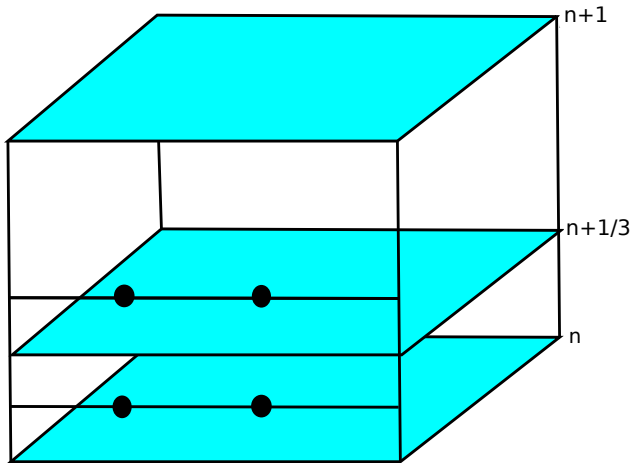


Figure 3.1: *Quadrature points for surface flux integration on a space-time element; blue slices represent instances in time when solution data is known or computed*

eight quadrature points at each time instance and will require a Riemann flux for each point. The weights at each Gauss point are $\frac{1}{2}$. To find the x, y coordinates of Gauss points x_{q1}, x_{q2} on the edge of the cell with length of ℓ and edge midpoint, x_m, one uses

$$x_{q1} = x_m - \frac{\ell}{2\sqrt{3}}, \tag{3.14}$$

$$x_{q2} = x_m + \frac{\ell}{2\sqrt{3}}. \tag{3.15}$$

As the solution is discontinuous along the edge, a numerical flux function must be used. In the present work, the HLLE flux function is used [31, 32]. As the Radau IIA method requires the solution at an intermediate $n + \frac{1}{3}$ step be calculated, four quadrature points per edge are needed. The quadrature points used for inter-cellular flux calculations are shown graphically in Figure 3.2. For this quadrature rule, the solution at the half-time for each update is needed as inputs to the flux functions.

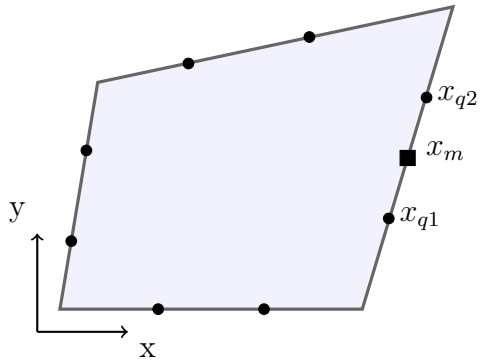


Figure 3.2: *Quadrature points for boundary flux calculations. The* (•) *denotes Gaussian quadrature points where a Riemann flux is computed. The* (■) *is a edge midpoint*

Predicting values at these points is Hancock's contribution to the method [7].

Based on Hancock's observation, flow quantities at cell centres evolve over the half time step without any interactions with neighbours. Hence, from the update formula for conserved variables, Eq. (3.8), element-face interactions can be removed

$$\overline{\boldsymbol{U}}_{\kappa}^{n+\psi} = \overline{\boldsymbol{U}}_{\kappa}^{n} + \frac{1}{A_{\kappa}} \left(- \iint\limits_{\partial\Gamma_{\kappa}\times T'} \hat{\boldsymbol{F}}_i \hat{n}_i \, \mathrm{d}\Gamma \mathrm{d}t + \iint\limits_{\Omega_{\kappa}\times T'} \boldsymbol{S} \, \mathrm{d}x_j \, \mathrm{d}t \right) . \tag{3.16}$$

Here T' is the time interval from n to $n+\psi$, and the flux tensor $\hat{\boldsymbol{F}}_i$ is simply evaluated using the interior solution at each Gaussian quadrature point of the element, obtained by Eq. (3.7). The predictor step, Eq. (3.16), to compute the cell average at $n+\frac{1}{6}$ and $n+\frac{1}{2}$ can be further simplified with the approximation of evaluating the flux integral at time n, and source term integral at $n+\psi$,

$$\overline{\boldsymbol{U}}_{\kappa}^{n+\psi} = \overline{\boldsymbol{U}}_{\kappa}^{n} + \frac{\psi\Delta t}{A_{\kappa}} \left(- \int\limits_{\partial\Gamma_{\kappa}} \boldsymbol{F}_i \hat{n}_i \, \mathrm{d}\Gamma + \int\limits_{\Omega_{\kappa}} \boldsymbol{S} \, \mathrm{d}x_j \right) . \tag{3.17}$$

Once the predicted cell-average quantity $\overline{\boldsymbol{U}}_{\kappa}^{n+\psi}$ in element Ω_κ at time-level $\psi = \frac{1}{2}$ or $\psi = \frac{1}{6}$ is obtained, the distribution of the solution along the edge Γ_κ can be computed by

$$\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(x,\ y) = \overline{\boldsymbol{U}}_{\kappa}^{n} + \boldsymbol{\phi}_\kappa \wedge (\overline{\partial_x \boldsymbol{U}}_{\kappa}^{n})\,(x - \tilde{x}_\kappa) + \boldsymbol{\phi}_\kappa \wedge (\overline{\partial_y \boldsymbol{U}}_{\kappa}^{n})\,(y - \tilde{x}_\kappa). \tag{3.18}$$

Here, $\boldsymbol{\phi}$ is a vector of slope limiters that may be used to ensure stability of discontinuous solutions. Note that the slope variables $\overline{\partial_x \boldsymbol{U}}_{\kappa}^{n}$ and $\overline{\partial_y \boldsymbol{U}}_{\kappa}^{n}$ are still associated with the time at n. In order to make the time integration of the source term implicit, the spatial integral of the source term is evaluated with the solution at just one Gauss point, which is centroid of the element Ω_κ,

$$\int_{\Omega_\kappa} \boldsymbol{S}\,\mathrm{d}x_j \approx A_\kappa\ \boldsymbol{S}(\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(\tilde{x}_\kappa,\ \tilde{y}_\kappa)). \tag{3.19}$$

Inserting Eq. (3.19) and Eq. (3.18) into Eq. (3.17), the solution state that is used for the input to inter-cell flux calculations is predicted using data that is internal to the cell. For this, the sub-cell solution is approximated as

$$\begin{aligned}\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(x,\ y) = \overline{\boldsymbol{U}}_{\kappa}^{n} + \boldsymbol{\phi}_\kappa \wedge (\overline{\partial_x \boldsymbol{U}}_{\kappa}^{n})\,(x - \tilde{x}_\kappa) + \boldsymbol{\phi}_\kappa \wedge (\overline{\partial_y \boldsymbol{U}}_{\kappa}^{n})\,(y - \tilde{x}_\kappa) \\ - \frac{\psi\ \Delta t}{A_\kappa} \int_{\partial\Gamma_\kappa} \boldsymbol{F}_i(\tilde{\boldsymbol{U}}_{\kappa}^{n}) \hat{n}_i \mathrm{d}\Gamma + \psi\ \Delta t \boldsymbol{S}(\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(\tilde{x}_\kappa,\ \tilde{y}_\kappa))\,.\end{aligned} \tag{3.20}$$

Here $\wedge$ denotes a term-wise product, and ψ will be either a half or a sixth. The path integral is evaluated using the solution state within the cell and the jump at the cell boundary is not considered. As the source term is treated implicitly, this can lead to

a nonlinear system of equations. Suzuki advocates using Newton's method, however, for the present work, the linearization

$$\boldsymbol{S}(\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(x,\ y)) = \boldsymbol{S}(\tilde{\boldsymbol{U}}_{\kappa}^{n}(x,\ y)) + \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}^{n} \left(\tilde{\boldsymbol{U}}_{\kappa}^{n+\psi}(x,\ y) - \tilde{\boldsymbol{U}}_{\kappa}^{n}(x,\ y)\right) \tag{3.21}$$

is used. This leads to a linear system that can be solved efficiently.

In order to find the final update for the cell average, one should approximate the two integrals in the right hand side of Eq. (3.8). The inter-cell flux along $\mathrm{d}\Gamma$ is obtained by a standard Riemann solver

$$\boldsymbol{F}_i \hat{n}_i \approx \tilde{\boldsymbol{F}}_{\zeta}, \tag{3.22}$$

where $\tilde{\boldsymbol{F}}_{\zeta}$ is the vector of fluxes normal to the edge Γ_{κ} projecting from element Ω_{κ}. Even though multidimensional problems are considered here, a Riemann solver is still based on one-dimensional physics, as it is in standard finite-volume methods. The surface integral of a flux can be evaluated exactly if the flux is linear. When a flux is nonlinear, a standard approach is to approximate the integral by a quadrature. If the solution is represented in a polynomial space P^k, then a quadrature for the spatial integration along edges must be exact for a polynomial of degree $2k+1$ [33]. Thus the spatial integration is replaced by a two-point Gaussian quadrature, which is exact for polynomial of degree three. The resulting quadrature rule is

$$\iint\limits_{\partial\Gamma_{\kappa}\times T} \boldsymbol{F}_i \hat{n}_i \,\mathrm{d}\Gamma \mathrm{d}t \approx \int\limits_{T} \sum_{\zeta} w_{\zeta} \tilde{\boldsymbol{F}}_{\zeta} \mathrm{d}t. \tag{3.23}$$

As mentioned, the Radau IIA method is used for the source-term time integration. The source term integration in space is evaluated at only one quadrature point in the cell centre. This integral for approximating the integral from time step n to $n+1$ becomes

$$\iint\limits_{\Omega_\kappa \times T} \boldsymbol{S}\,\mathrm{d}x_j\,\mathrm{d}t \approx A_\kappa \Delta t \left(\frac{3}{4}\overline{\boldsymbol{S}}^{n+\frac{1}{3}} + \frac{1}{4}\overline{\boldsymbol{S}}^{n+1}\right). \tag{3.24}$$

The final update formula for the cell-average values to $T = n+1$ and $T = n+\frac{1}{3}$ is

$$\begin{bmatrix} \overline{\boldsymbol{U}}_\kappa^{n+\frac{1}{3}} \\ \overline{\boldsymbol{U}}_\kappa^{n+1} \end{bmatrix} = \begin{bmatrix} \overline{\boldsymbol{U}}_\kappa^{n} \\ \overline{\boldsymbol{U}}_\kappa^{n} \end{bmatrix} - \frac{\Delta t}{A_\kappa} \begin{bmatrix} \frac{1}{3}\sum\limits_\zeta w_\zeta \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{6}} \\ \sum\limits_\zeta w_\zeta \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{2}} \end{bmatrix} + \Delta t \begin{bmatrix} \frac{5}{12}\boldsymbol{I} & -\frac{1}{12}\boldsymbol{I} \\ \frac{3}{4}\boldsymbol{I} & \frac{1}{4}\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \overline{\boldsymbol{S}}^{n+\frac{1}{3}} \\ \overline{\boldsymbol{S}}^{n+1} \end{bmatrix}. \tag{3.25}$$

Here, the path integrals around the cell boundary have been replaced by quadrature rules with weights, w_ζ. These weights include the edge length. The inter-cell flux function, $\tilde{\boldsymbol{F}}$, is evaluated using Eq. (3.20) to produce the inputs on either side of the edge. The source term is evaluated at the cell centre. It is again linearized as

$$\overline{\boldsymbol{S}}_\kappa^{n+\Psi} \approx \overline{\boldsymbol{S}}_\kappa^{n} + \left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^n \left(\overline{\boldsymbol{U}}_\kappa^{n+\Psi} - \overline{\boldsymbol{U}}_\kappa^{n}\right). \tag{3.26}$$

3.2.3 Slope Update

Once the cell average values at time intervals $n+\frac{1}{3}$ and $n+1$ are known, the updates for the slopes can be computed.

