Modeling collisional processes in plasmas using discontinuous numerical methods

Sean Miller

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

Doctor of Philosophy

University of Washington

2016

Reading Committee:

Uri Shumlak, Chair

Richard Milroy

Setthivoine You

Program Authorized to Offer Degree:
UW Aeronautical & Astronautical Engineering

University of Washington

Abstract

Modeling collisional processes in plasmas using discontinuous numerical methods

Sean Miller

Chair of the Supervisory Committee:
Professor Uri Shumlak
Aeronautical & Astronautical Engineering

Fluid-based plasma models are typically applied to parameter regimes where a local thermal equilibrium is assumed. The applicability of this regime is valid for many plasmas, however, it is limited to plasma dynamics dominated by collisional effects. This study attempts to extend the validity of the collisional fluid regime using an anisotropic 13-moment fluid model derived from the Pearson type-IV probability distribution. The model explicitly evolves the heat flux hyperbolically alongside the density, momentum, and energy in order to capture dynamics usually restricted to costly kinetic models. Each particle species is modeled individually and collectively coupled through electromagnetic and collision operators. To remove electromagnetic divergence errors inherent to numerical representations of Maxwell's equations, both hyperbolic and parabolic cleaning methods are presented. The plasma models are implemented using high-order finite volume and discontinuous Galerkin numerical methods designed for unstructured meshes. The unstructured code framework, numerical methods, and plasma models were developed in the University of Washington's WARPXM code for use on heterogeneous accelerated clusters.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Uri Shumlak, for his support and guidance over the course of my graduate career. I would also like to thank the entirety of the computational plasma dynamics group at the University of Washington including Bob Lilly, Noah Reddell, Eder Sousa, Peter Norgaard, Genia Vogman, Andrew Ho, Iman Datta, Sina Taheri, and Whitney Thomas for the near endless discussions of physics, philosophy, and politics that made my graduate school experience both educational and fun. Special thanks to Susan Griffith, whose motivating conversations about life beyond graduate school always kept me on track. Lastly, I would like to thank the Aeronautics & Astronautics department of the University of Washington for their support and resources, both educational and computational, that allowed for this research.

Chapter 1

INTRODUCTION

Plasma physics is the study of collective behavior of charged particles. Plasmas can range from weakly-ionized plasmas are dominated by neutral particle interactions[2;3], which has various applications in the aerospace community[4–6], to the high energy density plasmas used in fusion energy research[7–9]. This study focuses on the weakly-coupled plasma regime making up most laboratory plasmas. More specifically, this study deals with the multi-species plasma model requirements for resolving electron dynamics and moderate-collisionality.

Modeling moderately collisional plasmas is a field dominated by computationally expensive kinetic solvers, and the goal of this study is to develop a robust plasma model, as well as a computational framework, for capturing moderately collisional plasma dynamics in complex geometries inexpensively and in high-fidelity. The objectives and contributions of this research are separated into three categories for the plasma modeling, numerical methods, and computational framework.

1.1 Collisional transport in multi-species plasma models

Collisionality in plasmas is defined by the electromagnetic interactions between particles, which, over time, tend to drive a plasma toward a local thermal equilibrium[10]. Over longer time scales this local thermal equilibration extends to the global plasma. The rate at which the system reaches a local thermal equilibrium, known as the thermalization rate, is directly related to the frequency of interactions, or collisions, between particles.

In the case of uncharged fluids, the interparticle interactions are roughly approximated by

hard sphere collisions. In the case of plasmas, each particle is electromagnetically coupled to every other particle in the system. These electromagnetic effects are subdivided into long range effects governed by interactions with an electromagnetic field, and short range effects that happen below the scale of the plasma Debye length. The short-range interactions are classically approximated by Coulomb scattering collisions which describe the deviation in a particle's trajectory due to a conservative force field.

Collisional transport is the collective result of collisions both within a particle species (intraspecies) and between species (interspecies). For low-temperature or high-density plasmas, collisional transport is an important driver of particle dynamics, especially in the presence of magnetic fields[11]. A collisionless plasma contains no collisional transport such that charged particles are governed purely by long range electromagnetic interactions and advective flow. The magnitude of collisional transport in a plasma determines the relevant model used to simulate dynamics.

Low collisionality implies that particles are approach thermal equilibrium over a longer time scale and are best described by kinetic models in which the position and velocity of each particle is evolved in time. N-body plasma models (e.g. molecular dynamics) are prohibitively expensive due to the number of individual particles in a given plasma. Averaging over the discrete particles creates a six dimensional phase space distribution[10] which describes the number of particles in a given volume of space with a particular velocity range. Continuum kinetic models based on the Boltzmann equation, and its collisionless form, the Vlasov equation, are popular models for evolving the phase space distribution, but are computationally expensive due to the difficulty of resolving up to six dimensions.

High collisionality implies that particles are in, or near, local thermal equilibrium. This is a state where a species's phase space distribution is well represented by an isotropic Gaussian (Maxwellian) in velocity space as indicated by Boltzmann's H-theorem[10;12]. The assumption that the velocity space distribution is near some prescribed form, e.g. thermal equilibrium, allows the generation of moment models, where the Boltzmann equation is integrated over

velocity space to reduce the model's dimensionality from six to three. These compact forms greatly reduce the computational expense of the model, however, their validity is limited to regimes where the velocity space distribution remains near its prescribed form.

The moderately collision regime sits between the low and high collisionality regimes. While kinetic models can accurately resolve plasma dynamics in this regime, the goal of this research is to develop a more computationally tractable model capable of capturing non-equilibrium dynamics.

The general approach consists of computing velocity moments of the Boltzmann equation to eliminate the velocity space dimensions from the model. The details of deriving moment models can be found in Sec. 2. The multi-species moment models developed in this study are constructed from sets of moment models, each describing an individual species. Each species model describes advective flow as well as electromagnetic and collisional effects. As this is a multi-species model designed to capture electron dynamics that drive displacement currents, Maxwell's equations are used to describe the electric and magnetic field evolution. The classic form for multi-species plasma models is based on the 5-moment model[11;13–16] where the plasma is assumed to be near thermal equilibrium.

Higher-moment models beyond 5 moments have been used in neutral fluid dynamics[12;17–21] and near-collisionless plasma models[22;23], however, their application to moderately collisional plasmas is largely unexplored. Higher-moment models represent a unique and effective means of extending the validity of 5-moment plasma models outside of thermal equilibrium. The objective of this research is to construct a 13-moment extension of the 5-moment multi-species plasma model designed to accurately capture magnetized collisional transport. The 5-moment model, and its extension to 13 moments is discussed in Chapter 2. Chapter 5 compares results derived from 5-moment and 13-moment models, with the goal of highlighting the strengths and weaknesses of each mode.

1.2 High-order accurate numerical methods

Numerical methods represent the discretization of an analytical model representing physical behavior into an algorithm that can be implemented on a computer. The numerical methods developed in this paper are discontinuous numerical methods and include a high-order finite volume method and a discontinuous Galerkin method. These methods are designed to achieve high-order convergence accuracy for large systems of equations operating in complex geometries, and their discontinuous nature allows for a high level of parallelizability. High-order convergence accuracy is required for multi-species plasmas containing disparate spatial and temporal scales due to the different particle masses and charges. The stability of discontinuous methods in the presence of discontinuous solutions make them ideal for moment-based plasma models, especially higher-moment plasma models, which are constructed from nonlinear systems of equations.

An objective of this research is to develop computationally efficient, highly-parallelizable numerical methods for modeling systems containing dozens, if not hundreds, of coupled equations on complex geometries. This is a requirement for the multi-species 13-moment plasma model, where even a simple three species plasma is described by forty-five equations. The methods developed and implemented for this study are derived in Chapter 3 and tested in Chapter 5.

1.3 Computational framework

The code framework used for this study is the WARPXM framework under developed by the University of Washington's computational plasma dynamics group. WARPXM is an extension of a preexisting WARPX framework which was designed for modeling plasmas on structured grids. Over the course of this study, the framework was extended to incorporate unstructured meshes allowing for more complex plasma geometries designed by computer-aided design (CAD) programs. The move from a structured to unstructured geometry re-

quired additional components to be developed for the framework for calculating geometric quantities, managing and partitioning the mesh, and data visualization. WARPXM is designed to solve large-scale, multidimensional, multi-species plasma models by incorporating multiple levels of task and data parallelism. The framework is constructed to utilize modern heterogeneous accelerated cluster environments that include multiple compute devices such as CPUs and GPUs. The unstructured mesh interface combined with the complex memory restrictions of accelerated architectures necessitated the development of novel methods for storing and synchronizing data. The philosophy and structure of WARPXM is covered in Chapter 4.

Chapter 2

MULTI-SPECIES PLASMA MODELS

One of the main goals of this study is to construct a model capable of accurately capturing moderately-collisional, magnetized plasma transport in multidimensional systems containing multiple species. Multi-species models individually describe each species and couple the species together using a combination of Maxwell's equations and collision operators. This chapter discusses the derivation and applicability of multi-species plasma models with a focus on collisional transport in moment models.

There exist many methods for capturing collisional transport effects with varying levels of success and practicality. The kinetic Boltzmann-Maxwell model is considered to be one of most accurate means of modeling microscopic collisional transport in plasmas[24–26]. However, since this model effectively tracks the position and velocity of each particle using a six dimensional phase space distribution, it is a computationally expensive evaluation. The expense of the Boltzmann-Maxwell model can be reduced using sampling procedures known as particle-in-cell (PIC) methods[27], though they remain computationally expensive to evaluate.

Moment model representations converts the infinite scope of velocity space associated with the Boltzmann model's phase space distribution into a finite number of variables. The most prevalent moment based plasma model is the Braginskii[11] model, which captures three dimensional plasma dynamics using a set of five variables for each species. This model is ideal for modeling collisional transport in highly collisional plasmas near thermal equilibrium, however, additional moments are required as the plasma transitions out of this state[12].

The question of how many moments are necessary to resolve the phenomena of interest is problem dependent, and while including more moments can increase physical accuracy, it also drastically increases the model's scale and complexity. The objective of this research is to develop a moment model that balances robustness with practicality. Robustness refers to physical accuracy and stability, while practicality implies the model is computationally efficient to evaluate in multiple dimensions and can be adapted to work in various collisional regimes.

The plasma model development begins in Sec. 2.1 with a discussion of how to frame plasma model applicability and validity through the use of normalization parameters. Once the normalization scheme is defined, a brief discussion on the derivation of moment models from the fully kinetic formulation is given in Sec. 2.2. Section 2.3 derives the classic multi-species 5-moment plasma model for non-equilibrium plasmas. The 5-moment model, and its limitations, is used as a basis for the development of the 13-moment model in Sec. 2.4. Both the 5-moment and 13-moment models are derived with a consistent set of collision operators to handle short range electrostatic interactions between particles. These collision operators are derived and discussed in Sec. 2.5. Finally, since the kinetic, 5-moment, and 13-moment plasma descriptions rely on electromagnetic interactions, Sec. 2.6 discusses the models used to describe the evolution of the electric and magnetic fields.

2.1 Plasma model normalization and applicability

The validity and applicability of plasma models are best described using parameter regimes. Plasma parameters are values that capture the importance of resolving particular behaviors, such as electron waves or ion viscous boundary layers. The validity of a model applied to a physical system is based on whether the physical system fits within the model's applicable parameter regime. For the models developed in this dissertation, there are four normalization parameters of interest: speed of light, magnetization, charge separation, and collisionality. The following sections outline the purpose of these parameters and their derivation from

fundamental plasma physics.

2.1.1 Normalization of the Boltzmann-Maxwell model

Deriving the plasma parameters used in this study begins by developing a normalization scheme based on the kinetic plasma description. Both the kinetic model introduced in this section, and the normalization scheme applied to it, is used throughout this paper. In kinetic theory, each particle species α is defined by a charge q_α and mass m_α, and is represented by a six dimensional phase space distribution (PSD) f_α. The species are coupled to an electric field $\vec{E}$ and magnetic field $\vec{B}$, and undergo collisions represented by a collision operator C_α. The evolution of each species is described by the Boltzmann equation, which, in index notation, is

$$\frac{\partial}{\partial t} f_\alpha = -v_i \frac{\partial}{\partial x_i} f_\alpha - \frac{q_\alpha}{m_\alpha} E_i \frac{\partial}{\partial v_i} f_\alpha - \frac{q_\alpha}{m_\alpha} \epsilon_{ijk} v_j B_k \frac{\partial}{\partial v_i} f_\alpha + C_\alpha \tag{2.1.1}$$

with physical space coordinate $\vec{x}$, velocity coordinate $\vec{v}$, and Levi-Civita symbol ϵ_{ijk}. Ampere's law describes the evolution of the electric field

$$\frac{\partial}{\partial t} E_i = c^2 \epsilon_{ijk} \frac{\partial}{\partial x_j} B_k - \frac{1}{\epsilon_0} \sum_\alpha q_\alpha \left\langle v_i f_\alpha \right\rangle , \tag{2.1.2}$$

where c is the speed of light, ϵ_0 is the permittivity of free space, and the bracket notation

$$\langle f \rangle = \iiint_{-\infty}^{\infty} f \, dv_x dv_y dv_z \tag{2.1.3}$$

indicates integration over all of velocity space. The magnetic field evolution is described by Faraday's law

$$\frac{\partial}{\partial t} B_i = -\epsilon_{ijk} \frac{\partial}{\partial x_j} E_k. \tag{2.1.4}$$

Finally, the fields are constrained by Gauss's laws

$$\frac{\partial}{\partial x_i} E_i = \frac{1}{\epsilon_0} \sum_\alpha q_\alpha \langle f_\alpha \rangle \tag{2.1.5}$$

and

$$\frac{\partial}{\partial x_i} B_i = 0. \tag{2.1.6}$$

The normalization scheme used in this dissertation is based on proton characteristics. The time, length, and velocity scales for the system are normalized by the system's characteristic time scale $t = \tau \bar{t}$, length scale $x_i = L \bar{x}_i$, and velocity $v_i = v_0 \bar{v}_i$. For this section, the dimensionless variables are marked with a bar unless otherwise stated. The remaining terms are normalized by reference values

$$q_\alpha = Z_\alpha e \qquad B_i = B_0 \bar{B}_i \qquad f_\alpha = n_0 \bar{f}_\alpha$$
$$m_\alpha = A_\alpha m_p \qquad E_i = E_0 \bar{E}_i$$

where e is the elementary charge, m_p is the proton mass, and n_0 is the reference proton number density. The collisional term $C_\alpha = \nu_p n_0 \bar{C}_\alpha$ is normalized by the reference intraspecies proton collision frequency[25]

$$\nu_p = \frac{e^4 n_0 \ln(\Lambda)}{3\sqrt{2}\pi^{3/2} \epsilon_0^2 m_p^{1/2} T_0^{3/2}} \tag{2.1.7}$$

with Coulomb logarithm $\ln(\Lambda)$ and reference temperature $T_0 = m_p v_0^2$. For this study the Coulomb logarithm

$$\ln(\Lambda) = \ln\left(2\pi n_0 \lambda_D^3\right), \tag{2.1.8}$$

with reference proton Debye length

$$\lambda_D = \sqrt{\frac{\epsilon_0 T_0}{n_0 e^2}}, \tag{2.1.9}$$

is assumed to be species independent and slowly varying in space and time. Given the definition in Eq. (2.1.7), the normalized rate of thermalization of species α due to species β is[25]

$$\bar{\nu}_{\alpha\beta} = \frac{\nu_{\alpha\beta}}{\nu_p} = \sqrt{2}\frac{n_\beta Z_\alpha^2 Z_\beta^2}{\sqrt{A_\alpha} T_\alpha^{3/2}} \frac{1 + \frac{A_\alpha}{A_\beta}}{\left(1 + \frac{A_\alpha}{A_\beta}\frac{T_\beta}{T_\alpha}\right)^{3/2}}. \tag{2.1.10}$$

Applying this normalization to the Boltzmann-Maxwell model yields

$$\begin{aligned} \frac{\partial}{\partial \bar{t}} \bar{f}_\alpha = &- \frac{v_0 \tau}{L} \frac{\partial}{\partial \bar{x}_i} \left(\bar{v}_i \bar{f}_\alpha \right) - \frac{e E_0 \tau}{m_p v_0} \frac{Z_\alpha}{A_\alpha} \bar{E}_i \frac{\partial}{\partial \bar{v}_i} \bar{f}_\alpha \\ &- \frac{e B_0 \tau}{m_p} \frac{Z_\alpha}{A_\alpha} \epsilon_{ijk} \bar{v}_j \bar{B}_k \frac{\partial}{\partial \bar{v}_i} \bar{f}_\alpha + (\nu_p \tau) \bar{C}_\alpha, \end{aligned} \tag{2.1.11}$$

$$\frac{\partial}{\partial \bar{t}} \bar{E}_i = \frac{c^2 B_0 \tau}{E_0 L} \epsilon_{ijk} \frac{\partial}{\partial x_j} B_k - \frac{e v_0 n_0 \tau}{E_0 \epsilon_0} \sum_\alpha Z_\alpha \left\langle \bar{v}_i \bar{f}_\alpha \right\rangle, \tag{2.1.12}$$

$$\frac{\partial}{\partial \bar{t}} \bar{B}_i = - \frac{E_0 \tau}{B_0 L} \epsilon_{ijk} \frac{\partial}{\partial \bar{x}_j} \bar{E}_k, \tag{2.1.13}$$

and

$$\frac{\partial}{\partial \bar{x}_i} \bar{E}_i = \frac{e n_0 L}{E_0 \epsilon_0} \sum_\alpha Z_\alpha \left\langle \bar{f}_\alpha \right\rangle. \tag{2.1.14}$$

To simplify the model, the characteristic time and length scales are related by the characteristic velocity $v_0 = L/\tau$, while the permittivity of free space and the reference magnetic field

are related to the proton plasma frequency

$$\omega_p = \sqrt{\frac{n_0 e^2}{m_p \epsilon_0}} \tag{2.1.15}$$

and cyclotron frequency

$$\omega_c = \frac{eB_0}{m_p}. \tag{2.1.16}$$

The reference electric field is related to the plasma frequency by

$$E_0 = \frac{m_p \omega_p^2 L}{e}. \tag{2.1.17}$$

Altogether the normalized Boltzmann equation becomes

$$\begin{aligned} \frac{\partial}{\partial \bar{t}} \bar{f}_\alpha = &- \frac{\partial}{\partial \bar{x}_i} \left(\bar{v}_i \bar{f}_\alpha \right) - (\omega_p \tau)^2 \frac{Z_\alpha}{A_\alpha} \bar{E}_i \frac{\partial}{\partial \bar{v}_i} \bar{f}_\alpha \\ &- (\omega_c \tau) \frac{Z_\alpha}{A_\alpha} \epsilon_{ijk} \bar{v}_j \bar{B}_k \frac{\partial}{\partial \bar{v}_i} \bar{f}_\alpha + (\nu_p \tau) \bar{C}_\alpha \end{aligned} \tag{2.1.18}$$

with Maxwell's equations

$$\frac{\partial}{\partial \bar{t}} \bar{E}_i = \left(\frac{c}{v_0} \right)^2 \frac{(\omega_c \tau)}{(\omega_p \tau)^2} \epsilon_{ijk} \frac{\partial}{\partial x_j} \bar{B}_k - \sum_\alpha Z_\alpha \left\langle \bar{v}_i \bar{f}_\alpha \right\rangle, \tag{2.1.19}$$

$$\frac{\partial}{\partial \bar{t}} \bar{B}_i = - \frac{(\omega_p \tau)^2}{(\omega_c \tau)} \epsilon_{ijk} \frac{\partial}{\partial \bar{x}_j} \bar{E}_k, \tag{2.1.20}$$

$$\frac{\partial}{\partial \bar{x}_i} \bar{E}_i = \sum_\alpha Z_\alpha \left\langle \bar{f}_\alpha \right\rangle, \tag{2.1.21}$$

and

$$\frac{\partial}{\partial \bar{x}_i}\bar{B}_i = 0. \tag{2.1.22}$$

As a side note, the normalized proton skin depth

$$\frac{\delta_p}{L} = \left(\frac{c}{v_0}\right)\frac{1}{(\omega_p \tau)} \tag{2.1.23}$$

is seen in Eq. (2.1.19). For the rest of this document, the bar notation is dropped for the normalized terms. As stated previously, the goal behind this normalization is to identify the dominant behaviors in different plasma regimes.

2.1.2 Parameter regimes and model applicability

Parameter regimes for plasmas, especially multi-species plasmas, are complicated since plasma behavior is described by both spatial scales relating to the characteristic size of a physical phenomena and temporal scales relating to the phenomena's rate of change. Furthermore, temporal and spatial scales are species specific. Since multi-species models are made up of coupled sets of equations, the individual parameter regimes of the various species couple together to define new behaviors.

A high plasma density implies a high plasma frequency $n_0 \propto \omega_p^2$, which results in a plasma that undergoes little charge separation. Charge separation is generally a small effect for laboratory plasmas and is characterized by the ratio of the reference Debye length over the reference length scale λ_D/L. For most macro-scale laboratory plasmas the Debye length is small compared to the scales of interest ($\lambda_D \ll L$), which implies that charge separation is not observable. For micro-scale systems, such as the Debye sheath effects that determine sputtering behavior on the first wall of tokamaks, these charge separation dynamics become

important. When $\lambda_D \ll L$ a plasma is considered quasineutral

$$\sum_{\alpha} Z_\alpha n_\alpha \approx 0, \tag{2.1.24}$$

meaning the electrons density is strongly tied to the ion densities. Quasineutrality is one of the fundamental assumptions made in magnetohydrodynamics (MHD) models which are designed to ignore electron temporal and spatial scales. The accuracy of MHD models are generally limited to regimes concerning macro-scale dynamics over time scales much longer than the electron plasma frequency.

A large cyclotron frequency implies a strong coupling between the plasma and magnetic field ($B_0 \propto \omega_c$). The length scale associated with magnetized behavior is the reference Larmor radius

$$r_L = \frac{v_0}{\omega_c} \tag{2.1.25}$$

which describes the size of a proton gyro-orbit. When $r_L \ll L$ the proton species is considered strongly magnetized and exhibit highly anisotropic behavior.

A high plasma temperature implies a slow thermalization rate $T_0^{3/2} \propto \omega_p^2/\nu_p$. Collisionality in a plasma is characterized by the Knudsen number $\mathrm{Kn} = \lambda_{\mathrm{mfp}}/L$, with collisional mean free path

$$\lambda_{\mathrm{mfp}} = \frac{v_0}{\nu_p}. \tag{2.1.26}$$

Collisional plasma transport is a complex subject, but for the purposes of this study, collisionality is broken down into the four regimes shown in Fig. 2.1.1.

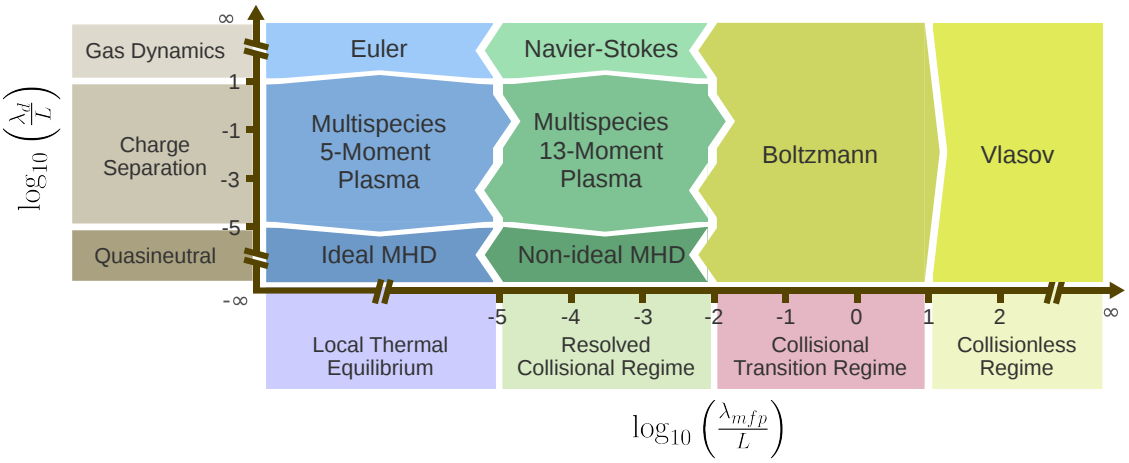


Figure 2.1.1: Collisional and charge separation regimes for various models. Local thermal equilibrium, resolved collisional regime, collisional transition regime, and collisionless regime are used to represent the boundaries between applicable plasma models.

The lowest level of collisionality Kn $>$ 10 is the collisionless regime where dynamics are dominated by advection and electromagnetic forces, and collisional transport is negligible. Astrophysical processes, particle beam physics, and high-temperature fusion-grade plasmas are found in this regime. Kinetic models, such as the Boltzmann model in Eq. (2.1.18), or its collisionless form the Vlasov equation, are required to accurately capture dynamics.

The collisional transition regime[10;18] $10^{-2} <$ Kn < 10 represents an intermediate regime where collisional transport has a large impact on the plasma dynamics, but is too weak to be accurately described by compact 5-moment models. The majority of models used in this regime are based on computationally expensive kinetic descriptions, and the goal of this study is to develop a computationally tractable alternative using moment model descriptions.

The resolved collisional regime $10^{-5} <$ Kn $< 10^{-2}$ is where plasma models must take viscosity and thermal conductivity into account, as well as anisotropies arising from magnetization, making it a complex regime that is well represented by the Braginskii equations[11], as the

plasma does not deviate far from thermal equilibrium. The 13-moment plasma model developed in this study is also applicable to this regime. Throughout this study, the concept of moderate collisionality implies a regime bounding the resolved collisional regime and the collisional transition regime where Kn $\approx 10^{-2}$. The highest collisional regime is where the plasma exists in a perpetual local thermal equilibrium Kn $< 10^{-5}$ such that viscosity and thermal conduction have little effect.

The parameters described above can also relate to coupled plasma regimes such as weakly-coupled[28] or magnetoviscous[29;30] plasmas. Collisions in weakly-coupled plasmas are approximated by small-angle Coulomb collisions as opposed to the hard-sphere collision approximation used for neutral fluids[31]. This weak-coupling condition is characterized by the relation

$$\lambda_{\text{mfp}} \gg \lambda_D, \tag{2.1.27}$$

which is equivalent to a condition on the Coulomb logarithm $\ln(\Lambda) > 1$. In a more practical sense, if charge separation is resolved, then the plasma is in a moderately or weakly collisional regime.

The magnetoviscous regime is best characterized by the Hall parameter

$$\Lambda_H = \frac{\omega_c}{\nu_p} = \frac{\lambda_{\text{mfp}}}{r_L}. \tag{2.1.28}$$

The Hall parameter describes the level of magnetization with respect to the level of collisionality in a plasma. When the Hall parameter is small $\omega_c \ll \nu_p$, collisional transport models for plasmas can be approximated by the same operators used in uncharge fluid dynamics. As the Hall parameter increases to $\omega_c \sim \nu_p$, the viscosity and thermal conductivity become anisotropic based on the orientation of the magnetic field, as discussed in Braginskii[11]. This is due to the decrease of particle mobility perpendicular to magnetic fields since the particles tend to orbit field lines. Magnetization represents one of the parameter regime boundaries

for this model since strongly magnetized collisional transport $\omega_c \gg \nu_p$ is best described by gyrokinetic models[32;33].

As the Boltzmann-Maxwell model presented in Sec. 2.1.1 is not in a relativistic formulation, any moment model derived from it must exist in the non-relativistic regime, which places an upper limit on the particle's thermal velocity

$$T_0 \ll m_p c^2. \tag{2.1.29}$$

A regime linked to the non-relativistic regime is the subluminal Alfvènic regime characterized by

$$\frac{\lambda_D}{r_L} = \frac{v_A}{c} \ll 1. \tag{2.1.30}$$

with reference Alfvén velocity

$$v_A = \frac{B_0}{\sqrt{\mu_0 m_p n_0}} = c\frac{\omega_c}{\omega_p} \tag{2.1.31}$$

where $\mu_0 = (\epsilon_0 c^2)^{-1}$ is the permeability of free space. This regime restricts particles from completing cyclotron orbits within their own Debye spheres. This regime excludes low density, highly magnetized plasmas that often exist in the highly magnetoviscous ($\nu_p \ll \omega_c$) and collisionless regimes.

Figure 2.1.2 shows the practical parameter regime for models developed in this study in terms of temperature and density for a pure hydrogen plasma. Most laboratory scale plasma devices exist in the collisional regime of interest where mean free paths are on the same scale as the device size. In these devices, higher-moment plasma models are advantageous for capturing transport as the plasma transitions into or out of local thermal equilibrium.

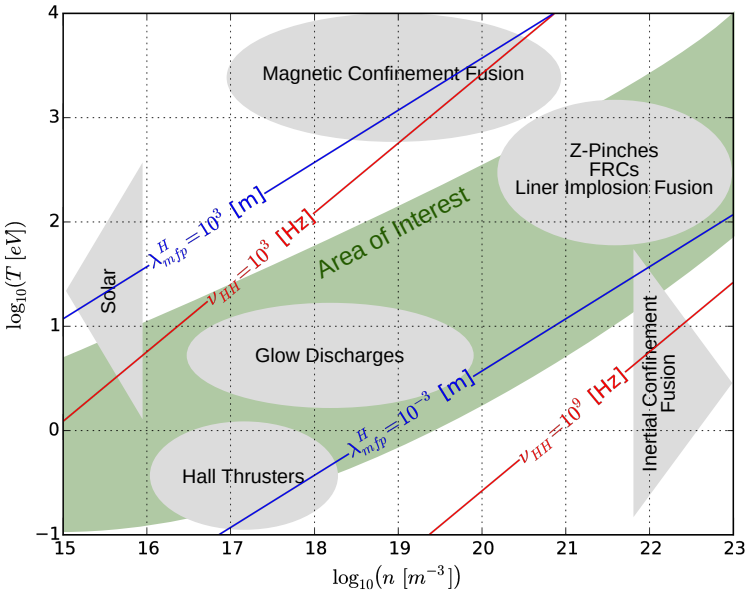


Figure 2.1.2: Area of interest (green) for modeling transport in moderately collisional plasmas given for a hydrogen plasma in terms of plasma temperature and density. Within the green region, the mean free paths associated with collisional transport are on a similar length scale as the noted plasma devices, which implies a moderately collisional plasma. Additional feature scales of interest are the Debye length scale (blue lines) and rate of thermalization (red lines).

An important aspect to consider for multi-species plasmas is that since each species is described by a separate equation set (e.g. Boltzmann equation) they can exist in different plasma parameter regimes. To gauge the individual parameter regimes for each species, the parameters are rewritten in a species specific form. The species temporal scales are governed by the interspecies collision frequency in Eq. (2.1.10), plasma frequency

$$\omega_p^\alpha = \sqrt{\frac{Z_\alpha^2 \bar{n}_\alpha}{A_\alpha}}\omega_p, \tag{2.1.32}$$

and cyclotron frequency

$$\omega_c^\alpha = \frac{Z_\alpha}{A_\alpha}\omega_c, \tag{2.1.33}$$

which all strongly depend on the mass and charge of the species. This implies that the rate of change in collisional, charge separation, and magnetization effects are generally be faster for lighter mass particles (e.g. electrons). When the parameters are reformulated as length scales an interesting result arises. The scale of charge separation is defined by the normalized Debye scale

$$\frac{\lambda_d^\alpha}{L} = \sqrt{\frac{\bar{T}_\alpha}{Z_\alpha^2 \bar{n}_\alpha}}\frac{\lambda_D}{L} \tag{2.1.34}$$

which implies that the charge separation length scales are independent of a particle's mass. A similar scaling is found for the species Knudsen number

$$\mathrm{Kn}_\alpha = \frac{\bar{T}^2}{\bar{n}_\alpha Z_\alpha^4}\mathrm{Kn} \tag{2.1.35}$$

which shows a strong scaling with charge and temperature, but is mass independent. The feature size of collisional transport and charge separation are similar between species with similar temperatures, densities, and charge numbers, however the rate at which the features change is mass dependent. The implication of the mass dependence of the temporal scales is most clearly seen with charge separation effects. When a perturbation in the electric field impacts a plasma, the electrons respond faster than the ions due to their lower mass. This prevents the ions from exhibiting charge separation effects, e.g. ion Langmuir oscillations, even though their Debye length scales are resolved. The collisional transport scaling in Eq. (2.1.10) expresses a similar condition where electrons are generally closer to thermal equilibrium than ions due to their faster rate of thermalization.

The species magnetization scale

$$\frac{r_L^\alpha}{L} = \sqrt{\frac{A_\alpha \bar{T}_\alpha}{Z_\alpha^2}} \frac{r_L}{L} \tag{2.1.36}$$

shows a weaker mass scaling than the temporal scale. This increases the complexity of magnetized collisional transport in multi-species models where different ion species can exist in different magnetoviscous regimes. The implication of these species scaling relations is that multi-species models must be handled carefully when exploring collisional transport in systems containing disparate temperatures, densities, charges, and masses, since each species may require a model particular to the parameter regime in which they exist.

2.2 Deriving moment models

Plasma moment models are derived by taking velocity space moments of the Boltzmann-Maxwell model discussed in Sec. 2.1.1. The normalized Boltzmann equation

$$\begin{aligned} \frac{\partial}{\partial t} f_\alpha = &-v_i \frac{\partial}{\partial x_i} f_\alpha - \frac{Z_\alpha}{A_\alpha} (\omega_p \tau)^2 E_i \frac{\partial}{\partial v_i} f_\alpha \\ &- \frac{Z_\alpha}{A_\alpha} (\omega_c \tau) \epsilon_{ijk} v_j B_k \frac{\partial}{\partial v_i} f_\alpha + (\nu_p \tau) C_\alpha \end{aligned} \tag{2.2.1}$$

is restated in its normalized form. For convenience, a moment Θ_α of the PSD f_α is defined

$$\Theta_\alpha = A_\alpha \iiint_{-\infty}^{\infty} \vartheta f_\alpha \, dv_x dv_y dv_z = A_\alpha \langle \vartheta f_\alpha \rangle, \tag{2.2.2}$$

where $\vartheta(\vec{v})$ is the basis over which the moment is taken. The evolution of each moment Θ_α is described by taking the moment of Eq. (2.2.1)

$$\begin{aligned}\partial_t \Theta_\alpha = &-A_\alpha \partial_{x_j} \langle v_j \vartheta f_\alpha \rangle - \frac{Z_\alpha}{A_\alpha}(\omega_p \tau)^2 E_j A_\alpha \left\langle \vartheta \partial_{v_j} f_\alpha \right\rangle \\ &- \frac{Z_\alpha}{A_\alpha}(\omega_c \tau)\epsilon_{jkl} B_k A_\alpha \left\langle v_k \vartheta \partial_{v_j} f_\alpha \right\rangle + (\nu_p \tau) A_\alpha \langle \vartheta C_\alpha \rangle .\end{aligned} \tag{2.2.3}$$

Note that the evolution of each moment introduces a higher moment in the first term on the right-hand-side of the equation set.

The electric and magnetic field terms in Eq. (2.2.3) can be simplified using integration by parts

$$\left\langle \vartheta \partial_{v_j} f_\alpha \right\rangle = \left\langle \partial_{v_j} (\vartheta f_\alpha) \right\rangle - \left\langle f_\alpha \partial_{v_j} \vartheta \right\rangle . \tag{2.2.4}$$

The divergence theorem can be applied to the first term on the right hand side to convert it to a closed surface integral over all of velocity space. Since the phase space distribution is assumed to be zero on this surface (i.e. no particles with infinite velocities), the first term is ignored such that

$$\begin{aligned}\partial_t \Theta_\alpha = &-A_\alpha \partial_{x_j} \langle v_j \vartheta f_\alpha \rangle + \frac{Z_\alpha}{A_\alpha}(\omega_p \tau)^2 E_j A_\alpha \left\langle f_\alpha \partial_{v_j} \vartheta \right\rangle \\ &+ \frac{Z_\alpha}{A_\alpha}(\omega_c \tau)\epsilon_{jkl} B_k A_\alpha \left\langle f_\alpha \partial_{v_j} (v_k \vartheta) \right\rangle + (\nu_p \tau) A_\alpha \langle \vartheta C_\alpha \rangle .\end{aligned} \tag{2.2.5}$$

The magnetic Lorentz force term can be further simplified

$$\begin{aligned}\epsilon_{jkl} \left\langle f_\alpha \partial_{v_j} (v_k \vartheta) \right\rangle &= \epsilon_{jkl} \left\langle f_\alpha v_k \partial_{v_j} \vartheta \right\rangle + \epsilon_{jkl} \left\langle f_\alpha \vartheta \partial_{v_j} v_k \right\rangle \\ &= \epsilon_{jkl} \left\langle f_\alpha v_k \partial_{v_j} \vartheta \right\rangle + \langle f_\alpha \vartheta \rangle \cancelto{0}{\epsilon_{jkl} \delta_{jk}} \\ &= \epsilon_{jkl} \left\langle f_\alpha v_k \partial_{v_j} \vartheta \right\rangle\end{aligned} \tag{2.2.6}$$

where δ_{ij} is the Kronecker delta. The generalized moment model

$$\begin{aligned}\partial_t \Theta_\alpha = &-A_\alpha \partial_{x_j} \left\langle v_j \vartheta f_\alpha \right\rangle + \frac{Z_\alpha}{A_\alpha}(\omega_p \tau)^2 E_j A_\alpha \left\langle f_\alpha \partial_{v_j} \vartheta \right\rangle \\ &+ \frac{Z_\alpha}{A_\alpha}(\omega_c \tau)\epsilon_{jkl} B_k A_\alpha \left\langle f_\alpha v_k \partial_{v_j} \vartheta \right\rangle + (\nu_p \tau) A_\alpha \left\langle \vartheta C_\alpha \right\rangle,\end{aligned} \tag{2.2.7}$$

is used to derive any moment model given a moment basis ϑ.

2.3 Multi-species 5-moment plasma model

The 5-moment plasma model is a commonly used plasma model[13–16] applicable around thermal equilibrium. The model describes the evolution of five moments associated with the moment basis $\vartheta \in \{1, v_x, v_y, v_z, v^2/2\}$, where $v^2 = \vec{v} \cdot \vec{v}$. The five conserved moments include the mass density

$$\rho_\alpha = A_\alpha \left\langle f_\alpha \right\rangle, \tag{2.3.1}$$

momentum density

$$p_i^\alpha = A_\alpha \left\langle v_i f_\alpha \right\rangle, \tag{2.3.2}$$

and isotropic energy density

$$\epsilon_\alpha = \frac{1}{2} A_\alpha \left\langle v^2 f_\alpha \right\rangle. \tag{2.3.3}$$

These mass density can be rewritten as the number density

$$n_\alpha = \frac{\rho_\alpha}{A_\alpha}, \tag{2.3.4}$$

which describes the number of particles per unit volume. The momentum density can also be written in terms of the flow velocity

$$u_i^\alpha = \frac{p_i^\alpha}{\rho_\alpha}. \tag{2.3.5}$$

The flow velocity is used to derive primitive moments based on a random velocity $\vec{w}_\alpha = \vec{v} - \vec{u}_\alpha$ associated with thermal motion. The isotropic pressure

$$P_\alpha = \frac{1}{3} A_\alpha \left\langle w_\alpha^2 f_\alpha \right\rangle \tag{2.3.6}$$

is the primitive form of the internal energy density of a species, and is valid for an adiabatic index of $\gamma = 5/3$. Conserved variables can always be rewritten in terms of non-conserved primitive moments. For example, the isotropic energy density is related to the isotropic pressure by

$$\epsilon_\alpha = \frac{3}{2} P_\alpha + \frac{1}{2} \rho_\alpha u_\alpha^2. \tag{2.3.7}$$

Using the moment generator in Eq. (2.2.7), the zeroth moment ($\vartheta = 1$) gives the conservation of mass equation

$$\frac{\partial}{\partial t} \rho_\alpha = -\frac{\partial}{\partial x_i} p_i^\alpha + A_\alpha \left\langle C_\alpha \right\rangle \tag{2.3.8}$$

where $A_\alpha \left\langle C_\alpha \right\rangle$ defines the total mass gain or loss due to interactions with other species. The first moment ($\vartheta = v_i$) yields the conservation of momentum equation

$$\frac{\partial}{\partial t} p_i^\alpha = -\frac{\partial}{\partial x_j} e_{ij}^\alpha + Z_\alpha (\omega_p \tau)^2 n_\alpha E_i + Z_\alpha (\omega_c \tau) n_\alpha \epsilon_{ijk} u_j^\alpha B_k + A_\alpha \left\langle v_i C_\alpha \right\rangle \tag{2.3.9}$$

where $A_\alpha \langle v_i C_\alpha \rangle$ is associated with momentum generated through collisions. The divergence operator in the momentum equation depends on the energy density tensor

$$\begin{aligned} e_{ij}^\alpha &= A_\alpha \langle v_i v_j f_\alpha \rangle \\ &= A_\alpha \left\langle (u_i^\alpha + w_i^\alpha)(u_j^\alpha + w_j^\alpha) f_\alpha \right\rangle \\ &= P_{ij}^\alpha + \rho_\alpha u_i^\alpha u_j^\alpha, \end{aligned} \tag{2.3.10}$$

where $P_{ij}^\alpha = A_\alpha \left\langle w_i^\alpha w_j^\alpha f_\alpha \right\rangle$ is the pressure tensor. Note that $P_\alpha = P_{ii}^\alpha/3$ and $\epsilon_\alpha = e_{ii}^\alpha/2$. The evolution of the isotropic energy density ($\vartheta = v^2/2$)

$$\frac{\partial}{\partial t}\epsilon_\alpha = -\frac{1}{2}\frac{\partial}{\partial x_i} H_i^\alpha + Z_\alpha (\omega_p \tau)^2 n_\alpha u_i^\alpha E_i + A_\alpha \frac{1}{2} \left\langle v^2 C_\alpha \right\rangle \tag{2.3.11}$$

depends on the collisional energy change $A_\alpha \langle v^2 C_\alpha \rangle$ and energy flux vector

$$\begin{aligned} H_i^\alpha &= A_\alpha \left\langle v_i v^2 C_\alpha \right\rangle \\ &= A_\alpha \left\langle (u_i^\alpha + w_i^\alpha)(u_j^\alpha + w_j^\alpha)(u_j^\alpha + w_j^\alpha) f_\alpha \right\rangle \\ &= \rho_\alpha u_i^\alpha u_\alpha^2 + 2u_j^\alpha P_{ij}^\alpha + 3u_i^\alpha P_\alpha + 2q_i^\alpha, \end{aligned} \tag{2.3.12}$$

with heat flux vector $q_i^\alpha = A_\alpha \langle w_i^\alpha w_\alpha^2 f_\alpha \rangle$.