The update to slope information is given by Eq. (3.9), to evaluate this equation, one should approximate the three integrals on the right-hand side of this equation. The previously computed inter-cell fluxes are used again in the surface flux integral from Eq. (3.9). The surface integral in the slope update becomes

$$\iint\limits_{\partial\Gamma_\kappa\times T} \begin{bmatrix} x-\tilde{x}_\kappa \\ y-\tilde{y}_\kappa \end{bmatrix} \boldsymbol{F}_i\,\hat{n}_i\,\mathrm{d}\Gamma\;\mathrm{d}t \approx \Delta t \begin{bmatrix} \sum\limits_\zeta w_\zeta(x_\zeta-\tilde{x}_\kappa)\;\tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{2}} \\ \sum\limits_\zeta w_\zeta(y_\zeta-\tilde{y}_\kappa)\;\tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{2}} \end{bmatrix}, \tag{3.27}$$

where $w_\zeta = \frac{1}{2}\ell$ is the weight for quadrature in space.

Volume integrals of the flux appear in the update formula of the slope quantities in Eq. (3.9). The time volume integral is again approximated by the two-point Radau IIA quadrature in time such that

$$\iint\limits_{\Omega_\kappa\times T} \begin{bmatrix} \boldsymbol{F}_x \\ \boldsymbol{F}_y \end{bmatrix} \mathrm{d}x_j\,\mathrm{d}t \approx \Delta t \int\limits_{\Omega_\kappa} \begin{bmatrix} \left(\frac{3}{4}(\boldsymbol{F}_x)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_x)_\eta^{n+1}\right) \\ \left(\frac{3}{4}(\boldsymbol{F}_y)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_y)_\eta^{n+1}\right) \end{bmatrix} \mathrm{d}x_j. \tag{3.28}$$

The volume-flux integral is evaluated using two-dimensional, four-point Gaussian quadrature based on an assumed solution where the cell-average has been updated, but slopes at time level n are still used. Quadrature locations for the slope update are shown in Figure 3.3.

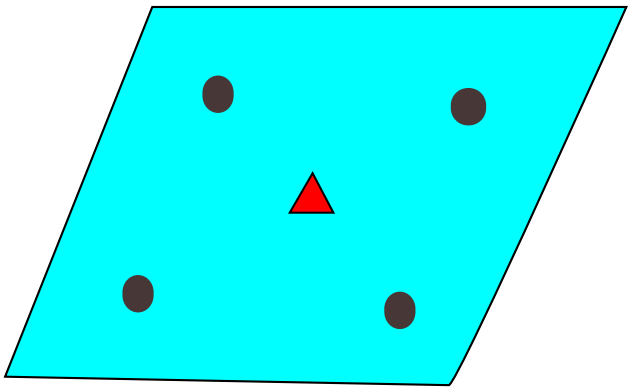


Figure 3.3: *Quadrature points for volume flux integral (•) and source integral (△) on a space element.*

For a reference element with $-1.0 < \alpha < 1.0$ and $-1.0 < \beta < 1.0$, the position of these Gauss points, $\eta(\alpha, \beta)$, are

$$\eta_0 = \left(-\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right), \ \eta_1 = \left(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right), \tag{3.29}$$

$$\eta_2 = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right), \ \eta_3 = \left(-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right). \tag{3.30}$$

This is simply the Cartesian product of the one-dimensional Gaussian quadrature rule. The quadrilateral cells used in this work can be any convex four-sided polygon. Therefore, positions of quadratures for evaluation solutions should be transferred to local coordinate. The solution at Gauss point ($\mathrm{u}(x)$) is found by Eq. (3.20) and let J be the Jacobian matrix of the coordinate transformation from η to x. The coordinate transformation between the global element K and the local element $\hat{K}$ is

$$\boldsymbol{x}(\alpha, \beta) = \frac{1-\alpha}{2}\frac{1-\beta}{2}\boldsymbol{x}_0 + \frac{1+\alpha}{2}\frac{1-\beta}{2}\boldsymbol{x}_1 + \frac{1+\alpha}{2}\frac{1+\beta}{2}\boldsymbol{x}_2 + \frac{1-\alpha}{2}\frac{1+\beta}{2}\boldsymbol{x}_3. \tag{3.31}$$

Here the x_i with $i = 0, 1, 2, 3$, are the positions of the nodes making up one element

of the mesh. These quadrature points are used to approximate the space volume integral using four-point Gaussian quadrature, such that

$$\int_K \mathrm{f}\left(\mathrm{u}\left(x\right)\right)\ \mathrm{d}x_j\ = \int_{\hat{K}} \mathrm{f}\left(\mathrm{u}\left(\eta\right)\right)\ \left|\mathrm{J}\left(\eta\right)\right|\mathrm{d}\eta\ \approx \sum_{i=0}^{3} w_i |\mathrm{J}\left(\eta_i\right)|\,\mathrm{f}\left(\mathrm{u}\left(\eta_i\right)\right). \tag{3.32}$$

Here the determinant of the Jacobian becomes,

$$\left|\mathrm{J}\left(\eta\right)\right| = \frac{\partial x}{\partial \alpha}\frac{\partial y}{\partial \beta} - \frac{\partial x}{\partial \beta}\frac{\partial y}{\partial \alpha}. \tag{3.33}$$

Finally, the volume integral of the flux in space and time is replaced by quadrature

$$\iint_{\partial\Gamma_\kappa \times T} \left[\begin{array}{c} x-\tilde{x}_\kappa \\ y-\tilde{y}_\kappa \end{array}\right] \boldsymbol{F}_i\,\hat{n}_i\,\mathrm{d}\Gamma\ \mathrm{d}t \approx \left[\begin{array}{c} \sum\limits_\eta w_\eta\ \left(\frac{3}{4}(\boldsymbol{F}_x)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_x)_\eta^{n+1}\right) \\ \sum\limits_\eta w_\eta\ \left(\frac{3}{4}(\boldsymbol{F}_y)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_y)_\eta^{n+1}\right) \end{array}\right]. \tag{3.34}$$

Here weights for all four quadrature points are $w_\eta = \left|\mathrm{J}\left(\eta\right)\right|$. The source term is linearized, in order to approximate the volume integral of the moment of the source term that appears in Eq. (3.9)

$$\boldsymbol{S}(\tilde{\boldsymbol{U}}_\kappa^n(x,\ y)) \approx \boldsymbol{S}(\tilde{\boldsymbol{U}}_\kappa^n(\tilde{x}_\kappa,\ \tilde{y}_\kappa)) + \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}^n \left(\overline{\partial_x \boldsymbol{U}}_\kappa^n (x_\zeta - \tilde{x}_\kappa) + \overline{\partial_y \boldsymbol{U}}_\kappa^n (y_\zeta - \tilde{y}_\kappa)\right). \tag{3.35}$$

Here, $(\tilde{x}_\kappa, \tilde{y}_\kappa)$ is the centroid of the element. As was shown in Figure 3.3, the volume integral is evaluated at the centroid. Hence, the spatial integration is done analytically, while the two-point Radau IIA quadrature is again applied in time. The resulting formulas become

$$\iint\limits_{\Omega_\kappa \times T} \begin{bmatrix} x-\tilde{x}_\kappa \\ y-\tilde{y}_\kappa \end{bmatrix} \boldsymbol{S}\,\mathrm{d}x_j\,\mathrm{d}t \approx \Delta t \begin{bmatrix} \frac{3}{4}\left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+\frac{1}{3}} & \left(\overline{\partial_x \boldsymbol{U}}\right)_\kappa^{n+\frac{1}{3}} + \frac{1}{4}\left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+1} & \left(\overline{\partial_x \boldsymbol{U}}\right)_\kappa^{n+1} \\ \frac{3}{4}\left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+\frac{1}{3}} & \left(\overline{\partial_y \boldsymbol{U}}\right)_\kappa^{n+\frac{1}{3}} + \frac{1}{4}\left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+1} & \left(\overline{\partial_y \boldsymbol{U}}\right)_\kappa^{n+1} \end{bmatrix}. \tag{3.36}$$

Once the cell-averaged variables at time levels n, $n+\frac{1}{3}$ and $n+1$, are known, volume integrals of the flux can be evaluated explicitly. Then the final update formulas for slope gradients are implicit with respect to the source term. The final update formula for the slopes in discrete form are

$$\begin{bmatrix} \overline{\partial_x \boldsymbol{U}}_\kappa^{n+\frac{1}{3}} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n+\frac{1}{3}} \\ \overline{\partial_x \boldsymbol{U}}_\kappa^{n+1} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n+1} \end{bmatrix} = \begin{bmatrix} \overline{\partial_x \boldsymbol{U}}_\kappa^{n} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n} \\ \overline{\partial_x \boldsymbol{U}}_\kappa^{n} \\ \overline{\partial_y \boldsymbol{U}}_\kappa^{n} \end{bmatrix} + \Delta t \boldsymbol{\Upsilon} \begin{bmatrix} \frac{1}{3}\sum_\zeta w_\zeta (x_\zeta - \tilde{x}_\kappa)\ \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{6}} \\ \frac{1}{3}\sum_\zeta w_\zeta (y_\zeta - \tilde{y}_\kappa)\ \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{6}} \\ \sum_\zeta w_\zeta (x_\zeta - \tilde{x}_\kappa)\ \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{2}} \\ \sum_\zeta w_\zeta (y_\zeta - \tilde{y}_\kappa)\ \tilde{\boldsymbol{F}}_\zeta^{n+\frac{1}{2}} \end{bmatrix}$$

$$+\Delta t \boldsymbol{\Upsilon} \begin{bmatrix} \frac{1}{3}\sum_\eta w_\eta \left(\frac{1}{2}(\boldsymbol{F}_x)_\eta^n + \frac{1}{2}(\boldsymbol{F}_x)_\eta^{n+\frac{1}{3}}\right) \\ \frac{1}{3}\sum_\eta w_\eta \left(\frac{1}{2}(\boldsymbol{F}_y)_\eta^n + \frac{1}{2}(\boldsymbol{F}_y)_\eta^{n+\frac{1}{3}}\right) \\ \sum_\eta w_\eta \left(\frac{3}{4}(\boldsymbol{F}_x)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_x)_\eta^{n+1}\right) \\ \sum_\eta w_\eta \left(\frac{3}{4}(\boldsymbol{F}_y)_\eta^{n+\frac{1}{3}} + \frac{1}{4}(\boldsymbol{F}_y)_\eta^{n+1}\right) \end{bmatrix} + \Delta t \begin{bmatrix} \frac{5}{12}\boldsymbol{I} & -\frac{1}{12}\boldsymbol{I} \\ \frac{3}{4}\boldsymbol{I} & \frac{1}{4}\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+\frac{1}{3}} & \left(\overline{\partial_x \boldsymbol{U}}\right)_\kappa^{n+\frac{1}{3}} \\ \left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+\frac{1}{3}} & \left(\overline{\partial_y \boldsymbol{U}}\right)_\kappa^{n+\frac{1}{3}} \\ \left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+1} & \left(\overline{\partial_x \boldsymbol{U}}\right)_\kappa^{n+1} \\ \left(\overline{\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}}}\right)_\kappa^{n+1} & \left(\overline{\partial_y \boldsymbol{U}}\right)_\kappa^{n+1} \end{bmatrix}. \tag{3.37}$$

Here, η is the index through the four-point flux-volume quadrature with w_η as the corresponding weight, $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{\Upsilon}$ is the matrix given by

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \overline{\boldsymbol{K}}_\kappa & 0 \\ 0 & \overline{\boldsymbol{K}}_\kappa \end{bmatrix}. \tag{3.38}$$

This completes the discretization of the DGH method for two-dimensional problems.

Though only linear basis functions are used, this scheme results in third-order accuracy in both space and time, the update in each cell requires neighbouring information only once. Note that, the maximum stable CFL number for this scheme is 0.664 [6]. It can therefore be said to be optimally efficient for parallel computation as only one message needs to be sent between any two processes during a single time step.

3.3 Moments of Inertia

In Eq. (3.38), $\overline{\boldsymbol{K}}_{\kappa}$ is the inverse of the matrix of moments of inertia for any simple polygon about its centroid which is defined as:

$$\overline{\boldsymbol{K}}_{\kappa} = \begin{bmatrix} K_{11} & K_{12} \\ K_{12} & K_{22} \end{bmatrix}. \tag{3.39}$$

Here,

$$K_{11} = \frac{K_{yy}}{K_{yy}\,K_{xx} - K_{xy}^2},\; K_{12} = \frac{-K_{xy}}{K_{yy}\,K_{xx} - K_{xy}^2},\; K_{22} = \frac{K_{xx}}{K_{yy}\,K_{xx} - K_{xy}^2}. \tag{3.40}$$

Where, K_{xx}, K_{yy} and K_{xy} are defined in Eq. (3.11) , to compute the moments of inertia of any simple polygon shape about the centroid. The numerical computation

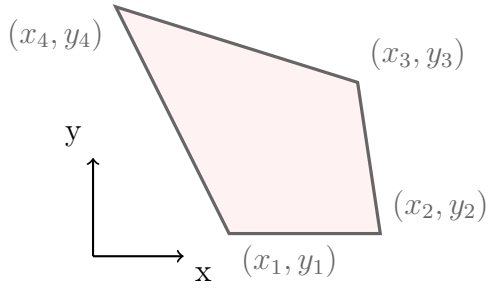


Figure 3.4: *Simple polygon*

for K_{xx}, K_{yy} and K_{xy}, in any simple polygon (Figure 3.4) can be found from

$$K_{yy} = \frac{1}{12}\sum_{i=1}^{n}(\hat{y_i}^2 + \hat{y_i}\hat{y}_{i+1} + \hat{y}_{i+1}^2)(\hat{x_i}\hat{y}_{i+1} - \hat{x}_{i+1}\hat{y_i})\,, \tag{3.41}$$

$$K_{xx} = \frac{1}{12}\sum_{i=1}^{n}(\hat{x_i}^2 + \hat{x_i}\hat{x}_{i+1} + \hat{x}_{i+1}^2)(\hat{x_i}\hat{y}_{i+1} - \hat{x}_{i+1}\hat{y_i})\,, \tag{3.42}$$

$$K_{xy} = \frac{1}{24}\sum_{i=1}^{n}(\hat{x_i}\hat{y}_{i+1} + 2\hat{x_i}\hat{y_i} + 2\hat{x}_{i+1}\hat{y}_{i+1} + \hat{x}_{i+1}\hat{y_i})(\hat{x_i}\hat{y}_{i+1} - \hat{x}_{i+1}\hat{y_i})\,. \tag{3.43}$$

Here $\hat{x_i}$ and $\hat{y_i}$ are $(x_i - \tilde{x_\kappa})$ and $(y_i - \tilde{y_\kappa})$, respectively, and n is the number of nodes. They should be listed in the order that they are connected on the boundary, either in clockwise or counterclockwise order.

The cell area for any simple polygon can be found by

$$|A_k| = \frac{1}{2}\left|\sum_{i=1}^{n} x_i\,(y_{i+1} - y_{i-1})\right|. \tag{3.44}$$

Chapter 4

Adaptive Mesh Refinement Scheme

4.1 Block-Based Mesh Refinement

The discontinuous-Galerkin-Hancock method is implemented within a large-scale framework for the parallel solution of hyperbolic-relaxation PDEs. The scheme uses block-based solution-directed adaptive mesh refinement on a body-fitted grid in a manner very similar to that developed by Sachdev *et al.* [34] and Gao and Groth [35]. Each block is a quadrilateral structured mesh and all blocks contain the same number of cells. If a particular block is flagged for refinement, it is subdivided into either four child blocks by doubling the resolution in both logical directions, or two blocks by doubling the resolution in one direction. Each child block has the same number of cells as the parent. Efficient load balancing for parallel calculations is achieved simply by dividing the blocks among available processors. Communication between neighbouring blocks is coordinated such that any two CPUs exchange at most one message per time step. Boundary communication between blocks is organized in such a way that blocks of any level of refinement can neighbour each other. That is to say, a cell on the edge of one block can have any number of neighbours. This allows for very flexible and localized mesh refinement.

4.2 Two Dimensional Blocks

In this study only two dimensional meshes consisting of quadrilateral elements have been used. Cells and nodes in these meshes are described using $i-j$ indexing. During a computation, each block can be refined in the i direction, the j direction, or both simultaneously, as shown in Fig. 4.1. Refining in these directions does not change the total number of cells in each block but increases the number of blocks. This approach, is useful when one wants to spread blocks on several CPUs for parallel calculation, so one can put an equal computational load on each CPU. For instance, when we have a single Cartesian block that has lengths of a and b in the x and y directions and it is refined in the i direction, the new blocks have the same length in the y direction as the parent, but the length of each block in the x direction is now $\frac{a}{2}$. Note that, in this study lengths are always divided by two in both directions during refinement. In Fig. 4.1, a parent block is shown that has four cells, this mesh is refined twice in both the i and j directions. As shown, the result is seven individual blocks that have four cells each, but they are all connected together, therefore there are more cells in bottom right of domain.

As the mesh changes dynamically during a calculation, the connectivity of blocks must be maintained across a large distributed computer. The present implementation is designed such that global connectivity information is never stored on one CPU. Only information about a block's immediate neighbours is stored locally. This is important if the code is to eventually scale to massive numbers of CPUs, as global information would eventually become too much to store on each computational process.

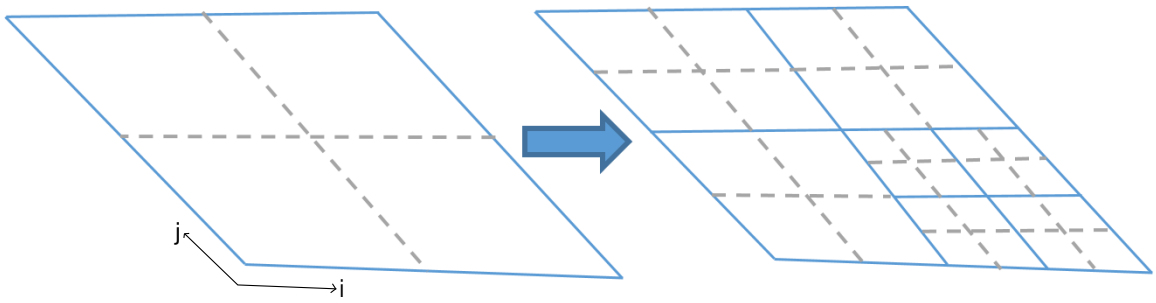


Figure 4.1: *Example of block-based mesh refinement. A single block is subdivided twice, resulting in seven blocks.*

4.3 Block Signature

In the current approach, each original block is assigned a number, which is called a root number. When one block is refined, in any directions, this is called a refinement event. Refinement events are added to an ordered list of such events that have lead to the creation of the current block.

A block's signature, therefore, consists of a root number and a list of refinement events that uniquely identify the "history" of each block. Refinement events are stored using an "enum" data type, which is simply an integer. Fig. 4.2 shows an example how block signature is changing for one block that is refined twice in the i and j directions.

In the parallel implementation, it is important that these block signatures be order-able. In the present implementation, signatures are first ordered based on root number. If two blocks have the same root, their refinement histories are then compared lexicographically.

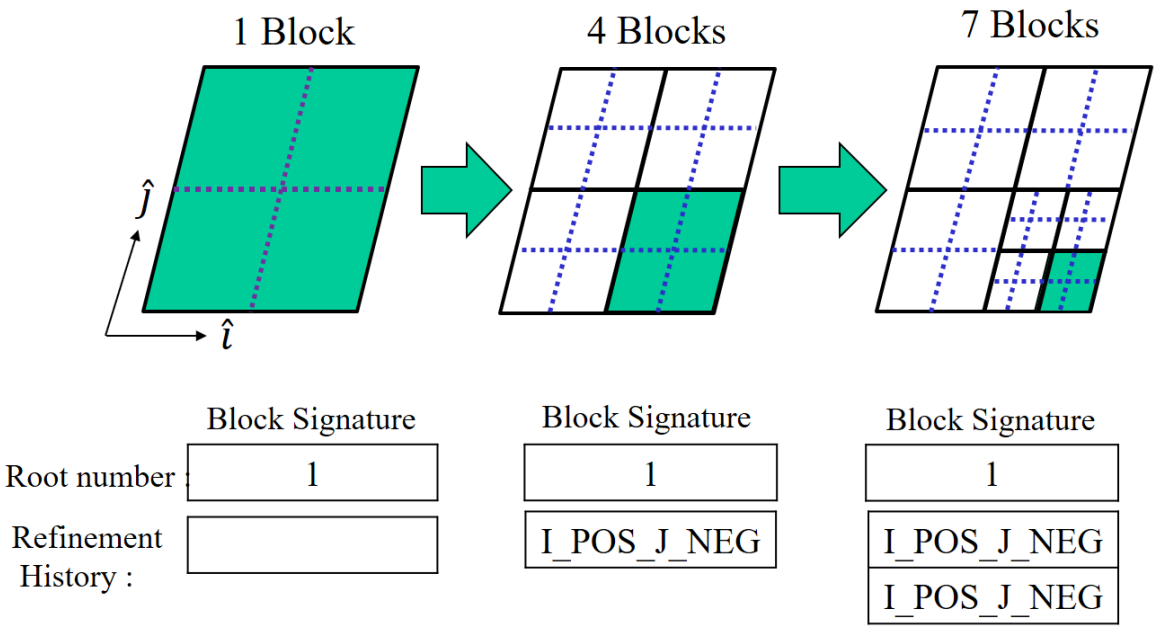


Figure 4.2: *Example of block signature change after refining. A single block with root number if 1 is subdivided twice, block signature history is shown.*

4.4 Boundary Connections

Once different blocks are generated, they should be connected together in order to calculate fluxes that pass between the blocks. Therefore, another signature is defined as the boundary signature, which is unique for each boundary edge between elements on different blocks. Note that this signature is for boundary edges that are connected to other blocks—boundaries that lie on the domain of the problem have no boundary signature.