Since the 5-moment model is designed to operate near thermal equilibrium where the pressure tensor is isotropic, the pressure tensor is usually rewritten using an anisotropic pressure tensor

$$\Pi_{ij}^\alpha = P_{ij}^\alpha - P_\alpha \delta_{ij}. \tag{2.3.13}$$

The anisotropic pressure tensor converts the momentum equation into a more familiar

form

$$\begin{aligned}\frac{\partial}{\partial t}p_i^\alpha = &-\frac{\partial}{\partial x_j}\left(\rho_\alpha u_i^\alpha u_j^\alpha + P_\alpha \delta_{ij} + \Pi_{ij}^\alpha\right) \\ &+ Z_\alpha(\omega_p\tau)^2 n_\alpha E_i + Z_\alpha(\omega_p\tau)^2 n_\alpha \epsilon_{ijk} u_j^\alpha B_k + A_\alpha \left\langle v_i C_\alpha \right\rangle,\end{aligned} \tag{2.3.14}$$

while the isotropic energy equation becomes

$$\frac{\partial}{\partial t}\epsilon_\alpha = -\frac{\partial}{\partial x_i}\left(u_i^\alpha\left(\epsilon_\alpha + P_\alpha\right) + u_j^\alpha \Pi_{ij}^\alpha + q_i^\alpha\right) + Z_\alpha(\omega_p\tau)^2 n_\alpha u_i^\alpha E_i + \frac{1}{2}A_\alpha\left\langle v^2 C_\alpha\right\rangle. \tag{2.3.15}$$

The collision integrals $A_\alpha \langle C_\alpha \rangle$, $A_\alpha \langle v_i C_\alpha \rangle$, and $A_\alpha \langle v^2 C_\alpha \rangle$ are discussed later.

At this point the 5-moment model is incomplete since the anisotropic pressure tensor, and heat flux vector are undefined. To close the 5-moment model, these unknown moments must be related to the known moments. In a highly collisional system, the phase space distribution is considered Maxwellian

$$f_\alpha = \frac{n_\alpha}{\left(2\pi\bar{u}_\alpha\right)^{\frac{3}{2}}}\exp\left(-\frac{1}{2}\frac{w_\alpha^2}{\bar{u}_\alpha^2}\right) \tag{2.3.16}$$

with thermal velocity

$$\bar{u}_\alpha = \sqrt{\frac{P_\alpha}{\rho_\alpha}}. \tag{2.3.17}$$

Since this Maxwellian form is entirely dependent on the five known moments n_α, $\vec{u}_\alpha$, and P_α, the closure scheme can be related to using the distribution to define the unknown moments, which is simply $\mathbf{\Pi}_\alpha = 0$ and $\vec{q}_\alpha = 0$. This is known as the ideal 5-moment model, which is only applicable to systems that remain in local thermal equilibrium. For small deviations around thermal equilibrium, the unknown moments can be derived using a perturbation method known as the Chapman-Enskog expansion[10]. The expansion becomes complicated for strongly magnetized plasmas (see Braginskii[11]), however for weakly magnetoviscous plas-

mas $\omega_c \ll \nu_p$, the process results in an anisotropic pressure tensor

$$\Pi_{ij}^{\alpha} = -\frac{\mu_\alpha}{(\nu_p \tau)} \left(\partial_{x_i} u_j^\alpha + \partial_{x_j} u_i^\alpha - \frac{2}{3} \delta_{ij} \partial_{x_k} u_k^\alpha \right), \tag{2.3.18}$$

with normalized viscosity μ_α. The heat flux vector

$$q_i^\alpha = -\frac{\kappa_\alpha}{(\nu_p \tau)} \partial_{x_i} T_\alpha \tag{2.3.19}$$

is similarly constructed with a normalized thermal conductivity κ_α and gradients of the isotropic temperature $T_\alpha = P_\alpha / n_\alpha$. The main drawback of this expansion based closure is that the model is only valid when the species is near thermal equilibrium and weakly magnetized. To understand the magnetically coupled collisional transport effects that are missing, it is important to include additional moments.

2.4 Multi-species 13-moment plasma model

Like the 5-moment model, the 13-moment model is derived by evaluating Eq. (2.2.7) for the first 13 scalar moments including the mass density ρ_α (Eq. (2.3.1)), the momentum density $\vec{p}_\alpha$ (Eq. (2.3.2)), the energy density tensor $\mathbf{e}_\alpha$ (Eq. (2.3.10)), and the energy flux vector $\vec{H}_\alpha$ (Eq. (2.3.12)). Note that all of these moments show up in the 5-moment model, however now the complete energy tensor and energy flux vector are evolved as well as the mass and momentum equations. The conservation of mass equation

$$\frac{\partial}{\partial t} \rho_\alpha = -\frac{\partial}{\partial x_i} p_i^\alpha + A_\alpha \langle C_\alpha \rangle \tag{2.4.1}$$

and the conservation of momentum equation

$$\frac{\partial}{\partial t} p_i^\alpha = -\frac{\partial}{\partial x_j} e_{ij}^\alpha + Z_\alpha (\omega_p \tau)^2 n_\alpha E_i + Z_\alpha (\omega_c \tau) n_\alpha \epsilon_{ijk} u_j^\alpha B_k + A_\alpha \langle v_i C_\alpha \rangle \tag{2.4.2}$$

remain unchanged, and are restated here for completeness. The evolution of the energy tensor

$$\begin{aligned}\frac{\partial}{\partial t}e_{ij}^{\alpha} = &-\frac{\partial}{\partial x_k}H_{ijk}^{\alpha} + Z_{\alpha}(\omega_p\tau)^2 n_{\alpha}\left(u_i^{\alpha}E_j + u_j^{\alpha}E_i\right) \\ &+ \frac{Z_{\alpha}}{A_{\alpha}}(\omega_c\tau)\left(\epsilon_{ikl}e_{jk}^{\alpha} + \epsilon_{jkl}e_{ik}^{\alpha}\right)B_l + A_{\alpha}\left\langle v_i v_j C_{\alpha}\right\rangle\end{aligned} \tag{2.4.3}$$

introduces an energy flux tensor

$$\begin{aligned}H_{ijk}^{\alpha} &= A_{\alpha}\left\langle v_i v_j v_k f_{\alpha}\right\rangle. \\ &= \rho_{\alpha}u_i^{\alpha}u_j^{\alpha}u_k^{\alpha} + u_i^{\alpha}P_{jk}^{\alpha} + u_j^{\alpha}P_{ik}^{\alpha} + u_k^{\alpha}P_{ij}^{\alpha} + h_{ijk}^{\alpha}\end{aligned} \tag{2.4.4}$$

with heat flux tensor $h_{ijk}^{\alpha} = A_{\alpha}\left\langle w_i^{\alpha}w_j^{\alpha}w_k^{\alpha}f_{\alpha}\right\rangle$. Note that the heat flux vector is defined $q_i^{\alpha} = h_{ijj}^{\alpha}/2$. The evolution of the energy tensor is anisotropically coupled to the magnetic field and includes anisotropic heating terms due to collisions, both of which are largely ignored in the 5-moment model presented in Sec. 2.3. The final evolution equation relates to the energy flux vector

$$\frac{\partial}{\partial t}H_i^{\alpha} = -\frac{\partial}{\partial x_j}G_{ij}^{\alpha} + \frac{Z_{\alpha}}{A_{\alpha}}(\omega_p\tau)^2\left(e_{jj}^{\alpha}E_i + 2e_{ij}^{\alpha}E_j\right) + \frac{Z_{\alpha}}{A_{\alpha}}(\omega_c\tau)\epsilon_{ijk}H_j^{\alpha}B_k + A_{\alpha}\left\langle v_i v^2 C_{\alpha}\right\rangle, \tag{2.4.5}$$

which includes a fourth tensor moment

$$\begin{aligned}G_{ij}^{\alpha} &= A_{\alpha}\left\langle v_i v_j v^2 f_{\alpha}\right\rangle = A_{\alpha}\left\langle (u_i^{\alpha}+w_i^{\alpha})(u_j^{\alpha}+w_j^{\alpha})(u_k^{\alpha}+w_k^{\alpha})(u_k^{\alpha}+w_k^{\alpha})f_{\alpha}\right\rangle \\ &= \rho_{\alpha}u_i^{\alpha}u_j^{\alpha}u_{\alpha}^2 + 3u_i^{\alpha}u_j^{\alpha}P_{\alpha} + 2u_i^{\alpha}u_k^{\alpha}P_{jk}^{\alpha} + 2u_j^{\alpha}u_k^{\alpha}P_{ik}^{\alpha} + u_{\alpha}^2P_{ij}^{\alpha} \\ &\quad + 2u_i^{\alpha}q_j^{\alpha} + 2u_j^{\alpha}q_i^{\alpha} + 2u_k^{\alpha}h_{ijk}^{\alpha} + g_{ij}^{\alpha}\end{aligned} \tag{2.4.6}$$

where $g_{ij}^{\alpha} = A_{\alpha}\left\langle w_i^{\alpha}w_j^{\alpha}w_{\alpha}^2 f_{\alpha}\right\rangle$ is the primitive fourth tensor moment. Note that the evolution equations for the energy tensor and energy flux couple to the magnetic field similar to what is seen in Braginskii[11] model. As was seen with the 5-moment model, the 13-moment model

is incomplete until $\mathbf{h}_\alpha$ and $\mathbf{g}_\alpha$ are defined.

2.4.1 13-moment closure scheme

Closing moment model means to define all unknown moments in terms of the known moments. In the case of the 5-moment model operating near thermal equilibrium, closure is accomplished by defining the anisotropic pressure tensor in Eq. (2.3.14) and the heat flux vector in Eq. (2.3.15) in terms of the density, flow velocity, and temperature. This process introduced viscosity and thermal conductivity as a consequence of leaving local thermal equilibrium.

Closure schemes become increasingly difficult to derive and evaluate for higher-moment models[12;19], as they tend to become nonlinear and lose both hyperbolicity and realizability. Hyperbolicity is required for wave-like dynamics consistent with physical causality, and helps enforce conservation and stability in numerical methods. Realizability implies physicality in that the velocity space distribution remains positive at all points in phase space. Nonlinear closures generally lack a closed form relation for defining the unknown moments and require iterative procedures to evaluate.

There are many options for higher-moment model closures, but most have the drawbacks of being computationally expensive and difficult to evaluate. Maximum entropy closures[12;17] attempt to solve the higher-moment closure problem by assuming a form for the PSD designed to enforce stability, however these methods tend to require complex iterative methods to solve. Grad methods treat higher-moment closure schemes by describing deviations from thermal equilibrium using Hermite polynomials, however these methods tend to lose stability in the presence of transonic flow and strong shearing forces. More recent developments of regularized Grad closures[20;21] have been shown to increase stability in multiple dimensions and may make for an ideal alternative for the closure presented in this section. The 13-moment plasma model developed in this study uses the closure from Torrilhon[19] which assumes the

phase space distribution is well described by a Pearson type-IV distribution. The Pearson-IV closure has been shown to be stable for a wide range of one dimensional dynamics, which is advantageous for systems containing strong deviations from thermal equilibrium. While the closure is nonlinear, the simplicity of its closure relations make it an attractive base for computationally efficient plasma models.

The Pearson type-IV distribution introduces skew and kurtosis to an otherwise Maxellian distribution. Skew refers to the leaning of a distribution, which is directly related to the odd moments of a fluid such as the heat flux $\mathbf{h}$. Kurtosis is a symmetric modification to the distribution relating to its pointedness and affects the even moments such as the 4^{th} tensor moment. Figure 2.4.1 shows two examples of the one dimensional Pearson type-IV distributions compared to the Maxwellian distribution used for the 5-moment model. All three distributions have the same density, average velocity, and thermal velocity, however the Pearson type-IV distribution can represent additional states associated with leaving thermal equilibrium.

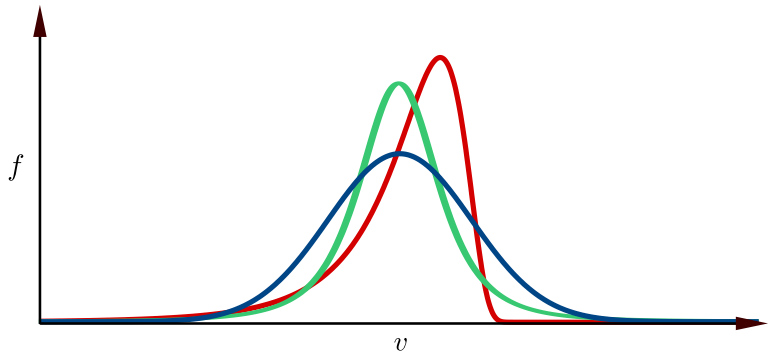


Figure 2.4.1: Comparison of two Pearson type-IV distributions (green and red) to a Maxwellian distribution (blue) representing a distribution in thermal equilibrium. All distributions have the same density ($n = 1$), average velocity ($u_x = 0$), and temperature ($T = 1$). The high kurtosis Pearson type-IV distribution (green) is more pointed than the Maxwellian and has more particles found in the tails of the distribution. The high skew Pearson type-IV distribution is noticeably non-symmetric, which has a large influence on the heat flux of a species.

The goal of the Pearson-IV closure is to include information about the width, skew, and kurtosis that arise from leaving thermal equilibrium which are not available in the 5-moment model. Ignoring species identifiers, the closure defines the two unknown tensor moments $\mathbf{h}$ and $\mathbf{g}$, expressed in Eqs. (2.4.4) and (2.4.6), in terms of a compact set of closure variables Y, $\vec{N}$, and K which relate to the shape of the Pearson type-IV distribution. Here Y scales with the magnitude of the distribution's skew, $\vec{N}$ relates to the skew direction, and K describes the distribution's kurtosis. The heat flux tensor can be written as[19]

$$h_{ijk} = \frac{1}{2} Y \left(N_i P_{jk} + N_j P_{ki} + N_k P_{ij} - N_i N_j N_k \right), \tag{2.4.7}$$

which, predictably, is purely dependent on the fluid's density, pressure, and the distribution's

skew. The fourth tensor moment[19]

$$\begin{aligned} g_{ij} = &\frac{1}{4}Y^2\left(N^2P_{ij} + 2N_k\left(N_jP_{ik} + N_iP_{jk}\right)\right) \\ &+ \frac{3}{4}Y^2N_iN_j\left(P - N^2\right) \\ &+ K\left(3PP_{ij} + 2P_{ik}P_{kj}\right) \end{aligned} \tag{2.4.8}$$

shows a direct relationship to the distribution's kurtosis.

The closure variables Y and $\vec{N}$ can be defined by taking the tensor contraction of Eq. (2.4.7)

$$q_i = \frac{1}{2}h_{ijj} = \frac{Y}{4}\left(\left(3P - N^2\right)\delta_{ij} + 2P_{ij}\right)N_j. \tag{2.4.9}$$

This is a nonlinear relation between Y and $\vec{N}$ and the known moments $\mathbf{P}$, and $\vec{q}$ which means that an iterative solver is required to retrieve the closure variables. There are two modifications that can be made to Eq. (2.4.9) to simplify the process. The first is to use the definition $N_i = \sqrt{P}_{ij}n_j$ from Torrilhon[19], where $\vec{n}$ is a unit vector pointing along the direction of skew, to yield

$$q_i = \frac{Y}{4}\left(\left(3P - P_{kl}n_kn_l\right)\delta_{ij} + 2P_{ij}\right)\sqrt{P}_{jm}n_m. \tag{2.4.10}$$

The second is to rotate the system to a reference frame where the symmetric tensor $\mathbf{P}$ is diagonal

$$P_{ij} = V_{ik}\Lambda_{kl}V_{jl}, \tag{2.4.11}$$

where $\mathbf{V}$ is the orthogonal rotation matrix and the diagonal values of $\mathbf{\Lambda}$ are the principal pressures. This diagonalization is used to define the square root of the pressure tensor

$$\sqrt{P}_{ij} = V_{ik}\sqrt{\Lambda_{kl}}V_{jl}. \tag{2.4.12}$$

Applying these simplifications to Eq. (2.4.10) results in three coupled equations

$$A_i \left(\frac{x_j x_k \Lambda_{jk}}{x_l x_l} \right) x_i = b_i \tag{2.4.13}$$

where

$$b_i = 4 V_{ji} q_j, \tag{2.4.14}$$

$$x_i = Y V_{ji} n_j, \tag{2.4.15}$$

and

$$A_i(\gamma) = (2\Lambda_i + 3P - \gamma)\sqrt{\Lambda_i}. \tag{2.4.16}$$

The nonlinear equation found in Eq. (2.4.13) is solved using the iterative process

$$x_i^{r+1} = \frac{b_i}{A_i(\gamma^r)} \tag{2.4.17}$$

with iterative guess

$$\gamma^r = \frac{x_i^r x_j^r \Lambda_{ij}}{x_k^r x_k^r}. \tag{2.4.18}$$

At least two iterations are required to converge to an adequate solution[19] with an initial guess of $\gamma^0 = P$. Note that additional iterations are cheap compared to diagonalizing $\mathbf{P}$. Once the solution is adequately converged, the closure variables are retrieved as

$$Y = |\vec{x}| \tag{2.4.19}$$

and

$$N_i = \frac{1}{Y} V_{ij} x_k \sqrt{\Lambda_{jk}}. \tag{2.4.20}$$

The kurtosis closure variable K is defined using the "singular closure" of Torrilhon[19] which relates the kurtosis to the magnitude of the distribution skew $K = 1/\rho + Y^2/4$. The combination of Y, $\vec{N}$, and K are used in Eqs. (2.4.7) and (2.4.8) to retrieve $\mathbf{h}$ and $\mathbf{g}$ which closes the 13-moment model. Given an efficient means for diagonalizing the positive definite, symmetric pressure tensor, the Pearson type-IV closure is an effective means of closing the 13-moment model. It is important to note that this closure is defined for neutral fluid dynamics, and does not take magnetization or thermalization into account. For application to strongly magnetized systems, where $\mathbf{h}$ and $\mathbf{g}$ are anisotropically coupled to the magnetic field, the closure needs to be modified. As this closure is used primarily for setting the groundwork for a weakly to moderately magnetized 13-moment model, the magnetic coupling of the unknown moments is ignored. Before this model can be implemented using a discontinuous numerical method, it is important to have an understanding of the closure's characteristics.

Characteristic speeds

The 13-moment model is presented in conservative form for use in discontinuous numerical methods. Enforcing stability in discontinuous methods, especially for nonlinear hyperbolic systems, depends on how fluxes on discontinuous interfaces between elements are evaluated. These interface fluxes, or numerical fluxes, depend strongly on the characteristics of the flux. Since the Pearson-IV closure scheme is nonlinear and relies on a iterative procedure to solve, a closed form characteristic decomposition of the flux Jacobian is difficult. To counter this, an approximate Riemann solution is used based the Harten-Lax-Van Leer (HLL) numerical flux[19]. The HLL flux is designed to circumvent the full characteristic decomposition

by approximating the full Riemann solution using two waves instead of a possible thirteen. The HLL flux does not require exact wave speeds; however, discontinuous methods may become excessively diffusive and/or numerically stiff when the wave speeds are poorly approximated.

An HLL numerical flux requires the maximum and minimum characteristic speeds $u_{\max}$ and $u_{\min}$ along the surface normal of a given interface between elements. As an example, the following characteristics are defined for flow along the x-axis. For this study, the characteristic speeds are approximated from the one dimensional closure solution defined in Torrilhon [19]

$$Y = \frac{4q_x}{N_x(P_{xx} + 3P)}. \tag{2.4.21}$$

where

$$\vec{N} = \begin{bmatrix} \sqrt{P_{xx}} & 0 & 0 \end{bmatrix}^T. \tag{2.4.22}$$

Deriving the characteristics is complex process; however, the result is represented by nine wave speeds λ. The wave speeds are defined in terms of a thermal velocity $\bar{u} = \sqrt{P/\rho}$, a directional thermal velocity $\bar{u}_{ij} = \sqrt{P_{ij}/\rho}$, and a transport velocity

$$\mu_x = \frac{2q_x}{\rho \upsilon_x^2} \tag{2.4.23}$$

where

$$\upsilon_x = \sqrt{\bar{u}_{xx}^2 + 3\bar{u}^2}. \tag{2.4.24}$$

The four simplest wave speeds are $\lambda = u_x$, $\lambda = u_x + \mu_x$, and $\lambda = u_x + \mu_x \pm \sqrt{\mu_x^2 + \bar{u}_{xx}^2}$. Additional characteristic speeds are given by the roots of the quintic equation $\sum_{n=0}^{5} a_n(\lambda -$

$u_x)^n = 0$ where

$$a_0 = -2A^2(A+1)\mu_x \upsilon_x^4, \tag{2.4.25}$$

$$a_1 = -A(-3A + B(8A+6))\upsilon_x^4, \tag{2.4.26}$$

$$a_2 = -4(-A(3+2A) + B(A+1))\mu_x \upsilon_x^2, \tag{2.4.27}$$

$$a_3 = 2(-2A(A+1) + B(2A+5))\upsilon_x^2, \tag{2.4.28}$$

$$a_4 = -2(A+3)\mu_x, \tag{2.4.29}$$

and

$$a_5 = 1 \tag{2.4.30}$$

with $A = (2 + (P_{yy} + P_{zz})/P_{xx})^{-1}$ and $B = \mu_x^2/\upsilon_x^2$. The bounds of these roots are given by the Laguerre-Samuelson inequality

$$|5(\lambda - u_x) + a_4| < 2\sqrt{6a_4^2 - 10a_3}, \tag{2.4.31}$$

which is adequate for approximating $u_{\max} = \max(\lambda)$ and $u_{\min} = \min(\lambda)$ in a HLL flux for applications where the phase space distribution remains near Maxwellian. Experimentation has shown that the characteristic speeds in subsonic flows are rarely above three times the thermal velocity.

2.5 Collision integrals

Up to this point the 5-moment and 13-moment models have been defined and closed, however the collision integrals describing the change in mass $A_\alpha \langle C_\alpha \rangle$, momentum $A_\alpha \langle v_i C_\alpha \rangle$, energy $A_\alpha \langle v_i v_j C_\alpha \rangle$, and energy flux $A_\alpha \langle v_i v^2 C_\alpha \rangle$ due to collisions both within and between species were left undefined. The first step in deriving these terms is to convert the collision integrals into primitive form. The primitive forms for the collision terms are found using the random velocity $\vec{w}_\alpha = \vec{v} - \vec{u}_\alpha$ to decouple the advective contribution due to the fluid's flow from the thermal contribution associated with the random velocity. Decoupling the collisional terms into thermal and translational contributions simplifies the evaluation of collision integrals. The decoupled form for the change in momentum,

$$A_\alpha \langle v_i C_\alpha \rangle = A_\alpha \iiint_{-\infty}^{\infty} v_i C_\alpha \, dv_x dv_y dv_z \tag{2.5.1}$$

$$= A_\alpha \langle (w_i^\alpha + u_i^\alpha) C_\alpha \rangle \tag{2.5.2}$$

$$= R_i^\alpha + u_i^\alpha A_\alpha \langle C_\alpha \rangle , \tag{2.5.3}$$

takes into account both the drag

$$R_i^\alpha = A_\alpha \langle w_i^\alpha C_\alpha \rangle \tag{2.5.4}$$

and change in mass due to collisions. The change in energy density,

$$\begin{aligned} A_\alpha \langle v_i v_j C_\alpha \rangle &= A_\alpha \left\langle (w_i^\alpha + u_i^\alpha)(w_j^\alpha + u_j^\alpha) C_\alpha \right\rangle \\ &= Q_{ij}^\alpha + u_i^\alpha R_j^\alpha + u_j^\alpha R_i^\alpha + u_i^\alpha u_j^\alpha A_\alpha \langle C_\alpha \rangle , \end{aligned} \tag{2.5.5}$$

contains a collisional heating term

$$Q_{ij}^\alpha = A_\alpha \left\langle w_i^\alpha w_j^\alpha C_\alpha \right\rangle . \tag{2.5.6}$$

The 5-moment model uses the change in isotropic energy

$$\begin{aligned} A_\alpha \left\langle v^2 C_\alpha \right\rangle &= A_\alpha \left\langle (w_i^\alpha + u_i^\alpha)(w_i^\alpha + u_i^\alpha) C_\alpha \right\rangle \\ &= 3Q_\alpha + 2u_i^\alpha R_i^\alpha + u_\alpha^2 A_\alpha \left\langle C_\alpha \right\rangle \end{aligned} \tag{2.5.7}$$

where $Q_\alpha = Q_{ii}^\alpha/3$ is the isotropic heating. The final term of interest is the change in energy flux

$$\begin{aligned} A_\alpha \left\langle v_i v^2 C_\alpha \right\rangle &= A_\alpha \left\langle (w_i^\alpha + u_i^\alpha)(w_j^\alpha + u_j^\alpha)(w_j^\alpha + u_j^\alpha) C_\alpha \right\rangle \\ &= W_i^\alpha + 3u_i^\alpha Q_\alpha + 2u_j^\alpha Q_{ij}^\alpha \\ &\quad + 2u_i^\alpha u_j^\alpha R_j^\alpha + u_\alpha^2 R_i^\alpha + u_i^\alpha u_\alpha^2 A_\alpha \left\langle C_\alpha \right\rangle \end{aligned} \tag{2.5.8}$$

with collisional heat flux term

$$W_i^\alpha = A_\alpha \left\langle w_i^\alpha w_\alpha^2 C_\alpha \right\rangle . \tag{2.5.9}$$

The collision integrals implemented for the 13-moment plasma model are designed to be consistent with the 5-moment plasma model, and the following sections introduce three collision operators to handle interactions both within a species and between species. These operators are defined in terms of the primitive collision terms $\langle C_\alpha \rangle$, $\vec{R}_\alpha$, $\mathbf{Q}_\alpha$, and $\vec{W}_\alpha$, which will be used in conjunction with Eqs. (2.5.3), (2.5.5), and (2.5.8) for use in conservation form 5-moment and 13-moment models. The two operators described in Sec. 2.5.1 and Sec. 2.5.2 deal with collisions within a species and act to enforce proper thermal equilibration and increase the stability of the 13-moment plasma model. The third collision operator, discussed in Sec.2.5.3, describes scattering collisions between species.

2.5.1 Intraspecies collision operator

In general, collisional processes in weakly-coupled plasmas are well represented by the Fokker-Planck collision operator[25]. As this study is a preliminary exploration into higher-moment plasma models, a simple BGK relaxation collision operator,

$$C_\alpha = -\nu_{\alpha\alpha}\left(f_\alpha - \tilde{f}_\alpha\right), \tag{2.5.10}$$

is used. The BGK collision operator represents scattering collisions within a species, with the effect of driving the system toward thermal equilibrium at a rate $\nu_{\alpha\alpha}$ defined in Eq. (2.1.10). The equivalent Maxwellian $\tilde{f}_\alpha$, defined in Eq. (2.3.16), represents the fluid in local thermal equilibrium while conserving density, momentum, and energy. The change in mass density due to scattering collisions is given by the zeroth primitive moment of the BGK collision operator,

$$\begin{aligned} A_\alpha \langle C_\alpha \rangle &= -\nu_{\alpha\alpha}\left(A_\alpha \langle f_\alpha \rangle - A_\alpha \left\langle \tilde{f}_\alpha \right\rangle\right) \\ &= -\nu_{\alpha\alpha}\left(\rho_\alpha - A_\alpha n_\alpha\right) \\ &= 0, \end{aligned} \tag{2.5.11}$$

which states that scattering collisions do not create or destroy mass. The drag associated with the BGK collision operator,

$$\begin{aligned} R_i^\alpha &= A_\alpha \langle w_i^\alpha C_\alpha \rangle \\ &= -\nu_{\alpha\alpha}\left(A_\alpha \langle w_i^\alpha f_\alpha \rangle - A_\alpha \left\langle w_i^\alpha \tilde{f}_\alpha \right\rangle\right), \end{aligned} \tag{2.5.12}$$

is simplified using the definitions

$$\langle w_i^\alpha f_\alpha \rangle = \langle v_i f_\alpha \rangle - u_i^\alpha \langle f_\alpha \rangle = 0 \tag{2.5.13}$$

and

$$\left\langle w_i^\alpha \tilde{f}_\alpha \right\rangle = \frac{n_\alpha}{(2\pi \bar{u}_\alpha)^{\frac{3}{2}}} \left\langle w_i^\alpha \exp\left(-\frac{1}{2}\frac{w_\alpha^2}{\bar{u}_\alpha^2}\right)\right\rangle = 0 \tag{2.5.14}$$

to show that scattering collisions within a species do not affect the fluid's momentum

$$\vec{R}_\alpha = 0. \tag{2.5.15}$$

The collisional heating,

$$\begin{aligned} Q_{ij}^\alpha &= A_\alpha \left\langle w_i^\alpha w_j^\alpha C_\alpha \right\rangle \\ &= -\nu_{\alpha\alpha}\left(A_\alpha \left\langle w_i^\alpha w_j^\alpha f_\alpha \right\rangle - A_\alpha \left\langle w_i^\alpha w_j^\alpha \tilde{f}_\alpha \right\rangle\right) \\ &= -\nu_{\alpha\alpha}\left(P_{ij}^\alpha - A_\alpha n_\alpha \bar{u}_\alpha^2 \delta_{ij}\right) \\ &= -\nu_{\alpha\alpha}\left(P_{ij}^\alpha - P_\alpha \delta_{ij}\right), \end{aligned} \tag{2.5.16}$$

drives the pressure tensor $\mathbf{P}_\alpha$ to isotropy, represented by $P_{ij}^\alpha = P_\alpha \delta_{ij}$, over a time scale related to the interspecies collision frequency. In the case of the 5-moment model, only the isotropic heating component,

$$Q_\alpha = Q_{ii}^\alpha = -\nu_{\alpha\alpha}\left(P_{ii}^\alpha - P_\alpha \delta_{ii}\right) = 0, \tag{2.5.17}$$

is required, which, when combined with the Eq. (2.5.11) and Eq. (2.5.15), implies that the BGK collision operator has no effect on systems in local thermal equilibrium. This is an expected result since the Maxwellian distribution is fully described by the five known moments. The collisional heat flux term

$$\begin{aligned} W_i^\alpha &= A_\alpha \left\langle w_i^\alpha w_\alpha^2 C_\alpha \right\rangle = -\nu_{\alpha\alpha}\left(A_\alpha \left\langle w_i^\alpha w_\alpha^2 f_\alpha \right\rangle - A_\alpha \left\langle w_i^\alpha w_\alpha^2 \tilde{f}_\alpha \right\rangle\right) \\ &= -\nu_{\alpha\alpha}\left(2q_i^\alpha - 0\right) = -2\nu_{\alpha\alpha} q_i^\alpha \end{aligned} \tag{2.5.18}$$

implies that the heat flux is driven to zero at the rate of thermalization, conducive to approaching thermal equilibrium.

Importantly, the BGK collision operator, along with all other collision operators, is designed for models that accurately describe collisionless plasma dynamics. Kinetic models, like the Boltzmann model, are able to represent collisionless dynamics well, while moment models, such as the 13-moment model, are generally not applicable to this regime. This implies that as the thermalization rate $\nu_{\alpha\alpha}$ decreases, the underlying features of the moment model's closure, namely the artificial wave or sub-shock structure[17;19], will begin to drive non-physical effects. For weakly-collisional systems, an additional "collisionless operator" is required to represent the kinetic effects associated with leaving thermal equilibrium.

2.5.2 Diffusive stabilization for moderate collisionality

The diffusive stabilization operator is designed to approximate effects associated with decreasing collisionality in a fluid. The closure scheme used for the 13-moment model does not take the effect of collisionality into account in the highest unknown moments h_{ijk} and g_{ij}. While the closure promotes stability, it does not describe collisionless dynamics on its own. This section attempts to develop an operator to be used in conjunction with the BGK collision operator from Sec. 2.5.1 to approximate moderate collisionality. Since the underlying closure of the 13-moment model is hyperbolic, the solution evolves as a set of artificial waves. These artificial waves, while stable, do not represent a physically accurate fluid flow, and the the goal of the operator is to add isotropic diffusion to damp the non-physical waves without modifying mass, flow velocity, or isotropic energy. Since the BGK collision operator enforces the conservation of mass, momentum, and isotropic energy, it is used as a base for the diffusive stabilization operator

$$C_\alpha = \frac{\partial}{\partial x_k}\left(D_\alpha \frac{\partial}{\partial x_k}\left(f_\alpha - \tilde{f}_\alpha\right)\right). \tag{2.5.19}$$

The operator applies diffusion to the anisotropic portions of the pressure tensor,

$$\begin{aligned} Q_{ij}^{\alpha} &= A_{\alpha} \left\langle w_i^{\alpha} w_j^{\alpha} C_{\alpha} \right\rangle \\ &= \frac{\partial}{\partial x_k} \left(D_{\alpha} \frac{\partial}{\partial x_k} \left(A_{\alpha} \left\langle w_i^{\alpha} w_j^{\alpha} \left(f_{\alpha} - \tilde{f}_{\alpha} \right) \right\rangle \right) \right) \\ &= \frac{\partial}{\partial x_k} \left(D_{\alpha} \frac{\partial}{\partial x_k} \left(P_{ij}^{\alpha} - P_{\alpha} \delta_{ij} \right) \right), \end{aligned} \tag{2.5.20}$$

which acts to keep the pressure tensor positive definite. This aids in stabilizing the 13-moment model in the presence of strong anisotropies associated with multidimensional flow and magnetization. The operator also applies diffusion to the heat flux vector,

$$\begin{aligned} W_i^{\alpha} &= A_{\alpha} \left\langle w_i^{\alpha} w_{\alpha}^2 C_{\alpha} \right\rangle \\ &= \frac{\partial}{\partial x_j} \left(D_{\alpha} \frac{\partial}{\partial x_j} \left(A_{\alpha} \left\langle w_i^{\alpha} w_{\alpha}^2 \left(f_{\alpha} - \tilde{f}_{\alpha} \right) \right\rangle \right) \right) \\ &= \frac{\partial}{\partial x_j} \left(2 D_{\alpha} \frac{\partial}{\partial x_j} q_i^{\alpha} \right), \end{aligned} \tag{2.5.21}$$

which helps keep the model stable in anisotropic systems. The rate of diffusion

$$D_{\alpha} = \frac{P_{\alpha}}{(\nu_p \tau)^2 \rho_{\alpha} \nu_{\alpha}} \tag{2.5.22}$$

is approximated by comparing the diffusive stabilization operator to a Chapman-Enskog expansion around a 13-moment distribution. In the limit of high collisionality the collisional diffusivity reduces to zero, allowing the relaxation BGK collision operator to describe thermal equilibration.

It is important to note that this operator is not intended to enforce the correct behavior approaching the collisionless limit as it drives the fluid to local thermal equilibrium where the heat flux and anisotropic pressure are diffused to zero. In practice the diffusivity is

clamped at a maximum value of

$$D_\alpha = \frac{P_\alpha}{(\nu_p \tau)\rho_\alpha \tilde{\nu}}, \tag{2.5.23}$$

where $\tilde{\nu}$ is the minimum collisionality allowed by the operator. Setting $\tilde{\nu} = 10$ applies an adequate amount of diffusion near the collisional transition regime. By limiting the collisionality in this operator the model retains stability for weakly collisionality systems, without driving strong, non-physical collisional transport. The combination of the BGK collision operator and the diffusive stabilization operator act to provide a proper representation of thermal equilibration within a species. The final component required to accurately describe scattering collisions within the 5-moment and 13-moment models is to include collisions between species.

2.5.3 Interspecies collisions

Scattering collisions between species have been well studied for 5-moment plasma models[11;25]. Generally, interspecies collisions drive the flow velocities and temperatures of the two species together. To represent the interspecies collisional effect, the collision operator C_α has multiple components,

$$C_\alpha = \sum_\beta C_{\alpha\beta}, \tag{2.5.24}$$

that describe the collisional effect on species α due to all other species β. Since scattering collisions are not reactive, they do not convert particles from one species to another, such that the impact of collisions on a plasma's mass, described in Eqs. (2.3.8) and (2.4.1), is given by

$$A_\alpha \langle C_\alpha \rangle = \sum_\beta A_\alpha \langle C_{\alpha\beta} \rangle = 0. \tag{2.5.25}$$

The resistive drag,

$$R_i^\alpha = \sum_\beta A_\alpha \left\langle w_i^\alpha C_{\alpha\beta} \right\rangle = \sum_\beta R_i^{\alpha\beta}, \tag{2.5.26}$$

drives the velocities of two species together and is approximated by[25]

$$R_i^{\alpha\beta} = -\nu_{\alpha\beta} A_\alpha n_\alpha u_i^{\alpha\beta} \tag{2.5.27}$$

where $\vec{u}_{\alpha\beta} = \vec{u}_\alpha - \vec{u}_\beta$. The collisional heating,

$$Q_{ij}^\alpha = \sum_\beta A_\alpha \left\langle w_i^\alpha w_j^\alpha C_{\alpha\beta} \right\rangle = \sum_\beta Q_{ij}^{\alpha\beta}, \tag{2.5.28}$$

is approximated by

$$Q_{ij}^{\alpha\beta} = -2\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(T_{ij}^{\alpha\beta} - A_\beta u_i^{\alpha\beta} u_j^{\alpha\beta} \right) \tag{2.5.29}$$

where $\mathbf{T}_{\alpha\beta} = \mathbf{T}_\alpha - \mathbf{T}_\beta$ is the difference in species' temperature $\mathbf{T}_\alpha = \mathbf{P}_\alpha / n_\alpha$. For 5-moment models, the isotropic heating

$$Q_\alpha = \frac{1}{3} \sum_\beta Q_{ii}^{\alpha\beta} \tag{2.5.30}$$

$$= -\frac{2}{3} \sum_\beta \nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(3T_{\alpha\beta} - A_\beta u_{\alpha\beta}^2 \right) \tag{2.5.31}$$

agrees with Hinton[25] in the low flow velocity limit. The temperature difference term acts to drive the internal energy of the species together while the mass weighting of the second term tends to drive the lower mass species more forcefully than higher mass species as is seen in Braginskii[11].

The change in heat flux due to collisions,

$$W_i^\alpha = \sum_\beta A_\alpha \left\langle w_i^\alpha w_\alpha^2 C_{\alpha\beta} \right\rangle = \sum_\beta W_i^{\alpha\beta}, \tag{2.5.32}$$

drives the flow of temperature together between species and is approximated

$$\begin{aligned} W_i^{\alpha\beta} = &-\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(\frac{q_i^\alpha}{n_\alpha} - \frac{q_i^\beta}{n_\beta} \right) \\ &+ \nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(3u_i^{\alpha\beta} T_{\alpha\beta} + 2u_j^{\alpha\beta} T_{ij}^{\alpha\beta} + \frac{3}{2} \left(A_\alpha - A_\beta \right) u_i^{\alpha\beta} u_{\alpha\beta}^2 \right), \end{aligned} \tag{2.5.33}$$

where $T_{\alpha\beta} = T_{ii}^{\alpha\beta}/3$.