The boundary signature contains the block signatures for each side. Furthermore, it contains a boundary index for each side, this way each edge connecting blocks has a unique boundary signature. Here, the boundary index is an integer number that is used to uniquely identify each boundary edge on a block. It is possible for one edge to

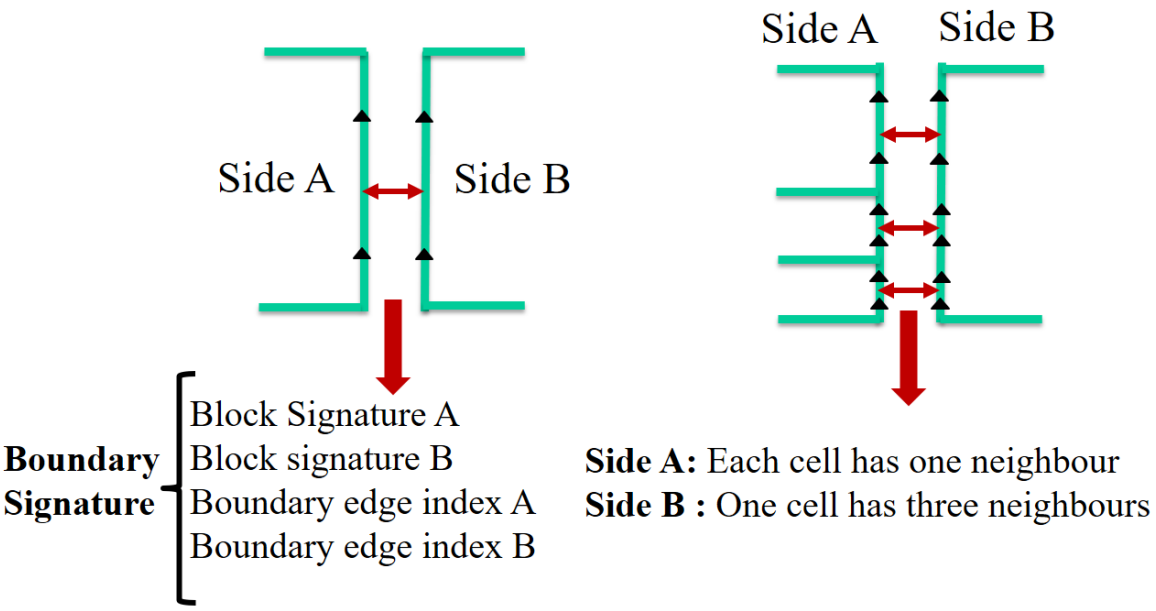


Figure 4.3: *Example of boundary signature for two different situation. A single edge connected to another single edge before refinement and tree cells that connected to one cell after refinement*

share an edge with multiple neighbours. Such an edge would store multiple boundary signatures, as demonstrated in Fig. 4.3. Note that, each boundary edge has a side defined to be either side ***A*** or ***B***. These labels are set when boundaries are created.

As with the block signatures, boundary signatures also need to be sort-able. This way, one can order them, when needed. Toward this aim, first we check if block signature in side ***A*** of each signature is bigger than signature for side ***B***, then boundary index for side ***A*** and ***B*** will be checked, respectively. Element edges that neighbour cells on a different block store an array of boundary signatures and an array containing the CPU number on which the neighbour exists.

4.5 The Comm Hub

In order to minimize and coordinate inter-CPU communication, each process contains a comm hub. This object takes a list of all inter-block element boundaries and the CPU that is responsible for their neighbour. Separate ordered lists of boundary signatures corresponding to each neighbour's CPU are constructed. As the signatures are properly sortable, two neighbouring CPUs will independently generate identical lists of their shared edges. Send and receive buffers are allocated with enough storage for all information that must be shared.

Based on their boundary signature, each element edge is given two pointers, one into its space in the send buffer and one into its space in the receive buffer. The send buffers are filled as early as possible in a time step. Then, while elements that are internal to each block are being updated, the messages are sent through the network. Boundary values are read from the receive buffer as late as possible in a time step, in order to minimize communication related bottlenecks. During mesh refinement and load balancing, information must be shared between neighbouring blocks. This is also done through the comm hub.

Chapter 5

Numerical Results

Five demonstration cases are considered here. First, a linear hyperbolic-relaxation equation is used to verify the order of accuracy of the scheme. Next a supersonic compressible Euler case is used to demonstrate the mesh refinement as well as the scheme's ability to capture sharp discontinuities. Third, a ten-moment closure is used to compute a viscous mixing layer. This serves to demonstrate the ability of the first-order PDEs and the DG scheme to efficiently compute viscous solutions. Fourth, a moment-closure is used to compute the solution for Stokes flow past a circular cylinder. This case reinforces the hyperbolic PDEs' ability to accurately predict viscous phenomena. As this case is very low speed, it also demonstrates the numerical technique's ability to accurately solve problems that are highly ill-conditioned due to the extremely low Mach number. Finally, the parallel efficiency for this algorithm on massive modern computing facilities is briefly assessed. In this study, continuum-regime test cases are studied. This is because these low-Knudsen-number cases have the stiffest collision operator and have been traditionally the hardest to solve numerically.

5.1 Specification of Initial Conditions

Before a calculation, initial conditions for the cell average, $\overline{\boldsymbol{U}}$, and slopes, $\overline{\partial \boldsymbol{U}}_i$, in each cell must be specified. Given a function, $U_0(x_i)$, these initial cell values are obtained from

$$\overline{\boldsymbol{U}_0} = \frac{1}{\mathrm{A_k}} \int\limits_{A_k} U_0\,(x, y)\,\mathrm{d}A, \tag{5.1}$$

$$\overline{\frac{\partial \boldsymbol{U}_0}{\partial x}} = \frac{\int\limits_{A_k} U_0\,(x, y)\ \ (x - x_\kappa)\,\mathrm{d}A}{\int\limits_{A_k} \ (x - x_\kappa)^2\,\mathrm{d}A}, \tag{5.2}$$

$$\overline{\frac{\partial \boldsymbol{U}_0}{\partial y}} = \frac{\int\limits_{A_k} U_0\,(x, y)\ \ (y - y_\kappa)\,\mathrm{d}A}{\int\limits_{A_k} \ (y - y_\kappa)^2\,\mathrm{d}A}, \tag{5.3}$$

where (x_κ, y_κ) is the cell centroid and A_k is the cell area. The cell-averaged value and average cell-slopes of the initial condition in each cell are obtained with sufficient accuracy using three-point Gaussian quadrature.

5.2 Linear Convection-Relaxation

As a demonstration of the scheme's order of accuracy, a simple convection-relaxation equation is first solved. The PDE of interest is

$$\frac{\partial \rho}{\partial t} + v_i \frac{\partial \rho}{\partial x_i} = -\frac{1}{\tau}\rho\,. \tag{5.4}$$

Here, the convection velocity, v_i, is chosen to be $-1\,\mathrm{m/s}$ in both the x and y directions, while the relaxation time is $\tau = 1\,\mathrm{s}$. The domain is a square with $-10\,\mathrm{m} < x < 10\,\mathrm{m}$

and $-10\,\mathrm{m} < y < 10\,\mathrm{m}$. The initial condition is given by

$$\rho_0 = e^{-\frac{x^2+y^2}{2}} \,. \tag{5.5}$$

An exact solution is critical to examining the order of accuracy of a method. The exact solution for this problem is

$$\rho = e^{-\frac{t}{\tau}} e^{-\frac{(x-v_x t)^2+(y-v_y t)^2}{2}} \,. \tag{5.6}$$

A zero-gradient boundary condition is used on all domain limits. Solutions are advanced to a final time of 3 s using a CFL number of 0.3, then the ℓ^2 norm of the error is computed as

$$\text{Error} = \sqrt{\sum_{n=0}^{N} (\rho_{t\,n} - \rho_n)^2 \times A_n} \,. \tag{5.7}$$

Here, A_n is area of the cell, ρ_n is the cell average solution from the scheme, and N is number of cells. No slope-limiter is used in this case. Table 5.1 demonstrates the convergence properties of the scheme for a range of resolutions. Clearly, the scheme is approaching third-order accuracy as the grid is refined. Figure 5.1 graphically depicts the order of accuracy of the scheme. The observed error is almost coincident with a reference line of slope -3. Figure 5.2 shows example solutions for this problem computed at three times on a 200×200 mesh. Note that, the maximum stable CFL number for this case was 0.664.

Table 5.1: Solution errors for convection-relaxation study

Number of Cells	ℓ^2 **error**	**Order**
50×50	1.2425×10^{-3}	—
100×100	1.8389×10^{-4}	2.76
200×200	2.4139×10^{-5}	2.93
400×400	3.0597×10^{-6}	2.98
800×800	3.8403×10^{-7}	2.99

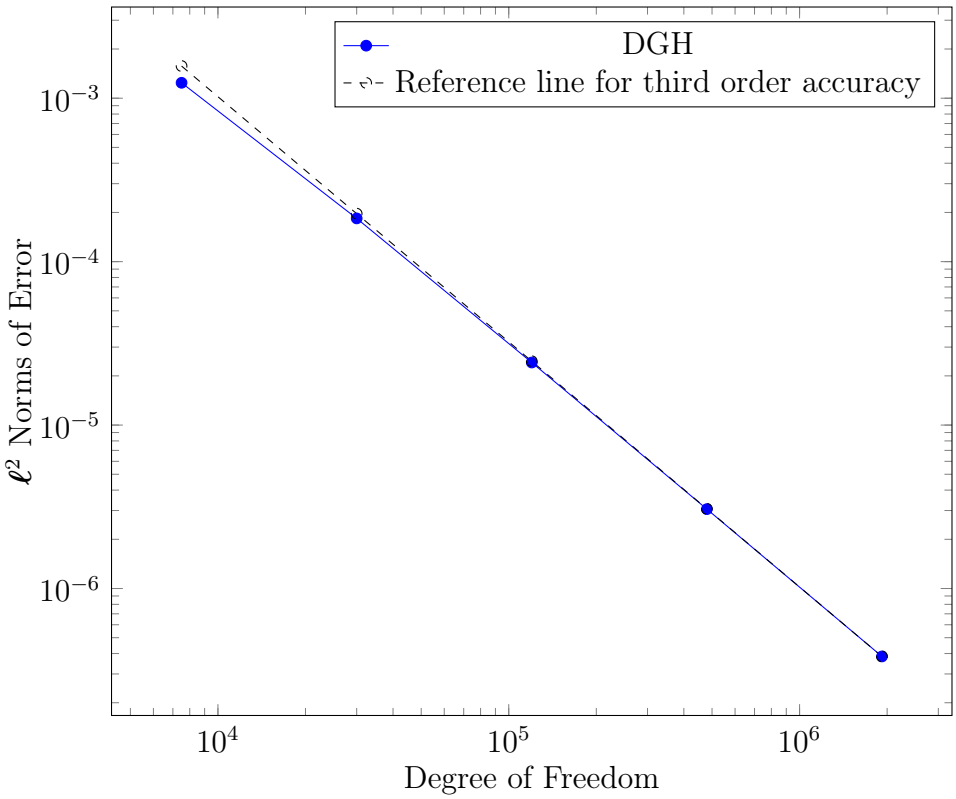
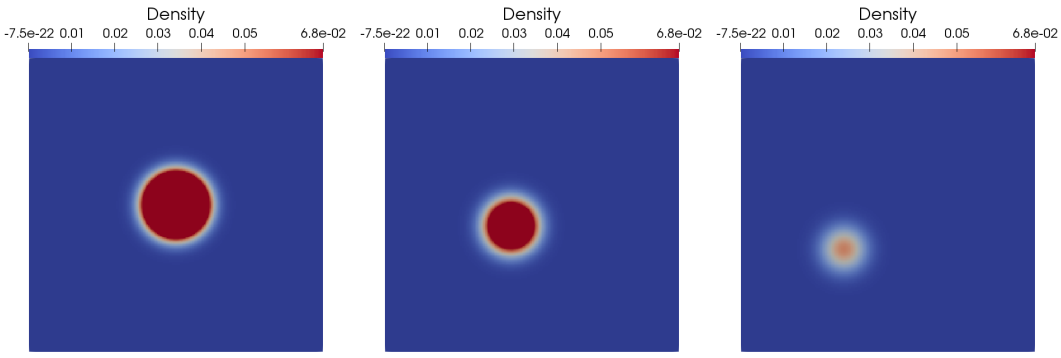


Figure 5.1: ℓ^2 *norms of error, showing the high-order accuracy of the DGH method.*



(a) *Initial condition,* $t = 0\,\mathrm{s}$ (b) $t = 1.5\,\mathrm{s}$ (c) *Final time,* $t = 3\,\mathrm{s}$

Figure 5.2: *Solutions computed for the convection-relaxation problem computed using a* 200×200 *mesh.*

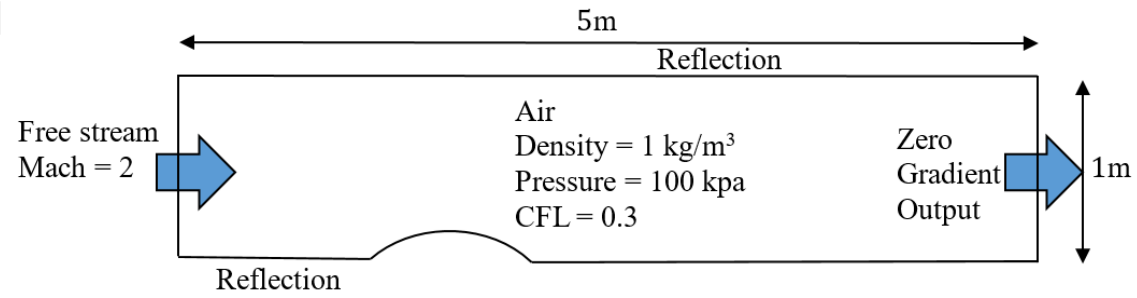


Figure 5.3: *Bump-channel flow problem.*

5.3 Supersonic Inviscid Flow

The next case considered is that of a supersonic inviscid flow through a channel with a bump. The channel is 1 m tall and 5 m long (from $x = -1\,\mathrm{m}$ to $4\,\mathrm{m}$). From $x = 0\,\mathrm{m}$ to $x = 1\,\mathrm{m}$, a section of a circular arc extends 10% into the channel. The top and bottom boundaries are reflections. The right boundary is a zero-gradient condition, while the left boundary is held fixed. A graphic of the problem is shown in Figure 5.3.

The compressible Euler equations, used for this calculation, are found by using the Maxwell-Boltzmann distribution function, since the gas is assumed to be in the

equilibrium state. The resulting PDEs are

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_k}\left(\rho u_k\right) = 0\,, \tag{5.8}$$

$$\frac{\partial}{\partial t}\left(\rho u_i\right) + \frac{\partial}{\partial x_k}\left(\rho u_i u_k + \delta_{ij} p\right) = 0\,, \tag{5.9}$$

$$\frac{\partial}{\partial t}\left(\frac{\rho u_i u_i}{2} + \frac{p}{\gamma - 1}\right) + \frac{\partial}{\partial x_k}\left(u_k\left(\frac{\rho u_i u_i}{2} + \frac{\gamma p}{\gamma - 1}\right)\right) = 0\,. \tag{5.10}$$

In this set of PDEs, Eq. (5.8) is the continuity equation, Eq. (5.9) is a vector momentum equations and Eq. (5.10) is the energy equation. Here, the flux and solution vector in two-dimensions are given by

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u_x \\ \rho u_y \\ \frac{p}{\gamma-1} + \frac{1}{2}\rho\left(u_x^2 + u_y^2\right) \end{bmatrix} \quad \boldsymbol{F}_x = \begin{bmatrix} \rho u_x \\ \rho u_x^2 + p \\ \rho u_x u_y \\ u_x\left(\frac{\gamma p}{\gamma-1} + \frac{1}{2}\rho\left(u_x^2 + u_y^2\right)\right) \end{bmatrix} \tag{5.11}$$

$$\boldsymbol{F}_y = \begin{bmatrix} \rho u_y \\ \rho u_y u_x \\ \rho u_y^2 + p \\ u_y\left(\frac{\gamma p}{\gamma-1} + \frac{1}{2}\rho\left(u_x^2 + u_y^2\right)\right) \end{bmatrix}\,. \tag{5.12}$$

The Euler equations are the lowest-order member of the maximum-entropy moment hierarchy. They describe a gas that is everywhere in local thermodynamic equilibrium, which is the maximum entropy state subject to constraints on the local mass, momentum, and energy densities.

The gas used in this problem is air with $\rho = 1\,\mathrm{kg/m^3}$, $p = 100\,\mathrm{kPa}$, $\gamma = 1.4$, and an incoming Mach number of 2. For this case, the Venkatakrishnan slope limiter is used to stabilize the discontinuous solution [36]. Again, a CFL number of 0.3 is used.

For this case, the solution is first computed on a mesh of five blocks, each with 6×6 cells. Once steady state is achieved, the mesh is refined and a new steady state is

computed. This is repeated four times. The criterion used to flag blocks for refinement is the divergence of the velocity field, which for this case if this value is more than 60000, the block will be refined. This is chosen as it is a good detector of shock waves. Figure 5.4 shows the series of solutions that are obtained. Clearly, the initial mesh of 180 cells does a poor job of capturing the details of the solution. However, as the mesh is successively refined, the discontinuities are efficiently captured. The final mesh, after four rounds of mesh refinement, has 572 blocks and 20 592 cells. If the entire mesh was at the same resolution as the most-refined blocks of this mesh, it would contain more than double the number of cells used here.

The flexibility of the connectivity of the blocks is also clear to see in Figure 5.4. For example, the left-most block is upstream of the initial shock and therefore contains a constant solution state. It is never flagged for refinement. It is neighboured on the right by nine different blocks at three different refinement levels. Smaller blocks can also straddle the cell boundaries of the larger block.

It should be noted that the choice of refinement criterion for this study is somewhat arbitrary. Surely the overall accuracy of the solution is not maximized by this choice. However, the goal is simply to demonstrate the flexibility of the mesh refinement technique and the robustness of the numerics to highly discontinuous solutions and mesh resolutions.

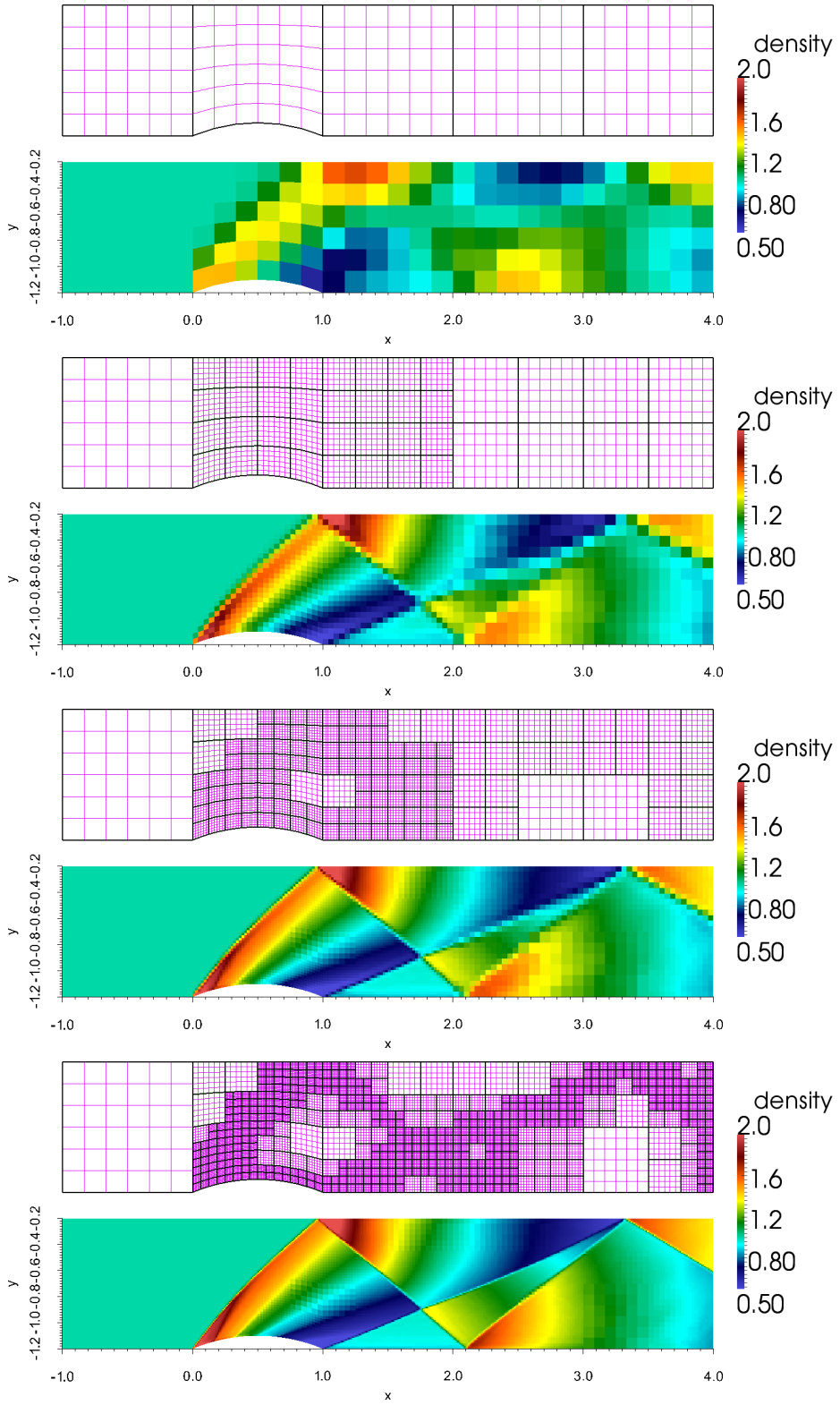


Figure 5.4: *Bump-channel flow with a Mach number of 2 computed using the compressible Euler equations using zero, two, three, and four levels of mesh refinement.*

5.4 Viscous Mixing Layer

This case is a low-speed viscous mixing layer. This situation demonstrates the ability of a first-order moment method to accurately capture a classical viscous result. The moment equations corresponding to the Gaussian closure can be obtained by using the weights $\boldsymbol{M} = [1, v_i, v_i v_j]$ in the entropy maximization process described in 2.5.1. The resulting distribution function, denoted by $\mathcal{G}$ for the Gaussian model, takes the form [11]

$$\mathcal{F} = \mathcal{G} = \frac{\rho}{m(2\pi)^{\frac{3}{2}}(\det \Theta_{ij})^{\frac{1}{2}}} \exp^{(-\frac{1}{2}\Theta_{ij}^{-1} c_i c_j)}, \tag{5.13}$$

where $\Theta_{ij} = \frac{P_{ij}}{\rho}$ is an anisotropic "temperature" tensor. This non-equilibrium distribution possesses a Gaussian-like distribution in each of the principal strain axes. Physically, it corresponds to a non-equilibrium condition with a different temperature in each direction.