Note that the definitions of the operators given in Eqs. (2.5.27), (2.5.29), and (2.5.33) are designed to balance the exchange of conserved moments between species. These conserved quantities are related to the primitive collision operators derived in Sec. 2.5. To properly enforce conservation, the momentum exchange is governed by

$$A_\alpha \left\langle v_i C_{\alpha\beta} \right\rangle = -A_\beta \left\langle v_i C_{\beta\alpha} \right\rangle \tag{2.5.34}$$

which, assuming zero mass exchange, implies

$$\vec{R}_{\alpha\beta} = -\vec{R}_{\beta\alpha}. \tag{2.5.35}$$

The energy exchange must obey

$$A_\alpha \left\langle v_i v_j C_{\alpha\beta} \right\rangle = -A_\beta \left\langle v_i v_j C_{\beta\alpha} \right\rangle \tag{2.5.36}$$

which can be written

$$Q_{ij}^{\alpha\beta} + u_i^\alpha R_j^{\alpha\beta} + u_j^\alpha R_i^{\alpha\beta} = -Q_{ij}^{\beta\alpha} - u_i^\beta R_j^{\beta\alpha} - u_j^\beta R_i^{\beta\alpha} \tag{2.5.37}$$

or more simply

$$Q_{ij}^{\alpha\beta} + Q_{ij}^{\beta\alpha} = -u_i^{\alpha\beta} R_j^{\alpha\beta} - u_j^{\alpha\beta} R_i^{\alpha\beta}. \tag{2.5.38}$$

The energy flux exchange conservation,

$$A_\alpha \left\langle v_i v^2 C_{\alpha\beta} \right\rangle = -A_\beta \left\langle v_i v^2 C_{\beta\alpha} \right\rangle \tag{2.5.39}$$

must take both the collisional heating and resistive drag into account. Combining the inter-species form of Eq. (2.5.8),

$$A_\alpha \left\langle v_i v^2 C_{\alpha\beta} \right\rangle = W_i^{\alpha\beta} + 3u_i^\alpha Q_{\alpha\beta} + 2u_j^\alpha Q_{ij}^{\alpha\beta} + 2u_i^\alpha u_j^\alpha R_j^{\alpha\beta} + u_\alpha^2 R_i^{\alpha\beta}, \tag{2.5.40}$$

with the relations in Eq. (2.5.35) and Eq. (2.5.38) gives the conservation requirement for the energy flux

$$W_i^{\alpha\beta} + W_i^{\beta\alpha} = -3u_i^{\alpha\beta} Q_{\alpha\beta} - 2u_j^{\alpha\beta} Q_{ij}^{\alpha\beta} - u_{\alpha\beta}^2 R_i^{\alpha\beta} - 2u_i^{\alpha\beta} u_j^{\alpha\beta} R_j^{\alpha\beta}. \tag{2.5.41}$$

These conservation requirements can be shown with the operators derived in this section using a property of the collision frequency,

$$A_\alpha n_\alpha \nu_{\alpha\beta} = A_\beta n_\beta \nu_{\beta\alpha}, \tag{2.5.42}$$

into account. Comparing the resistive drag between species gives

$$R_i^{\alpha\beta} = -A_\alpha n_\alpha \nu_{\alpha\beta} u_i^{\alpha\beta} = A_\beta n_\beta \nu_{\beta\alpha} u_i^{\beta\alpha} \tag{2.5.43}$$

$$= -R_i^{\beta\alpha} \tag{2.5.44}$$

which, as shown in Eq. (2.5.35), enforces the conservation of total momentum. A similar

relation is found for the heat exchange,

$$\begin{aligned} Q_{ij}^{\alpha\beta} &= -2\nu_{\alpha\beta}n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta}\left(T_{ij}^{\alpha\beta} - A_\beta u_i^{\alpha\beta}u_j^{\alpha\beta}\right) \\ &= -2\nu_{\beta\alpha}n_\beta \frac{A_\beta}{A_\alpha + A_\beta}\left(-T_{ij}^{\beta\alpha} - A_\beta u_i^{\beta\alpha}u_j^{\beta\alpha}\right) \\ &= 2\nu_{\beta\alpha}n_\beta \frac{A_\beta}{A_\alpha + A_\beta}\left(T_{ij}^{\beta\alpha} - A_\alpha u_i^{\beta\alpha}u_j^{\beta\alpha}\right) + 2\nu_{\beta\alpha}n_\beta A_\beta u_i^{\beta\alpha}u_j^{\beta\alpha} \\ &= -Q_{ij}^{\beta\alpha} + 2\nu_{\alpha\beta}n_\alpha A_\alpha u_i^{\alpha\beta}u_j^{\alpha\beta}, \end{aligned} \tag{2.5.45}$$

which is consistent with the conservation requirement in Eq. (2.5.38) given the definition of the resistive drag in Eq. (2.5.27). The heat flux exchange,

$$\begin{aligned} W_i^{\alpha\beta} &= -\nu_{\alpha\beta}n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta}\left(\frac{q_i^\alpha}{n_\alpha} - \frac{q_i^\beta}{n_\beta}\right) \\ &\quad + \nu_{\alpha\beta}n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta}\left(3u_i^{\alpha\beta}T_{\alpha\beta} + 2u_j^{\alpha\beta}T_{ij}^{\alpha\beta} + \frac{3}{2}\left(A_\alpha - A_\beta\right)u_i^{\alpha\beta}u_{\alpha\beta}^2\right) \\ &= -W_i^{\beta\alpha} + \nu_{\beta\alpha}n_\beta \frac{A_\beta}{A_\alpha + A_\beta}\left(3u_i^{\beta\alpha}T_{\beta\alpha} + 2u_j^{\beta\alpha}T_{ij}^{\beta\alpha} + \frac{3}{2}\left(A_\beta - A_\alpha\right)u_i^{\beta\alpha}u_{\beta\alpha}^2\right) \\ &\quad + \nu_{\alpha\beta}n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta}\left(3u_i^{\alpha\beta}T_{\alpha\beta} + 2u_j^{\alpha\beta}T_{ij}^{\alpha\beta} + \frac{3}{2}\left(A_\alpha - A_\beta\right)u_i^{\alpha\beta}u_{\alpha\beta}^2\right) \\ &= -W_i^{\beta\alpha} + 2\nu_{\alpha\beta}n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta}\left(3u_i^{\alpha\beta}T_{\alpha\beta} + 2u_j^{\alpha\beta}T_{ij}^{\alpha\beta}\right) \\ &\quad + 3\nu_{\alpha\beta}n_\alpha A_\alpha \frac{A_\alpha - A_\beta}{A_\alpha + A_\beta}u_i^{\alpha\beta}u_{\alpha\beta}^2, \end{aligned} \tag{2.5.46}$$

is shown to be conservative by plugging Eq. (2.5.29) and Eq. (2.5.27) into the conservation

requirement in Eq. (2.5.41) to give

$$\begin{aligned} W_i^{\alpha\beta} + W_i^{\beta\alpha} &= 2\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(3T_{\alpha\beta} - A_\beta u_{\alpha\beta}^2\right) u_i^{\alpha\beta} \\ &\quad + 4\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} u_j^{\alpha\beta} \left(T_{ij}^{\alpha\beta} - A_\beta u_i^{\alpha\beta} u_j^{\alpha\beta}\right) + 3\nu_{\alpha\beta} n_\alpha A_\alpha u_i^{\alpha\beta} u_{\alpha\beta}^2 \\ &= 2\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(3u_i^{\alpha\beta} T_{\alpha\beta} + 2u_j^{\alpha\beta} T_{ij}^{\alpha\beta}\right) \\ &\quad - 6\nu_{\alpha\beta} n_\alpha \frac{A_\alpha A_\beta}{A_\alpha + A_\beta} u_{\alpha\beta}^2 u_i^{\alpha\beta} + 3\nu_{\alpha\beta} n_\alpha A_\alpha u_i^{\alpha\beta} u_{\alpha\beta}^2 \\ &= 2\nu_{\alpha\beta} n_\alpha \frac{A_\alpha}{A_\alpha + A_\beta} \left(3u_i^{\alpha\beta} T_{\alpha\beta} + 2u_j^{\alpha\beta} T_{ij}^{\alpha\beta}\right) \\ &\quad + 3\nu_{\alpha\beta} n_\alpha A_\alpha \frac{A_\alpha - A_\beta}{A_\alpha + A_\beta} u_i^{\alpha\beta} u_{\alpha\beta}^2, \end{aligned} \tag{2.5.47}$$

which agrees with the expected form. Enforcing conservation in the collisional exchange operators ensures that no energy or energy flux is artificially generated, which can destabilize the model. Given the set of collision operators defined in Sec. 2.5 that describe scattering collisions both within and between species, the 5-moment and 13-moment models are complete.

2.5.4 Summary of collisional transport operators

The following is a summary of the 5-moment and 13-moment plasma models including all collisional transport operators. The evolution of the mass density for each species is given by

$$\frac{\partial}{\partial t}\rho_\alpha + \frac{\partial}{\partial x_i} p_i^\alpha = 0. \tag{2.5.48}$$

The evolution of the momentum density,

$$\frac{\partial}{\partial t} p_i^\alpha + \frac{\partial}{\partial x_j} e_{ij}^\alpha = Z_\alpha (\omega_p \tau)^2 n_\alpha E_i + Z_\alpha (\omega_c \tau) n_\alpha \epsilon_{ijk} u_j^\alpha B_k + (\nu_p \tau) \sum_\beta R_i^{\alpha\beta}, \tag{2.5.49}$$

is dependent on the resistive coupling between species $\vec{R}_{\alpha\beta}$, defined in Eq. (2.5.27). The evolution of the energy tensor,

$$\begin{aligned}\frac{\partial}{\partial t}e^{\alpha}_{ij}+\frac{\partial}{\partial x_k}H^{\alpha}_{ijk} &= Z_{\alpha}(\omega_p\tau)^2 n_{\alpha}\left(u^{\alpha}_i E_j+u^{\alpha}_j E_i\right)+\frac{Z_{\alpha}}{A_{\alpha}}(\omega_c\tau)\left(\epsilon_{ikl}e^{\alpha}_{jk}+\epsilon_{jkl}e^{\alpha}_{ik}\right)B_l \\ &\quad -(\nu_p\tau)\nu_{\alpha\alpha}\left(P^{\alpha}_{ij}-P_{\alpha}\delta_{ij}\right)+\frac{\partial}{\partial x_k}\left((\nu_p\tau)D_{\alpha}\frac{\partial}{\partial x_k}\left(P^{\alpha}_{ij}-P_{\alpha}\delta_{ij}\right)\right) \\ &\quad +(\nu_p\tau)\sum_{\beta}\left(Q^{\alpha\beta}_{ij}+u^{\alpha}_i R^{\alpha\beta}_j+u^{\alpha}_j R^{\alpha\beta}_i\right),\end{aligned} \tag{2.5.50}$$

includes the heating effects due to collisions between species $\mathbf{Q}_{\alpha\beta}$, defined in Eq. (2.5.29). The stabilization diffusivity D_{α} is given in Eq. (2.5.23). In the case of the 5-moment model, the evolution of the isotropic energy, $\epsilon_{\alpha}=e^{\alpha}_{ii}/2$, is given by

$$\frac{\partial}{\partial t}\epsilon_{\alpha}+\frac{1}{2}\frac{\partial}{\partial x_i}H^{\alpha}_i=Z_{\alpha}(\omega_p\tau)^2 n_{\alpha}u^{\alpha}_i E_i+(\nu_p\tau)\sum_{\beta}\left(\frac{3}{2}Q_{\alpha\beta}+u^{\alpha}_i R^{\alpha\beta}_i\right), \tag{2.5.51}$$

where the interspecies isotropic heating $Q_{\alpha\beta}=Q^{\alpha\beta}_{ii}/3$ is defined in Eq. (2.5.31). The evolution of the energy flux,

$$\begin{aligned}\frac{\partial}{\partial t}H^{\alpha}_i+\frac{\partial}{\partial x_j}G^{\alpha}_{ij} &= \frac{Z_{\alpha}}{A_{\alpha}}(\omega_p\tau)^2\left(e^{\alpha}_{jj}E_i+2e^{\alpha}_{ij}E_j\right)+\frac{Z_{\alpha}}{A_{\alpha}}(\omega_c\tau)\epsilon_{ijk}H^{\alpha}_j B_k \\ &\quad -2(\nu_p\tau)\nu_{\alpha\alpha}\left(q^{\alpha}_i+u^{\alpha}_i\left(P^{\alpha}_{ij}-P_{\alpha}\delta_{ij}\right)\right) \\ &\quad +\frac{\partial}{\partial x_k}\left(2(\nu_p\tau)D_{\alpha}\frac{\partial}{\partial x_k}\left(q^{\alpha}_i+u^{\alpha}_i\left(P^{\alpha}_{ij}-P_{\alpha}\delta_{ij}\right)\right)\right) \\ &\quad +(\nu_p\tau)\sum_{\beta}\left(W^{\alpha\beta}_i+3u^{\alpha}_i Q_{\alpha\beta}+2u^{\alpha}_j Q^{\alpha\beta}_{ij}+2u^{\alpha}_i u^{\alpha}_j R^{\alpha\beta}_j+u^2_{\alpha}R^{\alpha\beta}_i\right),\end{aligned} \tag{2.5.52}$$

contains the interspecies heat flux exchange $\vec{W}_{\alpha\beta}$ defined in Eq. (2.5.33). These equations serve to show the complexity in capturing collisional transport effects in higher moment models when transitioning out of thermal equilibrium. To complete the plasma model, Maxwell's equations describing the evolution of the electric and magnetic field must be included.

2.6 Maxwell's equations

In multi-species plasma models the charged particles interact with electromagnetic fields, as described in Sec. 2.1.1. While Ampere's law,

$$\frac{\partial}{\partial t}E_i = \left(\frac{c}{v_0}\right)^2 \frac{(\omega_c\tau)}{(\omega_p\tau)^2}\epsilon_{ijk}\frac{\partial}{\partial x_j}B_k - j_i, \tag{2.6.1}$$

with current density $\vec{j} = \sum_\alpha Z_\alpha n_\alpha \vec{u}_\alpha$, and Faraday's law,

$$\frac{\partial}{\partial t}B_i = -\frac{(\omega_p\tau)^2}{(\omega_c\tau)}\epsilon_{ijk}\frac{\partial}{\partial x_j}E_k, \tag{2.6.2}$$

expressed the time evolution of the electric and magnetic field, the Gauss's law constraints for the electric field,

$$\frac{\partial}{\partial x_i}E_i = \rho_c, \tag{2.6.3}$$

with charge density $\rho_c = \sum_\alpha Z_\alpha n_\alpha$, and magnetic field,

$$\frac{\partial}{\partial x_i}B_i = 0, \tag{2.6.4}$$

are preserved analytically by Eqs. (2.6.1) and (2.6.2) if they are satisfied initially. Numerical solutions to Maxwell's equations, however, can introduce deviations from Gauss's laws, characterized by the electric field divergence error

$$\Psi_E = \partial_{x_i}E_i - \rho_c \tag{2.6.5}$$

and magnetic field divergence error

$$\Psi_B = \partial_{x_i}B_i. \tag{2.6.6}$$

To prevent these errors from affecting the solution, Maxwell's equations are modified to control the divergence error growth rates. Two cleaning methods are used in this dissertation. The first method, known as the perfectly hyperbolic Maxwell's equations, uses a hyperbolic cleaning method making it ideal for conservative form numerical methods. A second model, known as the parabolically cleaned Maxwell's equations, is also implemented, which acts to clean divergence errors using a more localized approach.

2.6.1 Perfectly hyperbolic Maxwell's equations

The perfectly hyperbolic Maxwell's equations (PHMaxwell) is a common method for cleaning divergence errors from Maxwell's equations. The derivation of the method is beyond the scope of this dissertation and can be found in Munz et al.[34] and Kemm et al.[35]. The basis of the hyperbolic scheme is to represent the divergence errors through an electric field error potential ϕ_E and magnetic field error potential ϕ_B. The errors contained in these potentials are advected at speeds $\gamma_E\left(\frac{c}{v_0}\right)$ and $\gamma_B\left(\frac{c}{v_0}\right)$ through the domain until reaching a boundary, at which the errors leave the domain. This error potential equation for the electric field

$$\frac{\partial}{\partial t}\phi_E = -\gamma_E \Psi_E = -\gamma_E \frac{\partial}{\partial x_i} E_i + \gamma_E \rho_c \tag{2.6.7}$$

and magnetic field

$$\frac{\partial}{\partial t}\phi_B = -\gamma_B \Psi_B = -\gamma_B \frac{\partial}{\partial x_i} B_i \tag{2.6.8}$$

effectively describe the evolution of the error potentials as a decaying divergence errors. Ampere's law is modified to include the electric field error potential

$$\frac{\partial}{\partial t}E_i = \left(\frac{c}{v_0}\right)^2 \frac{(\omega_c \tau)}{(\omega_p \tau)^2} \epsilon_{ijk} \frac{\partial}{\partial x_j} B_k - j_i - \gamma_E \left(\frac{c}{v_0}\right)^2 \frac{\partial}{\partial x_i}\phi_E, \tag{2.6.9}$$

and Faraday's law is coupled to the magnetic field error potential

$$\frac{\partial}{\partial t}B_i = -\frac{(\omega_p\tau)^2}{(\omega_c\tau)}\epsilon_{ijk}\frac{\partial}{\partial x_j}E_k - \gamma_B\frac{\partial}{\partial x_i}\phi_B. \tag{2.6.10}$$

To apply cleaning to a multi-species plasma model, the values for the cleaning speeds must be $\gamma_E > 1$ and $\gamma_B > 1$. Since the cleaning speeds define the fastest characteristic velocities in a plasma, the upper bound on the cleaning speeds is dependent on desired runtime.

The strict hyperbolicity of the cleaning operator makes the model ideal for discontinuous numerical methods, however, the model has two drawbacks. The first drawback is that the divergence errors tied to the error potentials are coupled to their respective fields, which implies that as the error potentials are advected through the domain the divergence errors spread with them. This effectively pollutes the global solution with a locally generated divergence error at a speed greater than or equal to the speed of light. The second drawback associated with hyperbolic cleaning is that the error potentials require specific boundary conditions to remove the error from the domain. In the case of a system containing periodic boundary conditions, this option may not be available. For these reasons, a parabolic cleaning operator was implemented to clean the divergence errors associated with Maxwell's equations.

2.6.2 Parabolic cleaning of Maxwell's equations

Parabolically cleaned Maxwell's equations (PCMaxwell) are designed to locally remove divergence errors using a diffusion-style operator. The model is derived by taking the divergence of Ampere's and Faraday's laws to express the evolution of the divergence errors. The divergence of Ampere's law (Eq. (2.6.1))

$$\frac{\partial}{\partial t}\frac{\partial}{\partial x_i}E_i = -\frac{\partial}{\partial x_i}j_i \tag{2.6.11}$$

shows the relation between the divergence of the current density and the divergence of the electric field. The conservation of charge condition

$$\frac{\partial}{\partial t}\rho_c = -\frac{\partial}{\partial x_i} j_i \tag{2.6.12}$$

is used to modify Eq. (2.6.11) to describe the evolution of the divergence error

$$\frac{\partial}{\partial t}\frac{\partial}{\partial x_i} E_i = \frac{\partial}{\partial t}\rho_c \tag{2.6.13}$$

or in terms of the divergence error defined in Eq. (2.6.5)

$$\frac{\partial}{\partial t}\Psi_E = 0. \tag{2.6.14}$$

Taking the divergence of Faraday's law

$$\frac{\partial}{\partial t}\frac{\partial}{\partial x_i} B_i = 0 \tag{2.6.15}$$

gives a similar condition on the magnetic divergence error

$$\frac{\partial}{\partial t}\Psi_B = 0. \tag{2.6.16}$$

The conditions shown in Eqs. (2.6.14) and (2.6.16) state that the divergence error is constant in time analytically, however this is not necessarily the case for a numerical discretization. To counter the growth of the divergence error, an isotropic diffusion operator (divergence-of-gradient) is applied to the divergence error evolution equations such that

$$\frac{\partial}{\partial t}\Psi_E = \frac{\partial}{\partial x_i}\left(\chi_E \frac{\partial}{\partial x_i}\Psi_E\right) \tag{2.6.17}$$

and

$$\frac{\partial}{\partial t}\Psi_B = \frac{\partial}{\partial x_i}\left(\chi_B \frac{\partial}{\partial x_i}\Psi_B\right). \tag{2.6.18}$$

The spatially independent cleaning diffusivities χ_E and χ_B determine the rate of divergence error cleaning. Working backwards rebuilds Ampere's law

$$\frac{\partial}{\partial t}E_i = \left(\frac{c}{v_0}\right)^2 \frac{(\omega_c\tau)}{(\omega_p\tau)^2}\epsilon_{ijk}\frac{\partial}{\partial x_j}B_k - \left(j_i + \chi_E \frac{\partial}{\partial x_i}\rho_c\right) + \frac{\partial}{\partial x_i}\left(\chi_E \frac{\partial}{\partial x_j}E_j\right) \tag{2.6.19}$$

and Faraday's law

$$\frac{\partial}{\partial t}B_i = -\frac{(\omega_p\tau)^2}{(\omega_c\tau)}\epsilon_{ijk}\frac{\partial}{\partial x_j}E_k + \frac{\partial}{\partial x_i}\left(\chi_B \frac{\partial}{\partial x_j}B_j\right). \tag{2.6.20}$$

to include the parabolic cleaning. Note that instead of the divergence-of-gradient operator seen in Eqs. (2.6.17) and (2.6.17), the modified PCMaxwell equations contain a gradient-of-divergence operator. In practice, the cleaning diffusivities are not large, and should not affect the time step stability conditions for the numerical method. A consequence of this operator is that it is designed to clean short wavelength errors on the scale of the mesh resolution that arise around boundary conditions and charge density discontinuities.

Chapter 3

DISCONTINUOUS NUMERICAL METHODS

One of the objectives of this research is to implement practical, high accuracy numerical methods that run efficiently on large scale computing environments. Discontinuous numerical methods are computationally efficient and highly parallelizable making them advantageous for multi-species plasma modeling.

Discontinuous numerical methods, most notably finite volume methods[36] (FVM) and discontinuous Galerkin methods[37] (DGM), are ideal for working with balance law equation sets. Balance law equation sets are extensions of conservation laws which state that the change in a set of quantities $\vec{q}$ within an element depends on their flux $\mathbf{f}$ through the element's surface as well as any sources $\vec{s}$ within the element. The strong form of a balance law is

$$\frac{\partial}{\partial t} q_i + \frac{\partial}{\partial x_j} f_{ij} = s_i. \tag{3.0.1}$$

High-order accurate numerical discretizations of Eq. (3.0.1) generally separate the temporal component

$$\frac{\partial}{\partial t} q_i = \mathcal{L}_i \left(\vec{q}, t \right) \tag{3.0.2}$$

from the spatial component

$$\mathcal{L}_i \left(\vec{q}, t \right) = -\frac{\partial}{\partial x_j} f_{ij} + s_i \tag{3.0.3}$$

where the $\vec{\mathcal{L}}$ operator contains the spatial and temporal dependences of the equation set.

The spatial discretization is used to generate a high-order accurate evaluation of $\vec{\mathcal{L}}$, which is used in the time integration scheme used to solve Eq. (3.0.2). Two spatial discretization methods are implemented through this study which include a high-order finite volume method (HOFVM) and a discontinuous Galerkin method.

This chapter discusses the numerical methods implemented for this research in four sections. The discussion begins with an overview of discontinuous spatial discretizations in Sec. 3.1, which introduces concepts shared by both finite volume and discontinuous Galerkin methods. The finite volume method is discussed in Sec. 3.2, which is followed by the discontinuous Galerkin method in Sec. 3.3. The time integration schemes used for this study are examined in Sec. 3.4.

3.1 Discontinuous spatial discretization

Plasmas dynamics are dominated by hyperbolic fluxes and dispersive forces[38], meaning that the spatial discretization must be capable of accurately capturing wave dynamics. Hyperbolicity introduces the concept of locality, where the evolution of the solution depends only the local region based on how quickly the waves are propagating through the domain. Discontinuous numerical methods are ideal for hyperbolic systems exhibiting wave dynamics[36] as they are designed to enforce conservation which is required for hyperbolicity.

The spatially discretization of an equation set using discontinuous finite elements first requires the problem to be projected onto a mesh. A mesh is a representation of a physical domain, which has been broken down into a collection of individual elements or cells. Various kinds of elements can exist in a mesh for different geometries and dimensionalities, however for this study, only polygonal elements (i.e. element faces are flat) are allowed, which greatly simplifies the derivations for discontinuous methods. Figure 3.1.1 shows a one dimensional example of a discontinuous spatial discretization where the solution is considered continuous within each element, but is allowed to be discontinuous on the interfaces between elements.

The discontinuous interface solution leads to two major advantages. The first advantage is that allowing for discontinuous interfaces aids in the modeling of discontinuous systems such as the shock structures that arise from the nonlinear hyperbolic equation sets that make up plasma models. Discontinuous methods are also advantageous for parallelization schemes since each element and face can be treated individually.

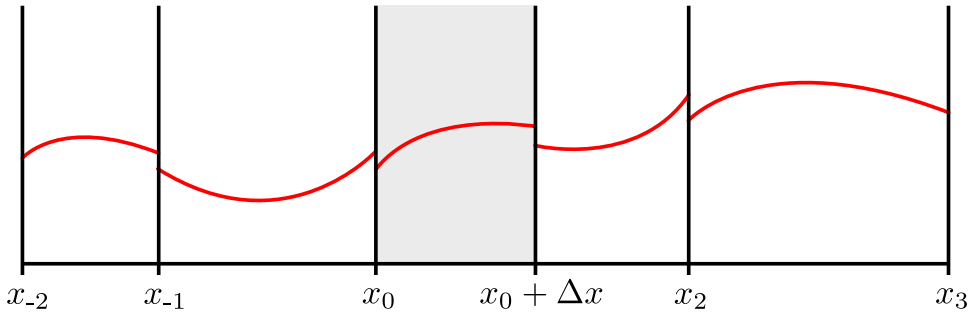


Figure 3.1.1: Discontinuous numerical methods attempt to spatially discretize a hyperbolic system of equations by representing solutions continuously (red line) within elements and allows discontinuous solutions between elements.

The spatial discretization approximates the solution $\vec{q} \approx \vec{q}_\lambda$ within each element λ, which is then expanded,

$$q_i^\lambda(\vec{x}, t) = q_{ij}^\lambda(t)\phi_j^\lambda(\vec{x}), \tag{3.1.1}$$

on a polynomial basis set $\vec{\phi}_\lambda$ which is dependent on the element geometry. Discontinuous numerical methods are a class of *hp* refinement methods[37;39] where the convergence accuracy order is dependent on a combination of the order of the polynomial basis and the element sizes. Increasing the resolution of the element discretization by breaking elements into smaller elements is known as *h* refinement, while increasing the order of the basis within the element is known as *p* refinement. A variety of bases may be used ranging from the orthogonal polynomial/fourier basis which are used in modal methods, and interpolation polynomial bases

used in nodal methods. In terms of accuracy these representations are equivalent[37], however, the computational expense between modal and nodal schemes can differ greatly.

In a finite volume method only the $\phi_0^\lambda = 1$ basis is stored, which implies that $q_{i,0}^\lambda$ is the integral of q_i over element λ. In order to reach higher-order convergence accuracies in a finite volume method, a high-order polynomial must be fit to the known solution's in a local set of elements. Discontinuous Galerkin methods store all of the modes representing the solution within an element, making them more compact, however this increases their complexity compared to finite volume methods. While the Galerkin method is more complex than the finite volume method, it generally reduces the numerical diffusion associated with discontinuous numerical methods for nonlinear equation sets containing disparate wave speeds[38].

Before deriving the finite volume and discontinuous Galerkin methods, it is important to overview the fundamental concepts of discontinuous numerical methods. These concepts are outlined by deriving the classic, first-order accurate finite volume discretization. The derivation begins by deriving the weak form of the balance law given in Eq. 3.0.1. Weak form representations are generated by multiplying the strong form equation set by a test function and taking the volume average over each element individually. Both low-order finite volume and disconitnuous Galerkin methods are examples of Galerkin finite element methods where the test functions are sampled from the basis functions $\vec{\phi}_\lambda$. Ignoring source terms, the weak form within each element λ, using the test function ϕ_k^λ, is then

$$\frac{\partial}{\partial t}\left(\frac{1}{V_\lambda}\int_\lambda q_i\phi_k^\lambda\,dV\right) = \frac{1}{V_\lambda}\int_\lambda \phi_j^\lambda\phi_k^\lambda\,dV\frac{\partial}{\partial t}q_{ij}^\lambda \tag{3.1.2}$$

$$= -\frac{1}{V_\lambda}\int_\lambda \phi_k^\lambda\frac{\partial}{\partial x_j}f_{ij}\,dV, \tag{3.1.3}$$

where V_λ is the element's volume. As noted earlier, only the first basis, $\phi_0^\lambda = 1$, is stored for

each element implying $i = j = 0$. Since $\phi_0^\lambda = 1$, the weak form can be rewritten as

$$\frac{\partial}{\partial t} Q_i^\lambda = -\frac{1}{V_\lambda} \int_\lambda \frac{\partial}{\partial x_j} f_{ij} \, dV \tag{3.1.4}$$

using the volume averaged solution

$$Q_i^\lambda = \frac{1}{V_\lambda} \int_\lambda q_i \, dV = q_{ij}^\lambda \frac{1}{V_\lambda} \int_\lambda \phi_j^\lambda \, dV. \tag{3.1.5}$$

Applying the divergence theorem to the flux integral

$$\int_\lambda \frac{\partial}{\partial x_j} f_{ij} \, dV = \oint_{\partial\lambda} n_j^\lambda f_{ij} \, dA (3.1.8) \tag{3.1.6}$$

leads to a total surface integral of the flux out of the element, where n_i^λ is the surface normal vector. Since the linear element has N_f flat faces which connect to a set of neighboring elements $\bar{\Gamma}_\lambda = [\gamma_0, \gamma_1, \ldots, \gamma_{N_f - 1}]$, the weak form simplifies to

$$\frac{\partial}{\partial t} Q_i^\lambda = -\sum_{\gamma \in \bar{\Gamma}_\lambda} \frac{1}{V_\lambda} \int_{\partial\lambda_\gamma} n_j^{\lambda\gamma} f_{ij} \, dA \tag{3.1.7}$$

where $n_i^{\lambda\gamma}$ is the outward pointing surface normal for the face connecting element λ to γ. Note that the flux must be single valued on the interface between elements to enforce conservation in discontinuous numerical methods.

3.1.1 Flux evaluation

Multiplying the flux tensor by the face normal, represented in Eq. (3.1.8), implies that only flux normal to the face is required. This effect is shown in Fig. 3.1.2 for a triangular mesh where the transverse flux moves $\vec{q}$ along the element's surface, but does not impact on the volume averaged solution within the element. This can be used to simplify the weak form

equation by rotating the problem from the global frame X to a local frame X' that is normal to the face. The flux is evaluated in the local frame, then rotated back to the global frame before being added in Eq. (3.1.7).

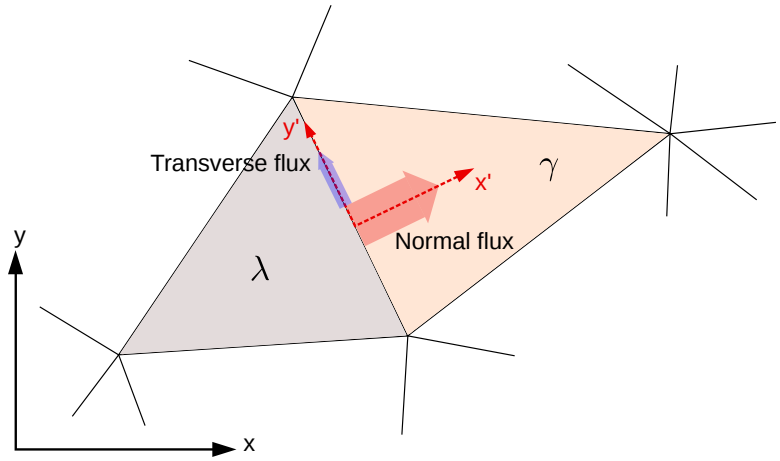


Figure 3.1.2: The flux from element λ to element γ can be described in terms of a normal flux (pink arrow) in a normal reference frame (dashed red). The unit vector for axis x' is the face normal vector. The transverse flux (blue) does not affect the volume averaged solution of λ or γ since the flux does not traverse the boundary.

The normal flux relation is written using rotation operators

$$n_j^{\lambda\gamma} f_{ij} = R^{-1}_{\lambda\gamma i}\left[\vec{F}\left(R_{\lambda\gamma}\left[\vec{q}\right]\right)\right]. \tag{3.1.8}$$

where $\vec{F}(\vec{q})$ is the normal flux vector (e.g. along the x' axis such that $F_i = f_{ix'}$). The rotation operator $R_{\lambda\gamma}\left[\vec{q}\right]$ rotates the solution vector $\vec{q}$ from the world frame X to the face normal reference frame X' that separates element λ from γ. The inverse rotation operator $R^{-1}_{\lambda\gamma}\left[\vec{q}\right]$ then rotates the solution from the face normal reference frame X' to the world frame X such

that

$$\vec{q} = R_{\lambda\gamma}^{-1}\left[R_{\lambda\gamma}\left[\vec{q}\right]\right]. \tag{3.1.9}$$

Note that the rotation operators take on the rank of its arguments, such that $R_{\lambda\gamma}\left[\vec{q}\right]$ is a vector and $R_{\lambda\gamma}\left[\mathbf{Q}\right]$ is a tensor.

The rotation operators are based on an orthogonal rotation matrix which is defined by the face normal $n_i^{\lambda\gamma}$ or $\hat{n}$ in short-hand. The rotation matrix,

$$\mathbf{R} = \begin{bmatrix} \hat{n}_x & \hat{n}_y & \hat{n}_z \\ \hat{t}_x & \hat{t}_y & \hat{t}_z \\ \hat{b}_x & \hat{b}_y & \hat{b}_z \end{bmatrix}, \tag{3.1.10}$$

with inverse

$$\mathbf{R}^{-1} = \begin{bmatrix} \hat{n}_x & \hat{t}_x & \hat{b}_x \\ \hat{n}_y & \hat{t}_y & \hat{b}_y \\ \hat{n}_z & \hat{t}_z & \hat{b}_z \end{bmatrix} = \mathbf{R}^T, \tag{3.1.11}$$

is written in terms of an arbitrary transverse vector $\hat{t}$ and binormal vector $\hat{b} = \hat{n} \times \hat{t}$. The arbitrary transverse vector $\hat{t}$ can be derived from the face normal vector $\hat{n}$ using a Gram-Schmidt process. The Gram-Schmidt process begins with creating an arbitrary vector $\vec{\tau}$ that is not parallel to $\hat{n}$, usually found using a conditional statement, and normalizing the components that are not parallel to $\hat{n}$

$$\hat{t} = \frac{\vec{\tau} - (\hat{n} \cdot \vec{\tau})\hat{n}}{|\vec{\tau} - (\hat{n} \cdot \vec{\tau})\hat{n}|}.$$

Rotating $\vec{q}$ is accomplished by breaking $\vec{q}$ into tensors of various rank and rotating each separately. For example, the 5-moment model is made up of three tensor moments, the

scalar density, the momentum vector, and the scalar energy. The solution for Maxwell's equations is made up of separate vectors for the electric and magnetic fields. The rotation operators for these tensors are as follows:

- Rank 0 tensors (scalars), such as density, are not rotated.
- Rank 1 tensors (vectors) $\vec{v}$, such as electric field or fluid momentum:

$$R\left[\vec{v}\right] = \mathbf{R} \cdot \vec{v} \tag{3.1.12}$$

$$R^{-1}\left[\vec{v}\right] = \mathbf{R}^{-1} \cdot \vec{v} \tag{3.1.13}$$

- Rank 2 tensors $\mathbf{M}$, such as the pressure tensor or energy density tensor:

$$R\left[\mathbf{M}\right] = \mathbf{R} \cdot \mathbf{M} \cdot \mathbf{R}^{\text{-1}} \tag{3.1.14}$$

$$R^{-1}\left[\mathbf{M}\right] = \mathbf{R}^{-1} \cdot \mathbf{M} \cdot \mathbf{R} \tag{3.1.15}$$

For the large systems of equations that make up plasma models, the rotation operation used to define the normal flux is advantageous over calculating the face flux in each direction and multiplying by the face normal.

Since the solution is discontinuous at interfaces between elements, the flux calculation is generally multivalued. Therefore, the method is modified to incorporate single-valued flux approximations, known as numerical fluxes, that help enforce stability and conservation.

3.1.2 Numerical fluxes

The use of a numerical flux to capture discontinuous effects between elements is known as the Godunov scheme[36], which postulates that systems involving discontinuities are well represented when the fluxes between elements are expressed as solutions to Riemann problems. Riemann problems describe the evolution of a hyperbolic conservation law across a discon-

tinuous interface separating two states. Figure 3.1.3 shows an example solution of a Riemann problem after a time increment Δt at an interface where the initial solution has a constant left and right state. In this example, the evolution mimics the density evolution of the Sod shock tube problem[40] for an ideal 5-moment model, where the Riemann solution is broken into three waves describing a shock wave, contact discontinuity, and a rarefaction wave. By solving for this solution at each element face the accurate representation of the under-resolved interface dynamics keeps discontinuous numerical methods stable and accurate.

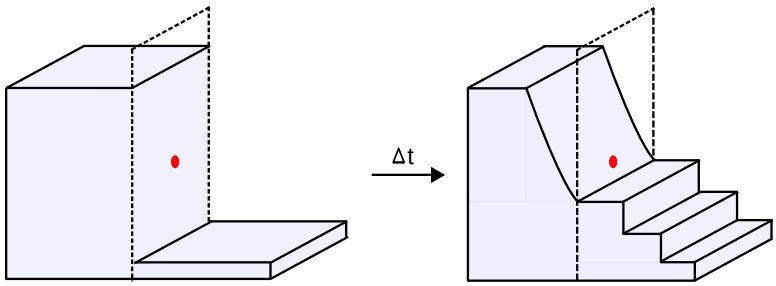


Figure 3.1.3: Solution to Riemann problem across a face (dashed line) for the fluid density in a Sod shock tube. High and low density states are initially separated and after a time increment Δt, the density evolves along three characteristic waves: the shock wave, contact discontinuity, and rarefaction wave. Solution to Riemann problem represents the amount of fluid density transfered across face.

The solutions to Riemann problems can be expressed in terms of the method of characteristics. The hyperbolicity of conservation form equation sets implies that the evolution of the solution depends on a set of characteristic wave speeds and paths represented by the eigenvalues and eigenvectors of the flux Jacobian

$$J_{ij}^{F} = \frac{\partial}{\partial q_j} F_i. \tag{3.1.16}$$

derived from the normal flux vector $\vec{F}$. If the exact characteristics are known, usually the case for linear equation sets, then the characteristic decomposition of the flux Jacobian leads

to an accurate approximation to the Riemann problem (see LeVeque[36]). If the characteristics cannot be derived in a closed form, or are computationally expensive to evaluate, an approximate Riemann solver may implemented instead. Approximate Riemann solvers tend to impart excessive numerical diffusion and/or dispersion into the system to help reinforce stability[36;37].