For this demonstration, the Gaussian ten-moment model is used. This moment closure has ten equations that predict the evolution of mass, momentum, and the anisotropic pressure tensor, P_{ij}, which is related to the fluid stresses by Eq. (2.14). The PDEs that describe the model can be written as

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_k}(\rho u_k) = 0\,, \tag{5.14}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_k}(\rho u_i u_k + P_{ik}) = 0\,, \tag{5.15}$$

$$\frac{\partial}{\partial t}(\rho u_i u_j + P_{ij}) + \frac{\partial}{\partial x_k}(\rho u_i u_j u_k + u_i P_{jk} + u_j P_{ik} + u_k P_{ij}) = -\frac{1}{3\tau}(3P_{ij} - \delta_{ij} P_{kk})\,. \tag{5.16}$$

The flux, solution and source-term vector for the 10-moment model in two-dimensions are given by

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u_x \\ \rho u_y \\ \rho u_x^2 + P_{xx} \\ \rho u_x u_y + P_{xy} \\ \rho u_y^2 + P_{yy} \\ \rho P_{zz} \end{bmatrix}, \; \boldsymbol{F}_x = \begin{bmatrix} \rho u_x \\ \rho u_x^2 + P_{xx} \\ \rho u_x u_y + P_{xy} \\ \rho u_x^3 + 3 u_x P_{xx} \\ \rho u_x^2 u_y + 2 u_x P_{xy} + u_y P_{xx} \\ \rho u_x u_y^2 + u_x P_{yy} + 2 u_y P_{xy} \\ \rho u_x P_{zz} \end{bmatrix}, \tag{5.17}$$

$$\boldsymbol{F}_y = \begin{bmatrix} \rho u_y \\ \rho u_x u_y + P_{xy} \\ \rho u_y^2 + P_{yy} \\ \rho u_x^2 u_y + 2 u_x P_{xy} + u_y P_{xx} \\ \rho u_y^2 u_x + u_x P_{yy} + 2 u_y P_{xy} \\ \rho u_y^3 + 3 u_y P_{yy} \\ \rho u_y P_{zz} \end{bmatrix}, \tag{5.18}$$

$$\boldsymbol{S} = \frac{1}{\tau} \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{(2P_{xx} - P_{yy} - P_{zz})}{3} \\ P_{xy} \\ \frac{(2P_{yy} - P_{xx} - P_{zz})}{3} \\ \frac{(2P_{zz} - P_{xx} - P_{yy})}{3} \end{bmatrix}. \tag{5.19}$$

Here, the collision integral on the right-hand side has been replaced by the BGK collision operator [12]. This simplified operator predicts non-equilibrium effects to relax away on a time scale, τ. The relaxation time can be chosen such that, in the continuum regime, the model recovers the viscosity of a Newtonian gas, μ. This is done by defining $\mu = \tau p$.

The particular mixing layer studied here is made up of two layers of Argon with standard atmospheric temperature and pressure. The viscosity of Argon is taken as $\mu = 2.117 \times 10^{-5}\,\mathrm{Pa\,s}$. Initially, the line $y = 0\,\mathrm{m}$ separates a stream flowing rightward

at $\hat{u} = 1\,\mathrm{m/s}$ from a stream flowing leftward at the same speed. The traditional incompressible Navier-Stokes solution for this situation can be computed analytically and is given by

$$u_x = \hat{u}\ \ \mathrm{erf}\left(\frac{y}{2}\sqrt{\frac{\rho}{\mu t}}\right) . \tag{5.20}$$

The goal is to recover this same solution using the numerical scheme applied to the Gaussian equations.

For this calculation, the domain in the y direction is taken to be $-0.012\,\mathrm{m} < y < 0.012\,\mathrm{m}$. As this is a one-dimensional calculation, only two cells are used in the x direction along with periodic boundary conditions. A zero-gradient boundary condition is used in the y direction. The Venkatakrishnan slope limiter is again used along with a CFL number of 0.4. The final time of the calculation is $t = 0.5\,\mathrm{s}$.

A zoom-in on the results in the area around the mixing layer is shown in Figure 5.5. Solutions are computed using 10, 20, 40, and 80 cells in the y direction. As can clearly be seen, in all cases the solution of the first-order hyperbolic PDEs agree extremely well with the traditional Navier-Stokes solution. It seems that having a half-dozen cells in the mixing layer is all that is necessary to have near perfect agreement.

The low errors observed in this study are especially impressive given the very low speed of the flow. The Mach number of this flow is less than 0.01, yet the excessive diffusion that would normally result from a compressible solver is not apparent. This is not a result of the use of a moment-method, but rather a result of the extremely low dissipation of the discontinuous-Galerkin-Hancock scheme.

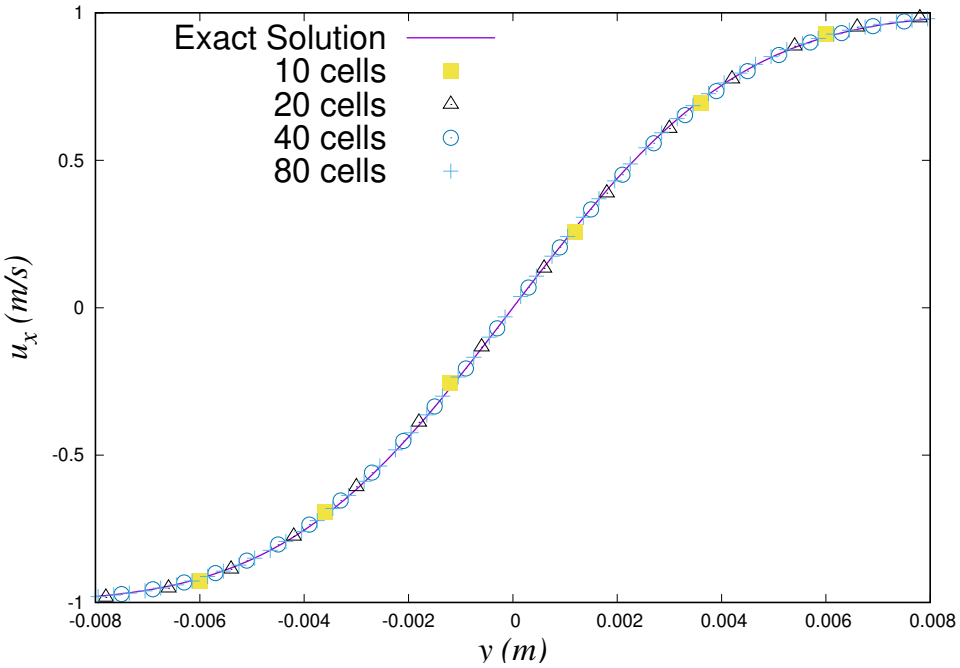


Figure 5.5: *Mixing-layer prediction using the Gaussian 10-moment closure as compared to the exact solution for incompressible Navier-Stokes.*

5.5 Stokes Flow Over Cylinder

This case is a low-Reynolds-number Stokes flow past a circular cylinder. The goal of this case is to again demonstrate the ability of the first-order method to accurately capture a classical viscous result. The fluid is Argon with a density of $\rho = 1.784\,\mathrm{kg/m^3}$, free-stream velocity and pressure of $0.5\,\mathrm{m/s}$ and $101\,325\,\mathrm{Pa}$ respectively, and constant viscosity of $\mu = 2.117 \times 10^{-5}\,\mathrm{Pa\,s}$. The cylinder diameter is $1 \times 10^{-5}\,\mathrm{m}$. This corresponds to a Reynolds number of 0.421 and Knudsen number of 0.0063. The Mach number in this case is 0.0018. A schematic of this problem is shown in Figure 5.6.

An exact solution is available for this situation using the traditional Navier-Stokes

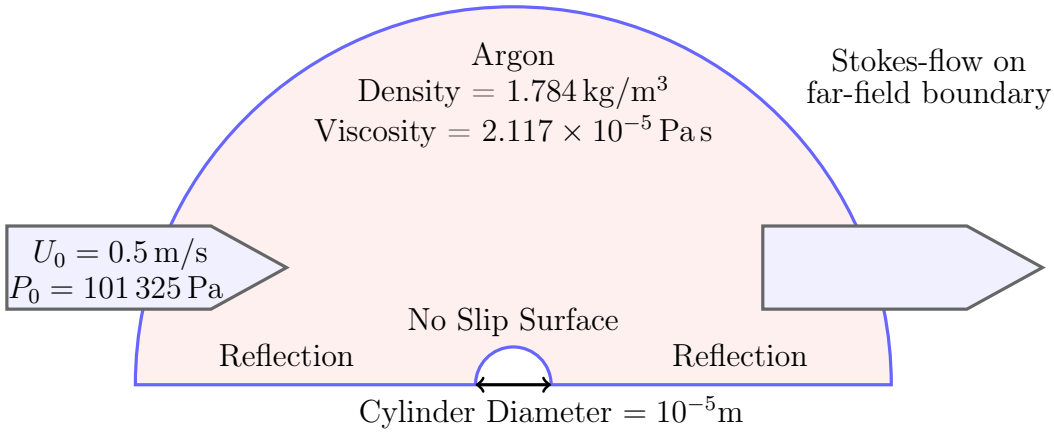


Figure 5.6: *Stokes flow around a circular cylinder.*

equations [37]. It is given in terms of a two-dimensional stream function, ψ,

$$\psi = \sin\theta \left(Ar^3 + Br \ln r + Cr + Dr^{-1} \right). \tag{5.21}$$

Here, A, B, C, and D are integration constants that are used to specify boundary conditions. For this case, no-slip is assumed at the cylinder wall and the velocity is assumed to be free-stream on a circle some given distance from the immersed body. The velocity in polar coordinates can be found as

$$v_\theta = -\frac{\partial \psi}{\partial r}, \quad v_r = \frac{1}{r}\frac{\partial \psi}{\partial \theta}. \tag{5.22}$$

Once the velocity field is known, the pressure and viscous-stress fields can also be determined from the Navier-Stokes equations. As this problem is symmetric, only half the domain is used for the calculation.