In the face normal reference frame X' of Fig. 3.1.2, the conservation law for each component of solution vector takes the form

$$\frac{\partial}{\partial t}\vec{q} + \frac{\partial}{\partial x'}\vec{F} = 0, \tag{3.1.17}$$

which can be rewritten using the flux Jacobian

$$\frac{\partial}{\partial t}\vec{q} + \mathbf{J}^F \cdot \frac{\partial}{\partial x'}\vec{q} = 0. \tag{3.1.18}$$

By taking the eigen decomposition of the flux Jacobian $\mathbf{J}^F = \tilde{\mathbf{\Lambda}} \cdot \tilde{\mathbf{\Delta}} \cdot \tilde{\mathbf{\Lambda}}^{-1}$ the conservation law is rewritten as

$$\frac{\partial}{\partial t}\vec{q} + \tilde{\mathbf{\Lambda}} \cdot \tilde{\mathbf{\Delta}} \cdot \tilde{\mathbf{\Lambda}}^{-1} \cdot \frac{\partial}{\partial x'}\vec{q} = 0. \tag{3.1.19}$$

By defining an eigen solution

$$\vec{w} = \tilde{\mathbf{\Lambda}}^{-1} \cdot \vec{q}, \tag{3.1.20}$$

and assuming the normal flux $\vec{F}$ is linear in $\vec{q}$, the conservation law becomes

$$\frac{\partial}{\partial t}\vec{w} + \tilde{\mathbf{\Delta}} \cdot \frac{\partial}{\partial x'}\vec{w} = 0 \tag{3.1.21}$$

which is a linear advection equation for the characteristic solution $\vec{w}$ with advective speeds given by the eigenvalues in the diagonal eigenvalue matrix $\tilde{\mathbf{\Delta}}$. The hyperbolicity of an

equation sets is dependent on the eigenvalues being real. Since the wave speeds can be both positive and negative, the solution $\vec{w}$ must be known on both sides of the interface to gauge an accurate solution to the Riemann problem. For the linear advection equation, the exact solution for each component w is given by

$$w_i(x', t + \Delta t) = w_i(x' - \tilde{\Delta}_{ii}\Delta t, t), \tag{3.1.22}$$

where $\tilde{\Delta}_{ii}$ is the i^{th} diagonal component of $\tilde{\mathbf{\Delta}}$. For an initially discontinuous system with an interface at x'_0, with constant $x' < x'_0$ state $\vec{w}^-$ and $x' > x'_0$ state $\vec{w}^+$, the solution at x'_0 is

$$w_i(x'_0, t + \Delta t) = \begin{cases} w_i^- & \text{if } \tilde{\Delta}_{ii} > 0 \\ w_i^+ & \text{if } \tilde{\Delta}_{ii} < 0 \end{cases} \tag{3.1.23}$$

or

$$w_i(x'_0, t + \Delta t) = w_i^- \frac{\tilde{\Delta}_{ii} + |\tilde{\Delta}_{ii}|}{2} + w_i^+ \frac{\tilde{\Delta}_{ii} - |\tilde{\Delta}_{ii}|}{2}. \tag{3.1.24}$$

Taking the eigen decomposition into account, the components of the solution vector at the interface are

$$q_i(x'_0, t + \Delta t) = \tilde{\Lambda}_{ij} w_j^- \frac{\tilde{\Delta}_{jj} + |\tilde{\Delta}_{jj}|}{2} + \tilde{\Lambda}_{ij} w_j^+ \frac{\tilde{\Delta}_{jj} - |\tilde{\Delta}_{jj}|}{2} \tag{3.1.25}$$

or, when explicitly containing the flux and the $\vec{q}_\pm$ states,

$$F_i(x'_0, t + \Delta t) = \frac{1}{2}\left(F_i(\vec{q}_-) + F_i(\vec{q}_+)\right) - \frac{1}{2}\tilde{\Theta}_{ij}\left(q_j^+ - q_j^-\right) \tag{3.1.26}$$

where $q_i^\pm = \tilde{\Lambda}_{ij} w_j^\pm$, $F_i(q^\pm) = \tilde{\Lambda}_{ij}\tilde{\Delta}_{jk} w_k^\pm$ and $\tilde{\Theta}_{ij} = \tilde{\Lambda}_{ik}|\tilde{\Delta}_{kl}|\tilde{\Lambda}^{-1}_{lj}$. The general form of this

solution is known as the numerical flux $\vec{\mathcal{F}}(\vec{q}^{-}, \vec{q}^{+})$ which is defined as

$$\vec{\mathcal{F}}(\vec{q}^{-}, \vec{q}^{+}) = \vec{F}(x_0', t + \Delta t) = \frac{1}{2}\left(\vec{F}(\vec{q}_{-}) + \vec{F}(\vec{q}_{+})\right) - \frac{1}{2}\tilde{\mathbf{\Theta}} \cdot (\vec{q}_{+} - \vec{q}_{-})\,. \tag{3.1.27}$$

For nonlinear systems, the solution becomes more complicated and can be approximated using a Roe solver[36]. In terms of the flux evaluation method outlined in Sec. 3.1.1, the numerical flux is implemented as

$$n_j^{\lambda\gamma} f_{ij} = R_{\lambda\gamma}^{-1}\left[\mathcal{F}\left(R_{\lambda\gamma}\left[\bar{q}_\lambda\right], R_{\lambda\gamma}\left[\bar{q}_\gamma\right]\right)\right] \tag{3.1.28}$$

where $\bar{q}_\lambda$ is the internal $(-)$ face solution and $\bar{q}_\gamma$ is the external $(+)$ solution. Combining this with Eq. (3.1.7) results in the general finite volume discretization for the evolution of the volume average solution

$$\frac{\partial}{\partial t} Q_i^\lambda = -\sum_{\gamma \in \bar{\Gamma}_\lambda} \frac{1}{V_\lambda} \int\limits_{\partial\lambda_\gamma} R_{\lambda\gamma}^{-1}\left[\mathcal{F}\left(R_{\lambda\gamma}\left[\bar{q}_\lambda\right], R_{\lambda\gamma}\left[\bar{q}_\gamma\right]\right)\right] dA. \tag{3.1.29}$$

For first-order finite volume methods, the fluxes are evaluated assuming a uniform solution in each element ($q_i^\lambda \approx Q_i^\lambda$) which simplifies Eq. (3.1.29) to

$$\frac{\partial}{\partial t} Q_i^\lambda = -\sum_{\gamma \in \bar{\Gamma}_\lambda} \frac{A_{\lambda\gamma}}{V_\lambda} R_{\lambda\gamma}^{-1}\left[\mathcal{F}\left(R_{\lambda\gamma}\left[Q_\lambda\right], R_{\lambda\gamma}\left[Q_\gamma\right]\right)\right]. \tag{3.1.30}$$

The definition of the numerical flux $\mathcal{F}$ is dependent on the equation set being solved. While the low order accuracy and excessive diffusion of the above method makes it impractical for plasma research, the concepts of characteristic decomposition, numerical fluxes, and rotation operations extend to all high-order discontinuous numerical methods.

3.2 High-order finite volume method

Finite volume methods are a weak form approximation of the strong form balance law given in Eq. (3.0.1), where only solution's volume average is stored and evolved. The high-order finite volume method (HOFVM) discussed herein follows the derivation given in Sec. 3.1 with an improved representation of the solution in the element and at the interfaces $\vec{q}_-$ and $\vec{q}_+$. The general form for the finite volume discretization in Eq. (3.1.29) is rewritten as

$$\frac{\partial}{\partial t}Q_i^\lambda = -\sum_{\gamma\in\bar{\Gamma}_\lambda}\frac{1}{V_\lambda}\int_{\partial\lambda_\gamma} R_{\lambda\gamma}^{-1}\left[\mathcal{F}\left(R_{\lambda\gamma}\left[\bar{q}_\lambda\right], R_{\lambda\gamma}\left[\bar{q}_\gamma\right]\right)\right] dA + \int_\lambda s_i\, dV \tag{3.2.1}$$

which includes source term integration. The finite volume method used for this research used quadrature integration to evaluate the surface and volume integrals in Eq. (3.2.1). Surface integration is handled by evaluating the flux at a set of quadrature points $\vec{x}_k^{\lambda\gamma}$ using surface quadrature weights $\omega_k^{\lambda\gamma}$

$$\int_{\partial\lambda_\gamma} g\, dA = \sum_k \omega_k^{\lambda\gamma} g\left(\vec{x}_k^{\lambda\gamma}\right) \tag{3.2.2}$$

Volume integration is handled with a different set of quadrature points $\vec{x}_k^\lambda$ and associated volume quadrature weights Ω_k^λ

$$\int_{\lambda_\gamma} g\, dV = \sum_k \Omega_k^\lambda g\left(\vec{x}_k^\lambda\right) \tag{3.2.3}$$

The surface and volume quadrature points and weights used can be found in Appendix A. Applying these quadrature methods to Eq. (3.2.1) results in

$$\begin{aligned}\frac{\partial}{\partial t}Q_i^\lambda = &-\sum_{\gamma\in\bar{\Gamma}_\lambda} R_{\lambda\gamma}^{-1}\left[\sum_k \frac{\omega_k^{\lambda\gamma}}{V_\lambda}\mathcal{F}\left(R_{\lambda\gamma}\left[q_\lambda\left(\vec{x}_k^{\lambda\gamma}\right)\right], R_{\lambda\gamma}\left[q_\gamma\left(\vec{x}_k^{\lambda\gamma}\right)\right]\right)\right] \\ &+\sum_k \frac{\Omega_k^\lambda}{V_\lambda} s_i\left(\vec{x}_k^\lambda\right)\end{aligned} \tag{3.2.4}$$

The complicated closed surface integral over a flux tensor in Eq. (3.0.1) has now been deconstructed into a summation and rotation operation acting on a numerical flux. Everything discussed up to this point can be accomplished to arbitrary orders of accuracy depending on the quadrature order and the accuracy of $\vec{q}$ and $\vec{s}$ at the quadrature points. A reconstruction scheme is needed in order to accurately evaluate the fluxes and source terms at the quadrature points. For this study, the k-exact reconstruction method[41;42] is used.

3.2.1 k-exact reconstruction

Evaluating the fluxes and sources accurately in a finite volume method requires an accurate evaluation of q within and on the faces of an element. The only knowledge of the solution in the domain is the volume averaged solutions Q. In order to generate an accurate form of q within an element, it must be reconstructed from the surrounding solution. This study uses the k-exact polynomial reconstruction method[41;42] for its simplicity. The idea behind k-exact reconstruction is shown in Fig. 3.2.1 where a continuous form of q is fitted to the volume averaged quantities Q in a stencil surrounding the element of interest.

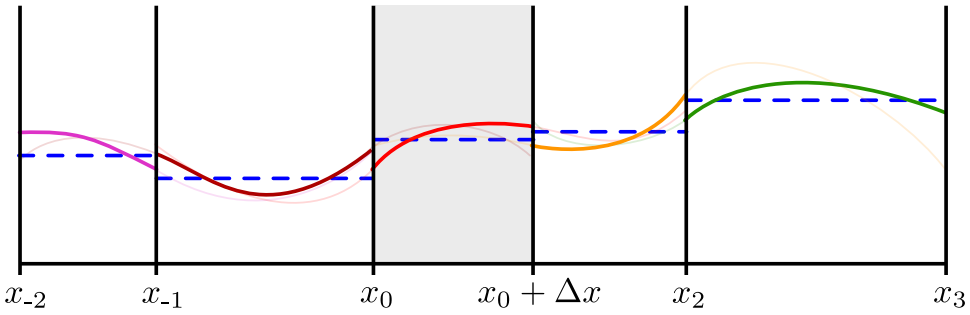


Figure 3.2.1: High-order finite volume methods reconstruct a continuous solution within each element (dark lines) using the volume average solutions (dashed blue lines) of the surrounding elements. The volume average of each element's continuous solution in neighboring elements (light lines) must be fitted to the corresponding neighbor's volume average solution.

The k-exact reconstruction method locally reconstructs the solution $q \approx q_\lambda$ within an element λ. The local reconstruction q_λ is expanded on the monomial basis which is centered at the element's geometric centroid $\vec{x}_\lambda$. The reconstruction takes the form

$$q_\lambda = \sum_{i,j,k=0}^{i+j+k \leq p} D_{ijk}^\lambda (x - x_\lambda)^i (y - y_\lambda)^j (z - z_\lambda)^k \tag{3.2.5}$$

where D_{ijk}^λ are the polynomial fitting coefficients and p is the maximal order of the polynomial. The continuous solution q is related to the reconstruction by

$$q = q_\lambda + \mathcal{O}\left((x - x_\lambda)^{p+1}, (y - y_\lambda)^{p+1}, (z - z_\lambda)^{p+1}, (x - x_\lambda)^p (y - y_\lambda), \ldots\right) \tag{3.2.6}$$

which defines the order of accuracy of the reconstruction. The reconstruction can be directly related to the volume averaged solution Q_λ using

$$Q_\lambda = \frac{1}{V_\lambda} \int_\lambda q_\lambda dV = \sum_{i,j,k=0}^{i+j+k \leq p} D_{ijk}^\lambda B_{ijk}^\lambda \tag{3.2.7}$$

where

$$B_{ijk}^{\lambda} = \frac{1}{V_\lambda} \int_{\lambda} (x - x_\lambda)^i (y - y_\lambda)^j (z - z_\lambda)^k \, dV \tag{3.2.8}$$

are geometric coefficients that depend only on the node positions that make up element λ. This element local reconstruction q_λ can also be integrated within an alternate element γ to compare against Q_γ. The result is

$$\begin{aligned} Q_\gamma &= \frac{1}{V_\gamma} \int_{\gamma} q_\lambda \, dV + \rho_{\lambda\gamma} \\ &= \frac{1}{V_\gamma} \sum_{i,j,k=0}^{i+j+k \leq p} D_{ijk}^{\lambda} \int_{\gamma} (x - x_\lambda)^i (y - y_\lambda)^j (z - z_\lambda)^k \, dV + \rho_{\lambda\gamma} \\ &= \sum_{i,j,k=0}^{i+j+k \leq p} D_{ijk}^{\lambda} C_{ijk}^{\gamma\lambda} + \rho_{\lambda\gamma} \end{aligned} \tag{3.2.9}$$

where $\rho_{\lambda\gamma}$ is a residual (error) to the fitting. Here an offset geometry coefficient

$$\begin{aligned} C_{ijk}^{\gamma\lambda} &= \frac{1}{V_\gamma} \int_{\gamma} (x - x_\lambda)^i (y - y_\lambda)^j (z - z_\lambda)^k \, dV \\ &= \frac{1}{V_\gamma} \int_{\gamma} \left((x - x_\gamma) + (x_\gamma - x_\lambda)\right)^i \left((y - y_\gamma) + (y_\gamma - y_\lambda)\right)^j \left((z - z_\gamma) + (z_\gamma - z_\lambda)\right)^k \, dV \\ &= \sum_{l=0}^{l \leq i} \sum_{m=0}^{m \leq j} \sum_{n=0}^{n \leq k} \binom{i}{l} \binom{j}{m} \binom{k}{n} (x_\gamma - x_\lambda)^l (y_\gamma - y_\lambda)^m (z_\gamma - z_\lambda)^n B_{i-l,j-m,k-n}^{\gamma} \end{aligned} \tag{3.2.10}$$

has been introduced using binomial expansion to relate $C_{ijk}^{\gamma\lambda}$ to the known B_{ijk}^{γ} and element centroids $\vec{x}_\lambda$ and $\vec{x}_\gamma$. By definition $B_{000}^{\lambda} = C_{000}^{\gamma\lambda} = 1$, which, when combined with Eq. (3.2.7),

gives the solution at the element centroid

$$q_\lambda\left(\vec{x}_\lambda\right) = D_\lambda^{000} = Q_\lambda - \sum_{\substack{i,j,k=0 \\ i+j+k\neq 0}}^{i+j+k\leq p} D_{ijk}^\lambda B_{ijk}^\lambda. \tag{3.2.11}$$

This implies that only the volume averaged differences are required for reconstructing D_λ^{abc}

$$Q_\gamma - Q_\lambda = \sum_{\substack{i,j,k=0 \\ i+j+k\neq 0}}^{i+j+k\leq p} D_{ijk}^\lambda \left(C_{ijk}^{\gamma\lambda} - B_{ijk}^\lambda\right) + \rho_{\lambda\gamma} \tag{3.2.12}$$

or in vector form

$$Q_\gamma - Q_\lambda = \vec{\mathcal{L}}_{\gamma\lambda} \cdot \vec{D}_\lambda + \rho_{\gamma\lambda}. \tag{3.2.13}$$

In general, the closer two elements are to one another, the more important they are for the reconstruction process. To enforce this, an element weighting scheme is applied based on the distance between the two elements $|\vec{x}_\lambda - \vec{x}_\gamma|$. The weighting is applied to Eq. (3.2.13) as

$$\Delta Q_{\gamma\lambda} = \frac{Q_\gamma - Q_\lambda}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} = \frac{\vec{\mathcal{L}}_{\gamma\lambda}}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} \cdot \vec{D}_\lambda + \frac{\rho_\lambda^\gamma}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} = \vec{L}_{\lambda\gamma} \cdot \vec{D}_\lambda + r_{\lambda\gamma} \tag{3.2.14}$$

where $r_{\lambda\gamma}$ is the weighted residual and n is a weight scaling factor ($n = 2$ works well). Since the rank 3 tetrahedral tensor representation of the L and D factors can have arbitrary maps to the vectors $\vec{L}_{\lambda\gamma}$ and $\vec{D}_\lambda$, their exact structure is application dependent. In general $\vec{L}_{\lambda\gamma}$

and $\vec{D}_\lambda$ can be written as

$$\frac{Q_\gamma - Q_\lambda}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} = \left[\begin{array}{cccc} \frac{C_{\gamma\lambda}^{100} - B_\lambda^{100}}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} & \frac{C_{\gamma\lambda}^{200} - B_\lambda^{200}}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} & \cdots & \frac{C_{\gamma\lambda}^{00k} - B_\lambda^{00k}}{|\vec{x}_\lambda - \vec{x}_\gamma|^n} \end{array} \right] \cdot \left[\begin{array}{c} D_\lambda^{100} \\ D_\lambda^{200} \\ \vdots \\ D_\lambda^{00k} \end{array} \right] + r_{\gamma\lambda}. \tag{3.2.15}$$

Now consider a set of N_s elements $\Gamma_\lambda = \{\gamma_0, \gamma_1 \ldots \gamma_{N_s\text{-}1}\}$ where $\lambda \notin \Gamma_\lambda$. Γ_λ is known as the stencil of element λ and represents the set of elements the reconstruction is fitted to. Applying the fitting to all stencil elements gives

$$\underbrace{\left[\begin{array}{c} \frac{Q_{\gamma_0} - Q_\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_0}\right|^n} \\ \frac{Q_{\gamma_1} - Q_\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_1}\right|^n} \\ \vdots \\ \frac{Q_{\gamma_{N_s\text{-}1}} - Q_\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_{N_s\text{-}1}}\right|^n} \end{array} \right]}_{\Delta\vec{Q}_\lambda} = \underbrace{\left[\begin{array}{cccc} \frac{C_{100}^{\gamma_0\lambda} - B_{100}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_0}\right|^n} & \frac{C_{200}^{\gamma_0\lambda} - B_{200}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_0}\right|^n} & \cdots & \frac{C_{00p}^{\gamma_0\lambda} - B_{00p}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_0}\right|^n} \\ \frac{C_{100}^{\gamma_1\lambda} - B_{100}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_1}\right|^n} & \frac{C_{200}^{\gamma_1\lambda} - B_{200}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_1}\right|^n} & \cdots & \frac{C_{00p}^{\gamma_1\lambda} - B_{00p}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_1}\right|^n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{C_{100}^{\gamma_{N_s\text{-}1}\lambda} - B_{100}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_{N_s\text{-}1}}\right|^n} & \frac{C_{200}^{\gamma_{N_s\text{-}1}\lambda} - B_{200}^\lambda}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_{N_s\text{-}1}}\right|^n} & \cdots & \frac{C_{00p}^{\gamma_{N_s\text{-}1}\lambda} - B_\lambda^{00p}}{\left|\vec{x}_\lambda - \vec{x}_{\gamma_{N_s\text{-}1}}\right|^n} \end{array} \right]}_{\mathbf{L}_\lambda} \cdot \underbrace{\left[\begin{array}{c} D_{100}^\lambda \\ D_{200}^\lambda \\ \vdots \\ D_{00p}^\lambda \end{array} \right]}_{\vec{D}_\lambda} + \underbrace{\left[\begin{array}{c} r_{\lambda\gamma_0} \\ r_{\lambda\gamma_1} \\ \vdots \\ r_{\lambda\gamma_{N_s-1}} \end{array} \right]}_{\vec{r}_\lambda} \tag{3.2.16}$$

or in matrix form

$$\Delta\vec{Q}_\lambda = \mathbf{L}_\lambda \cdot \vec{D}_\lambda + \vec{r}_\lambda. \tag{3.2.17}$$

The length of vector $\Delta\vec{Q}_\lambda$ is given by the stencil size N_s, while the length of vector $\vec{D}_\lambda$ is N_D, making $\mathbf{L}_\lambda$ a $N_D \times N_s$ matrix. In general the system should be overdetermined ($N_s > N_D$) and a perfect fitting does not exist ($|\vec{r}_\lambda| \neq 0$). In order to minimize the residual $\vec{r}_\lambda$ a least squares method is applied with singular value decomposition $\mathbf{L}_\lambda = \mathbf{U}_\lambda \cdot \mathbf{S}_\lambda \cdot \mathbf{V}_\lambda^T$ (note that $\mathbf{L}_\lambda$ is real making $\mathbf{U}_\lambda$ and $\mathbf{V}_\lambda$ orthogonal matrices)

$$\vec{D}_\lambda \doteq \mathbf{V}_\lambda \cdot \mathbf{S}_\lambda \cdot \mathbf{U}_\lambda^T \cdot \Delta\vec{Q}_\lambda = \tilde{\mathbf{L}}_\lambda^{\text{-}1} \cdot \Delta\vec{Q}_\lambda \tag{3.2.18}$$

where $\doteq$ means equivalent in the least squares sense. Note that for high-order reconstructions on unstructured meshes, $\mathbf{L}_\lambda$ can be poorly conditioned and care must be taken when truncating the singular values for the pseudoinverse $\tilde{\mathbf{L}}_\lambda^{-1}$. Given the dimensionality of the problem N_d and the reconstruction polynomial order k, N_D is given by[41;42]

$$N_D = \left[\frac{1}{N_d!} \prod_{i=1}^{i \leq N_d} (k+i) \right] - 1. \tag{3.2.19}$$

For this study, the implementation of the k-exact reconstruction scheme is setup as follows:

1. Initialization:
   (a) Generate and store B_{ijk}^{λ} (Eq. (3.2.8)) for all elements in domain. This study used numerical quadrature to evaluate B_{ijk}^{λ}.
   (b) Generate and store stencils Γ_λ for all elements in domain. For this study a recursive path finding algorithm was developed which jumps from each element to its neighbors to its neighbor's neighbors until the stencil is filled.
   (c) For each element λ with stencil Γ_λ:
      i. Calculate and store weights $w_{\gamma\lambda} = |\vec{x}_\lambda - \vec{x}_\gamma|^{-n}$ for each $\gamma \in \Gamma_\lambda$.
      ii. Calculate $C_{ijk}^{\gamma\lambda}$ (Eq. (3.2.10)) for each $\gamma \in \Gamma_\lambda$.
      iii. Calculate weighted geometry matrix $\mathbf{L}_\lambda$ as posed in Eq. (3.2.16).
      iv. Generate SVD of $\mathbf{L}_\lambda$
      v. Truncate singular values to zero below some threshold (10^{-10} below largest singular value works well for double precision).
      vi. Generate and store pseudoinverse $\tilde{\mathbf{L}}_\lambda^{-1}$.
2. Simulation:

(a) For each element λ:

i. Load stencil Γ_λ, weights $w_{\lambda\gamma}$ and pseudoinverse $\tilde{\mathbf{L}}_\lambda^{-1}$.

ii. For each variable q_i:

A. Generate $\Delta\vec{Q}_\lambda$ from $\Delta Q_{\gamma\lambda} = w_{\gamma\lambda}\left(Q_\gamma - Q_\lambda\right)$ for $\gamma \in \Gamma_\lambda$.