For this computation, a mesh of 22 blocks and 22 880 cell is used, as shown in Figure 5.8. A zoom-in of the area surrounding the cylinder is shown in Figure 5.8(b). The boundary condition at the cylinder is no-slip while the symmetry line is modelled as a reflection. On the far-field boundary, the exact solution for Stokes flow is imposed.

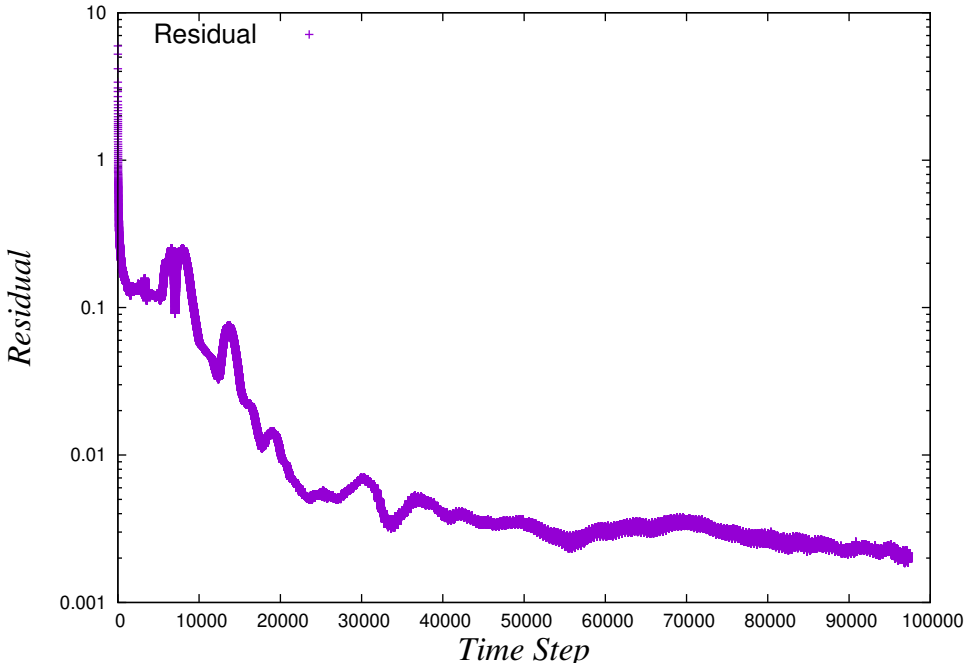
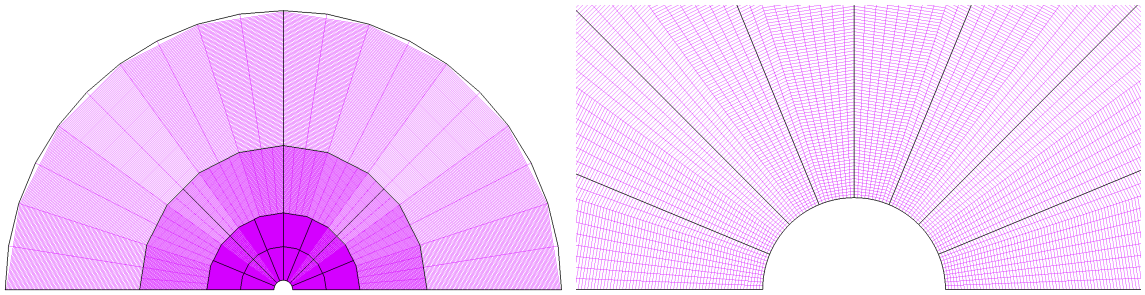


Figure 5.7: *Residual for the 10-moment model for Stokes flow past a cylinder.*

Figure 5.7 shows the residual for this case. The residual is reduced by more than three orders of magnitude. Much longer calculations have been run, but the residual stalls at this level. Even after much longer simulations, solutions do not change perceptibly.

Figure 5.9 compares the DGH moment-closure solutions with the exact solution. The top of each figure is the moment method solution, while the bottom is the reflection of the exact solution. Figure 5.9(a) compares the prediction for the shear pressure, P_{xy}, with the exact solution. Again, this shear pressure is the negative of the viscous shear stress from the Navier-Stokes equations, τ_{xy}. The breaks in the contour lines are the result of the plotting software being unable to interpolate across block boundaries. One can observe that the two halves are almost identical. The moment method in combination with the DGH scheme is clearly able to predict this classical viscous result. In Figure 5.9(b), the predicted Mach number and streamlines are



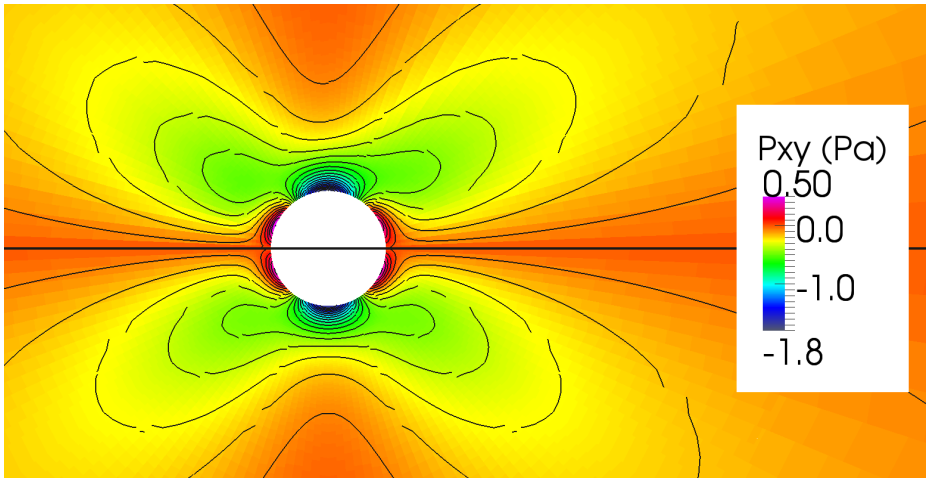
(a) *Semi-cylinder grid* (b) *Zoom-in view*

Figure 5.8: *Semi-cylindrical grid used for Stokes-flow calculations. The grid comprises twenty-two blocks at three different refinement levels and* 22 880 c*ells total.*

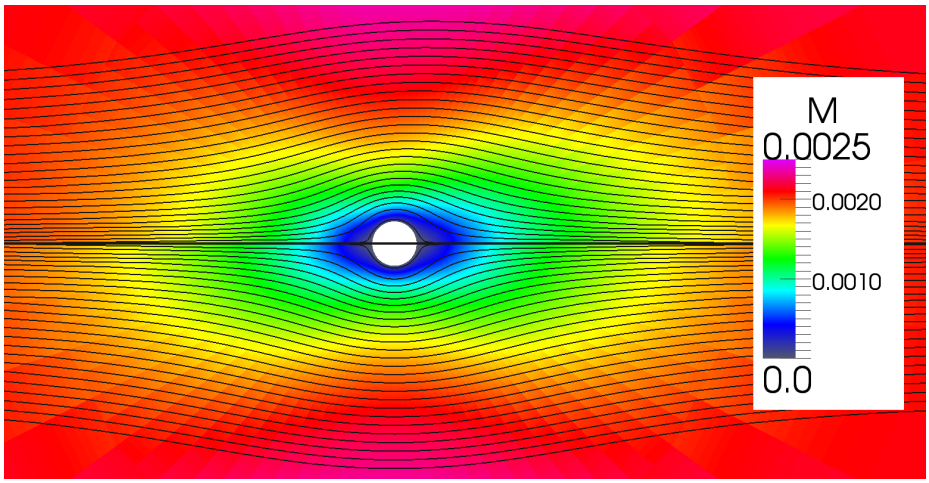
compared with the exact result. Once again agreement is very good. The low-speed region in the moment-closure prediction seems to shift downstream slightly, however this may be due to numerical dissipation in this extremely low-speed problem. Finally, the deviation of P_{xx} from the thermodynamic pressure p is shown in Figure 5.9(c). This value corresponds to the negative of the normal component of the viscous stress in the traditional Navier-Stokes model. Again agreement is very strong. The moment method seems to over-predict the magnitude of the stress in front of the cylinder while slightly under predicting its magnitude behind. Again, this is likely numerical error caused by the extreme low-speed flow leading to a flow problem with very high condition number.

5.6 Parallel Efficiency

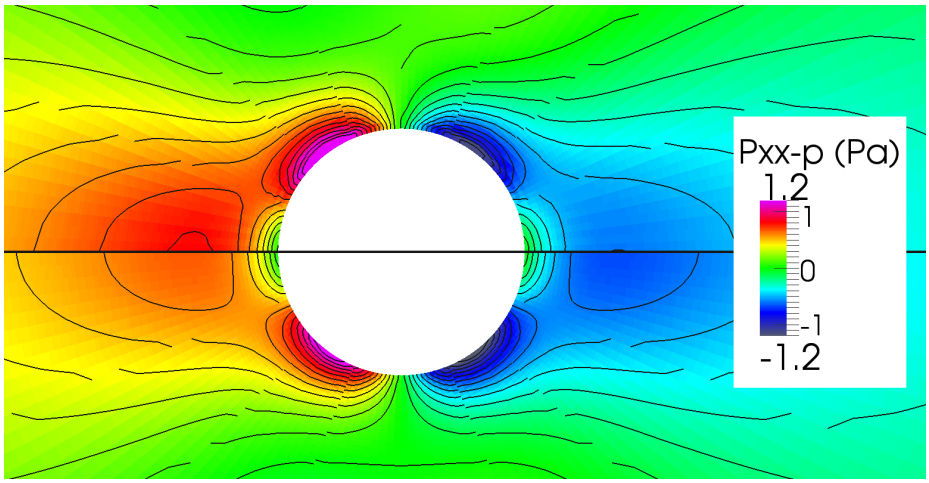
The Discontinuous Galerkin-Hancock method has been implemented within a parallel block-based AMR code. The computations for this study were carried out on a homogeneous cluster, Niagara, owned by the University of Toronto and operated by SciNet



(a) *Shear pressure,* P_{xy}



(b) *Mach number and streamlines*



(c) *Deviatoric* x*-direction pressure,* $P_{xx} - p$

Figure 5.9: *Stokes flow around a circular cylinder. Top half is the ten-moment solution while the bottom half is the reflection of the exact Navier-Stokes solution.*

and Compute Canada. The facilities includes 1500 nodes, each with 40 Intel Skylake Xenon Gold 5115 hyperthreaded cores with a clock speed of 2.4GHz. This gives a total of 60000 cores. Each node has 202GB of RAM. The present implementation was written using the C++ programming language, OpenMP, and the MPI (message passing interface) library.

An important aspect of the DGH scheme is its high parallel efficiency. Calculations of the parallel performance and scalability of the algorithm are shown in Fig 5.10. The parallel speed-up, S_n, is defined as

$$S_n = \frac{t_1}{t_n}, \tag{5.23}$$

where t_1 is the wall time the computation takes on one node and t_n is the wall time the computation takes on n nodes. For perfect stability, S_n should be a line of slope 1. The parallel efficiency, E_n, given by

$$E_n = \frac{S_n}{n}. \tag{5.24}$$

Again, for perfect scaling, the efficiency would be 100%. Obviously this is unattainable, as inter-CPU communication reduces the efficiency. The problem considered is the viscous mixing layer (Sec 5.4), but now with 2560 blocks of 50×30 cells, giving a $3,840,000$-cell computational mesh. The efficiency, E_n, is shown as a function of the number of nodes. The parallel efficiency remains as high as 91.4% for 640 cores. This result is high, since the scheme passes only one message per time step. Table 5.2 shows the wall clock time to run the case on different numbers of cores. Note that, in this study one MPI process is used for each available CPU core. As the cores are hyperthreaded, two OpenMP threads are used in each MPI process.

Table 5.2: Wall time for running on different number of nodes for the mixing-layer case (Sec 5.4), with 2560 blocks of 50×30 elements.

Number of Nodes	**Number of Cores**	**Wall Time** (s)
1	40	18399
2	80	9803
4	160	4881
8	320	2477
16	640	1258

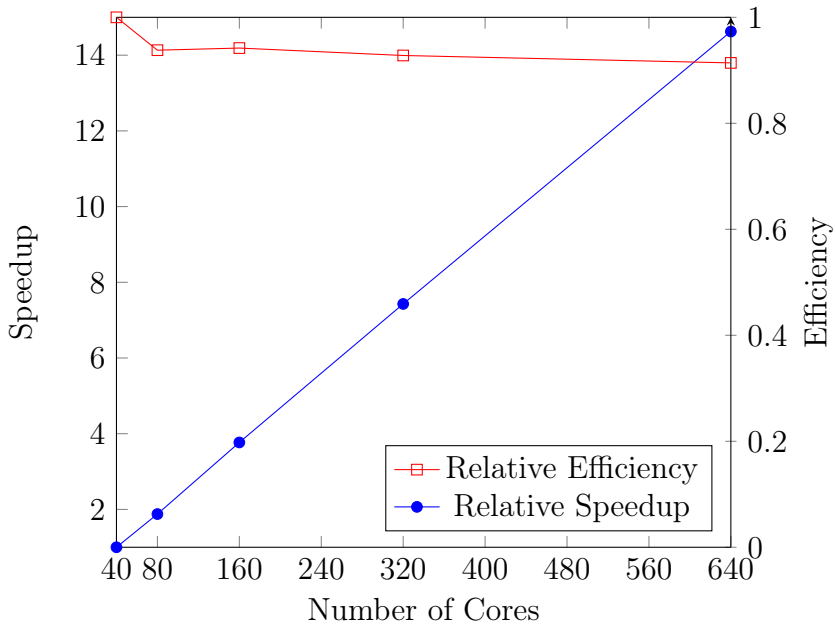


Figure 5.10: *Parallel efficiency for the mixing-layer problem (Sec 5.4) comprising* 2560 *blocks of* 50×30 *elements.*

Chapter 6

Conclusion

6.1 Summary

In this thesis, a moment method approach to computational fluid dynamics is described. The advantages of first order PDEs models that are derived from the moment method as compared to the Navier-Stokes equations are explained. With models that are derived from moment closure approach one can accurately solve problems beyond the continuum regime.

The first implementation of a coupled space-time discontinuous-Galerkin-Hancock method is presented. This method is designed specifically for the efficient solution of hyperbolic-relaxation equations that result from moment closures of kinetic theory. It may be surprising to some that viscous flow behaviour can be accurately predicted by first-order PDEs. However, the applicability of hyperbolic moment methods to both continuum and non-equilibrium gas flows is now well established. Such a first-order treatment brings many physical and computational advantages to gas flow prediction. The efficiency of the scheme also has been examined.

In order to do the adaptive mesh refinement (AMR), boundary and block signatures were defined. Boundary signatures are object data types, that exist on each boundary edges with individual signature, that not only make it possible to connect

boundaries from different blocks, but can also connect boundary edges to two blocks. These signatures are used to coordinate communication through a comm hub such that the number of messages sent remains at the minimum.

It was verified that this scheme is third-order accurate in space and time. The scheme also has high efficient in parallel calculations as it requires only one message be passed during each time step. In each time step, local data in each cell was used as a predictor for inputs to Riemann fluxes. There are three degrees of freedom in each cell: the average of the solution value and the average solution derivatives. Furthermore, since there is a stiff source term, a point implicit method was chosen. The scheme was shown to be stable and accurate.

The scheme's stability to model strong shocks near curved boundaries was also verified. Solutions remain stable when the mesh is refined locally. Block connectivity and refinement for this implementation were explained. Numerical solutions of 10-moment equations with the DGH method were presented and compared with numerical results derived from Navier Stokes equations. Results for Stokes flow using the 10-moment equations on a locally refined mesh were found and compared with exact solutions from the Navier Stokes equations. Finally, it was shown that the scheme has high efficiency in parallel remaining above 90% for 640 cores.

6.2 Suggestion for Future Works

There are great number of issues that need to be explored in the moment method and DGH schemes. Immediate avenues for future progress are

- The 10-moment equations, used in this thesis are a robust set of moment equations. However, higher moment models are available. Using models beyond the 10-moment equation, with the DGH scheme should be explained.

- The accuracy of the scheme on unstructured meshes remains to be studied.

- The extension of the two-dimensional Discontinuous-Galerkin-Hancock method to three-dimensional grids should be pursued.

- The application of the scheme to more complex, real-world gas and multiphase flow problems should be investigated.

References

[1] J. G. McDonald, J. S. Sachdev, and C. P. T. Groth. Application of Gaussian moment closure to micro-scale flows with moving and embedded boundaries. *American Institute of Aeronautics and Astronautics Journal*, 52:1839–1857, 2014.

[2] G. A. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*. Clarendon Press, Oxford, 1994.

[3] L. Mieussens. Discrete velocity model and implicit scheme for the BGK equation of rarefied gas dynamics. *Mathematical Models and Methods in Applied Sciences*, 10(8):1121–1149, 2000.

[4] H. T. Huynh. An upwind moment scheme for conservation laws. *Proceeding of the Third International Conference on Computational Fluid Dynamics*, pages 761–766, 2006.

[5] Y. Suzuki and B. van Leer. A discontinuous Galerkin method with Hancock-type time integration for hyperbolic systems with stiff relaxation source terms. In H. Deconinck and E. Dick, editors, *ICCFD4, Ghent, Belgium, July 10–14, 2006*, pages 59–64, Heidelberg, 2006. Springer-Verlag.

[6] Y. Suzuki. *Discontinuous Galerkin Methods for Extended Hydrodynamics*. PhD thesis, University of Michigan, 2008.

[7] G. D. van Albada, B. van Leer, and W. W. Roberts, Jr. A comparative study of computational methods in cosmic gas dynamics. *Astronomy and Astrophysics*, 108(1):76–84, 1982.

[8] H. Struchtrup. *Macroscopic Transport Equations for Rarefied Gas Flows*. Springer-Verlag, Berlin, 2005.

[9] S. Chapman and T. G. Cowling. *The Mathematical Theory of Non-Uniform Gases*. Cambridge University Press, Cambridge, 1970.

[10] James Clerk Maxwell. Illustrations of the dynamical theory of gases. *Philosophical Magazine*, 91:19–32, 1860.

[11] T. I. Gombosi. *Gaskinetic Theory*. Cambridge University Press, Cambridge, 1994.

[12] P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems. *Physical Review*, 94(3):511–525, 1954.

[13] H. Grad. On the kinetic theory of rarefied gases. *Communications on Pure and Applied Mathematics*, 2:331–407, 1949.

[14] W. Dreyer. Maximization of the entropy in non-equilibrium. *Journal of Physics A: Mathematical and General*, 20:6505–6517, 1987.

[15] I. Müller and T. Ruggeri. *Extended Thermodynamics*. Springer-Verlag, New York, 1993.

[16] C. D. Levermore. Moment closure hierarchies for kinetic theories. *Journal of Statistical Physics*, 83:1021–1065, 1996.

[17] M. Junk. Domain of definition of Levermore's five-moment system. *Journal of Statistical Physics*, 93(5/6):1143–1167, 1998.

[18] J. G. McDonald and C. P. T. Groth. Towards realizable hyperbolic moment closures for viscous heat-conducting gas flows based on a maximum-entropy distribution. *Continuum Mechanic Thermodynamics*, 25:573–603, 2013.

[19] J. G. McDonald and M. Torrilhon. Affordable robust moment closures for CFD based on the maximum-entropy hierarchy. *Journal of Computational Physics*, 251:500–523, 2013.

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

[21] P. LeSaint and P.A. Raviart. *on a finite element method for solving the neutron transport equation.* Academic Press, 1974.

[22] C. Johnson and J. Pitkaränta. An aanalysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Mathematics of Computation*, (46):1–26, 1986.

[23] T. Peterson. A note on the convergence of the discontinuous Galerkin method for a scalar hyperbolic equation. *Society for Industrial and Applied Mathematics*, (28):133–140, 1991.

[24] G.R. Richter. An optimal-order error estimate for the discontinuous Galerkin method. *Mathematics of Computation*, (50):75–88, 1988.

[25] G. Chavent and G. Salzano. A finite element method for the 1d water flooding problem with gravity. *Journal of Computational Physics*, (45):307–344, 1982.

[26] G. Chavent and B. Cockburn. The local projection p0 p1-discontinuous-Galerkin finite element method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, (23):23–565, 1989.

[27] B. van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second order scheme. *Journal of Computational Physics*, 14:361–370, 1974.

[28] B. Cockburn and C.W. Shu. The Runge-Kutta local projection p1-discontinuous Galerkin method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, (25):337–361, 1991.

[29] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.

[30] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes II. *Journal of Computational Physics*, 83:32–78, 1989.

[31] A. Harten, P. D. Lax, and B. van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *Society for Industrial and Applied Mathematics Review*, 25(1):35–61, 1983.

[32] B. Einfeldt. On Godunov-type methods for gas dynamics. *Society for Industrial and Applied Mathematics Numerical Analysis*, 25:294–318, 1988.

[33] B. Cockburn, S. Hou, and C.-W. Shu. TVB Runge-Kutta local projection discontinous Galerkin finite-element method for conservation laws IV: The multidimensional case. *Journal of Computational Physics*, 54:545, 1990.

[34] J. S. Sachdev, C. P. T. Groth, and J. J. Gottlieb. A parallel solution-adaptive scheme for predicting multi-phase core flows in solid propellant rocket motors. *International Journal of Computational Fluid Dynamics*, 19(2):159–177, 2005.

[35] X. Gao and C. P. T. Groth. A parallel adaptive mesh refinement algorithm for predicting turbulent non-premixed combusting flows. *International Journal of Computational Fluid Dynamics*, 20(5):349–357, 2006.

[36] V. Venkatakrishnan. On the accuracy of limiters and convergence to steady state solutions. Paper 93-0880, American Institute of Aeronautics and Astronautics, January 1993.

[37] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, 2000.