B. Find the differential reconstruction coefficients terms $\vec{D}_\lambda = \tilde{\mathbf{L}}_\lambda^{-1} \cdot \Delta\vec{Q}_\lambda$.

C. Find the cell centered value D_{000}^{λ} using Eq. (3.2.11).

D. Find q_i at any point in element using Eq. (3.2.5).

3.2.2 Central essentially non-oscillatory method

First-order finite volume methods are stable as long as the stability requirement of the time integration scheme (e.g. Courant number) is fulfilled. The reason for this stability is that the first order finite volume scheme is highly diffusive, which inherently eliminates any stability violating oscillations that could develop. When moving to higher-order discontinuous methods, the amount of numerical dissipation is decreased, and spurious oscillations are able to grow and destabilize the numerical method.

Stability in discontinuous methods is dependent on enforcing monotonicity. Monotonicity in hyperbolic systems implies that a system transitioning from one state to another does so without developing any extrema in the process. In other words, monotonicity enforcement requires that the hyperbolic evolution of an initially monotonic solution remain monotonic[43]. When monotonic evolution is violated, spurious oscillations form, which can destabilize numerical methods. There are both linear and nonlinear methods for enforcing monotonicity.

Linear monotonicity enforcement schemes are where the solution q_λ in an element is replaced by a piecewise linear approximation. Total variation diminishing/bounding (TVD/B) lim-

iters are two such instances that generate a linear approximation by limiting the slope of q_λ such as to locally introduce enough dissipation to reduce spurious oscillations. In general, these schemes are excessively diffusive, and drop the order of accuracy around the discontinuity to between first and second order accuracy[41;42;44]. While many such linear limiting methods exist, most tend to degrade extrema, as the methods do not differentiate between an extrema and a discontinuity.

Nonlinear monotonicity enforcement schemes, such as weighted essentially non-oscillatory methods (WENO), are less stringent than TVD/B and do not strictly eliminate spurious oscillations. The goal is to balance the removal of spurious oscillations with the degradation in accuracy around discontinuities. The accuracy degradation is generally less in nonlinear limiting schemes, but is dependent on the mesh geometry. More importantly, nonlinear monotonicity enforcement schemes preserve extrema.

Enforcing monotonicity is difficult on unstructured meshes. Most limiting schemes are developed for rectilinear grids and lose their effectiveness when moving to a general geometry system. An effective means of decreasing spurious oscillations in reconstruction based finite volume methods is to oversample the dataset. Oversampling implies that the number of elements making up the reconstruction stencil is larger than the number of reconstruction coefficients ($N_s > N_D$). This imposes low pass filtering effect, which appears to be monotonicity enforcing, however, it lowers the resolution and accuracy of the solver for mesh scale dynamics. This implies that $N_s \gtrsim N_D$ is an accuracy constraint. While there is no formal requirement, McDonald[42] and Ivan[41] have found that symmetric stencils should at least include the element's nearest neighbors. For this study, the number of stencil elements is based on a combination of the number of reconstruction coefficients and the number of faces per element

$$N_s = \max(N_D + 1, N_f). \tag{3.2.20}$$

The only real control over generating a stable and accurate reconstruction is the composition of the reconstruction stencil. Ideally the stencil is made up of a symmetric grouping of the nearest surrounding elements; however, in the presence of discontinuous solutions, this reconstruction introduces oscillations. To avoid this, most HOFVMs employ a combination of stencils to generate the least oscillatory reconstruction. These multi-stencil schemes are collectively known as essentially non-oscillatory (ENO) style schemes and are subdivided into strictly ENO schemes and weighted ENO (WENO) schemes. For strictly ENO schemes, the reconstructions from the various stencils are compared and the smoothest reconstruction is chosen. WENO schemes weigh the different reconstructions together based on the reconstructed polynomial's smoothness, the stencil shape, and the stencil location to create the best possible reconstruction.

ENO style schemes are not strictly total variation diminishing (TVD) or bounded (TVB), however they roughly produce monotonic results. Many ENO style schemes exist, WENO being the most popular, and while they retain high-order accuracy, the generation and analysis of multiple reconstructions requires a large computational effort. In order to decrease the computational expense, a hybrid ENO scheme was selected for this study, known as the central essentially non-oscillatory (CENO) scheme, which uses high-order reconstructions for smoothly varying solutions, and reduces to a faster TVD reconstruction (limited reconstruction) when oscillatory behavior is expected[41;42]. The CENO method combines the best of both ENO schemes with their high-order accuracy and extrema preservation[41;42], and TVD schemes for speed and stability. The method is designed to quickly evaluate high-order accurate solutions with sharp interfaces around shocks and discontinuities.

The CENO scheme only requires two stencils as depicted in Fig. 3.2.2: a low-order and high-order stencil. The high-order stencil is used to generate a high-order accurate reconstruction of the continuous solution within an element. This high-order accurate reconstruction represents the maximal order of accuracy achievable by the HOFVM. Where the solution is discontinuous, a TVD limited reconstruction is used, based on the smaller low-order stencil

which is designed to reconstruct a $p = 1$ or a second-order accurate polynomial reconstruction.

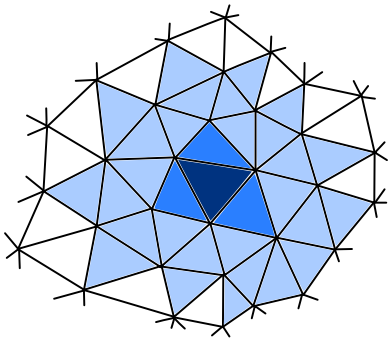


Figure 3.2.2: Example CENO stencils for central element (dark blue). High-order $p = 6$ stencil (medium+light blue) and low-order $p = 1$ stencil (medium blue). If the high-order stencil's reconstruction is not smooth, then the low-order stencil is used instead.

For a given element λ, the criteria for using the limited versus high-order reconstruction is based on a smoothness indicator denoted by $\mathcal{S}_\lambda$. Smoothness indicators do not define a specific method for measuring smoothness, and are generally complicated for non-rectilinear meshes. The goal of a smoothness indicator is to quickly generate a rough guess of the least squares residual magnitude of the reconstruction. The smaller $\mathcal{S}_\lambda$ is, the better the fit, and more "smooth" the reconstruction is. When $\mathcal{S}_\lambda$ is above some threshold $\mathcal{S}$, then the high-order reconstruction is replaced with the limited reconstruction.

The two reconstructions are outlined as follows:

- **Limited reconstruction**:

  The low-order, TVD limited solution is generated from a piecewise linear ($p = 1$) reconstruction, usually based on a stencil of the nearest N_f neighbors. Without the limiting procedure, the second order reconstruction also generates spurious oscillations around discontinuities.

When monotonicity enforcement was first introduced, the goal was to prevent the generation of spurious oscillations in piecewise linear reconstructions. The same process is applied in CENO. The solution q is reconstructed at the element vertexes where the piecewise linear solution exhibits extrema. If the vertex reconstructions sit outside the bounds of the stencil's volume averaged values, a slope limiter is applied. Mathematically, in element λ with volume averaged solution Q_λ the reconstruction q_λ takes the form

$$q_\lambda = D^\lambda_{000} + D^\lambda_{100}(x - x_\lambda) + D^\lambda_{010}(y - y_\lambda) + D^\lambda_{001}(z - z_\lambda) \tag{3.2.21}$$

where the reconstruction coefficients D^λ_{100}, D^λ_{010}, and D^λ_{001} define the slope of q_λ in the x, y, and z directions respectively. The limited solution $\tilde{q}_\lambda$ is given by

$$\tilde{q}_\lambda = \tilde{D}^\lambda_{000} + \tilde{D}^\lambda_{100}(x - x_\lambda) + \tilde{D}^\lambda_{010}(y - y_\lambda) + \tilde{D}^\lambda_{001}(z - z_\lambda) \tag{3.2.22}$$

where the adjusted slopes $\tilde{D}^\lambda_{100} = \Theta_\lambda D^\lambda_{100}$, $\tilde{D}^\lambda_{010} = \Theta_\lambda D^\lambda_{010}$, and $\tilde{D}^\lambda_{001} = \Theta_\lambda D^\lambda_{001}$ are written in terms of a slope limiter Θ_λ. The adjusted cell center value,

$$\tilde{D}^\lambda_{000} = (1 - \Theta_\lambda)\, Q_\lambda + \Theta_\lambda D^\lambda_{000}, \tag{3.2.23}$$

enforces conservation when applying slope limiters on non-rectilinear meshes. This cell center adjustment affects monotonicity, however since the limiter approach is excessively diffusive, the effect on stability is minimal. The slope limiter Θ_λ is given by the Barth-Jesperson limiter[41;42;44;45]. The choice was based on the ease and locality of calculation in a numerical context. Given element λ's neighbors $\gamma \in \bar{\Gamma}_\lambda$, the Barth-Jesperson limiter is found using the following steps:

1. Given an element with vertex locations $\vec{x}^\lambda_i$, find the vertex reconstructions $\bar{q}^i_\lambda = q_\lambda(\vec{x}^\lambda_i)$.

2. Given the neighboring elements, find the extrema bounds of volume averaged solutions for the stencil.

$$\Delta Q_\lambda^{\max} = \max\left(\max_{\gamma\in\Gamma_\lambda}(Q_\gamma) - Q_\lambda, 0\right) \tag{3.2.24}$$

$$\Delta Q_\lambda^{\min} = \min\left(\min_{\gamma\in\Gamma_\lambda}(Q_\gamma) - Q_\lambda, 0\right) \tag{3.2.25}$$

3. Generate the vertex limiters

$$\Theta_\lambda^i = \begin{cases} \min\left(1, \frac{\Delta Q_\lambda^{\max}}{\vec{q}_\lambda^i - Q_\lambda}\right) & \text{if } \vec{q}_\lambda^i - Q_\lambda > 0 \\ \min\left(1, \frac{\Delta Q_\lambda^{\min}}{\vec{q}_\lambda^i - Q_\lambda}\right) & \text{if } \vec{q}_\lambda^i - Q_\lambda < 0 \\ 1 & \text{else} \end{cases} \tag{3.2.26}$$

4. The element's slope limiter is the minimal vertex limiter

$$\Theta_\lambda = \min_i\left(\Theta_\lambda^i\right) \tag{3.2.27}$$

- **High-order reconstruction**:

  High-order reconstructions are generated as discussed in Sec. 3.2.1. Testing the smoothness can be done in a variety of ways, and this study used a simple comparison of reconstructed values at stencil element centroids to the corresponding volume average solutions[41;42]. Given element λ, with volume average solution Q_λ and reconstruction q_λ, a reconstruction based on stencil elements Γ_λ results in a smoothness indicator,

$$\mathcal{S}_\lambda = \frac{N_s - N_D + 1}{N_D + 1}\frac{\alpha_\lambda}{\max(\epsilon, 1 - \alpha_\lambda)}, \tag{3.2.28}$$

where

$$\alpha_\lambda = 1 - \sum_{\gamma \in \Gamma_\lambda} \frac{\left(q_\lambda\left(\vec{x}_\gamma\right) - Q_\gamma\right)^2}{\left(Q_\lambda - Q_\gamma\right)^2}. \tag{3.2.29}$$

The factor $\epsilon = 10^{-8}$ is used to avoid singularities. An accurate, yet expensive, alternative smoothness indicator for unstructured meshes can be found in Hu and Shu [46]. The threshold smoothness $\mathcal{S}$ is moderately mesh dependent, however a value of 10^3 was found to work relatively well at removing oscillations for most problems [41;42].

The main drawback with the CENO method, and most reconstruction based finite volume methods, is that boundary conditions must be based on some Dirichlet application of values to ghost elements or nodal locations on the boundary of the domain. This is done to keep the reconstruction stencils symmetric along the domain's border. This boundary condition issue is the main motivations for the development of a discontinuous Galerkin method.

3.3 Nodal discontinuous Galerkin method

Discontinuous Galerkin methods (DGM) are popular discontinuous finite element methods for modeling shocks and discontinuities for nonlinear hyperbolic equation sets. This research focuses on applications to the one dimensional slab lines, two dimensional slab triangles, and three dimensional tetrahedra primitives (see Appendix A). Unlike FVMs where a local stencil of surrounding elements are used to define q within an element, DGMs use a basis representation within the element. Figure 3.3.1 shows how the solution in each element is fully represented by a set of nodal values. This representation allows for arbitrarily high-orders of accuracy on poorly conditioned unstructured meshes.

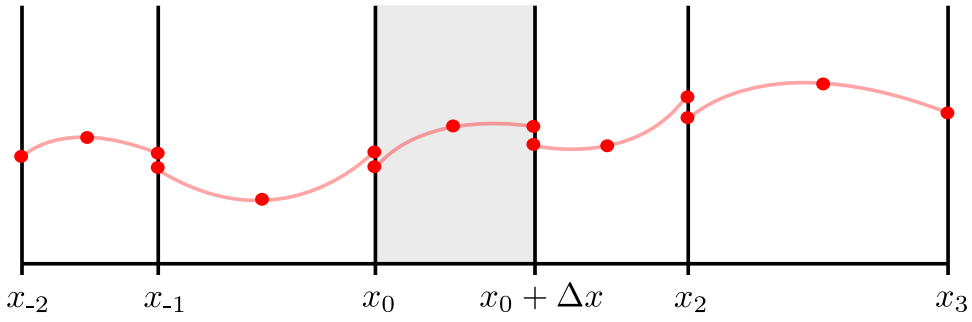


Figure 3.3.1: Discontinuous Galerkin methods have high-order representations of the solution contained within each element. For this study, a nodal representation is used, meaning the solution is known at a set of points (red dots) and interpolation functions (red lines) are used to find the solution between points.

This representation implies that all elements can be mapped to an isoparametric primitive form described in Appendix A. The mapping from configuration space (x, y, z) to the isoparametric space (ξ_0, ξ_1, ξ_2) is expressed through a coordinate transform Jacobian matrix

$$J_{ij}^{\lambda} = \frac{\partial}{\partial x_i} \xi_j \tag{3.3.1}$$

which exists for each element λ individually. In the case of polygonal simplex elements, J_{ij}^{λ} is independent of the isoparametric spatial position and can be pulled out of any integrals over or differentiations of isoparametric space. The isoparametric mapping is applied to Eq. (3.0.1) for element λ by pulling the coordinate transform Jacobian matrix out of the spatial derivatives

$$\frac{\partial}{\partial t} q_i^{\lambda} + J_{jk}^{\lambda} \frac{\partial}{\partial \xi_k} f_{ij}^{\lambda} = s_i^{\lambda}. \tag{3.3.2}$$

Within each element λ, the spatially dependent components in the equation are expanded

on a basis $\vec{\phi}(\vec{\xi})$ such that the solution vector becomes

$$q_i^\lambda(t,\vec{\xi}) = q_{ij}^\lambda(t)\phi_j(\vec{\xi}), \tag{3.3.3}$$

where the j index in q_{ij}^λ represents the nodal location within element λ. Applying the basis expansion to Eq. (3.3.2) results in the basis expanded strong form

$$\phi_j \frac{\partial}{\partial t} q_{ij}^\lambda + J_{jk}^\lambda \frac{\partial}{\partial \xi_k} f_{ij}^\lambda = s_i^\lambda. \tag{3.3.4}$$

The basis expanded strong form is converted to a weak form by multiplying by a test function and integrating over the element volume. Galerkin methods are special case of finite element methods where the test function is chosen from the basis set $\vec{\phi}$. Multiplying Eq. (3.3.4) by the test function ϕ_k and integrating over the isoparametric element yields

$$\langle \phi_m \phi_j \rangle \frac{\partial}{\partial t} q_{ij}^\lambda = -J_{jk}^\lambda \left\langle \phi_m \frac{\partial}{\partial \xi_k} f_{ij}^\lambda \right\rangle + \left\langle \phi_m s_i^\lambda \right\rangle. \tag{3.3.5}$$

where the angle brackets denote integration over an isoparametric primitive,

$$\langle g \rangle = \int g \, dV = \begin{cases} \frac{1}{2} \int_{-1}^{1} g \, d\xi_0 & \text{Line (1D)} \\ 2 \int_0^1 \int_0^{1-\xi_0} g \, d\xi_1 d\xi_0 & \text{Triangle (2D)} \\ 6 \int_0^1 \int_0^{1-\xi_0} \int_0^{1-\xi_0-\xi_1} g \, d\xi_2 d\xi_1 d\xi_0 & \text{Tetrahedron (3D).} \end{cases} \tag{3.3.6}$$

In order to describe the evolution of the solution at a single node, the mass matrix,

$$\mathbf{M}_{ij} = \langle \phi_i \phi_j \rangle, \tag{3.3.7}$$

must be removed from the time derivative. Multiplying Eq. (3.3.5) by the inverse of the

mass matrix, and swapping some indexes, yields

$$\frac{\partial}{\partial t} q_{ij}^{\lambda} = -J_{mn}^{\lambda} \mathbf{M}_{jk}^{\text{-}1} \left\langle \phi_k \frac{\partial}{\partial \xi_n} f_{im}^{\lambda} \right\rangle + \mathbf{M}_{jk}^{\text{-}1} \left\langle \phi_k s_i^{\lambda} \right\rangle . \tag{3.3.8}$$

As was discussed in Sec. 3.1, the flux term can be rewritten using integration by parts

$$\left\langle \phi_i \frac{\partial}{\partial \xi_j} f_{kl}^{\lambda} \right\rangle = \left\langle \frac{\partial}{\partial \xi_j} \left(\phi_i f_{kl}^{\lambda} \right) \right\rangle - \left\langle f_{kl}^{\lambda} \frac{\partial}{\partial \xi_j} \phi_i \right\rangle . \tag{3.3.9}$$

The divergence theorem is applied to the first term on the right-hand-side to give

$$\left\langle \frac{\partial}{\partial \xi_j} \left(\phi_i f_{kl}^{\lambda} \right) \right\rangle = \sum_{\gamma \in \bar{\Gamma}_{\lambda}} \left\langle m_j^{\lambda\gamma} \phi_i f_{kl}^{\lambda} \right\rangle_{\lambda\gamma} , \tag{3.3.10}$$

where $\bar{\Gamma}_{\lambda}$ is the set of elements neighboring element λ, $m_i^{\lambda\gamma}$ is the outward pointing isoparametric surface normal, and $\langle \cdot \rangle_{\lambda\gamma}$ represents the surface integral over the interface between elements λ and γ. Inserting this relation into Eq. (3.3.8) yields

$$\frac{\partial}{\partial t} q_{ij}^{\lambda} = \underbrace{J_{mn}^{\lambda} \mathbf{M}_{jk}^{\text{-}1} \left\langle f_{im}^{\lambda} \frac{\partial}{\partial \xi_n} \phi_k \right\rangle}_{\text{Internal Flux}} - \underbrace{J_{mn}^{\lambda} \sum_{\gamma \in \bar{\Gamma}_{\lambda}} \mathbf{M}_{jk}^{\text{-}1} m_n^{\lambda\gamma} \left\langle \phi_k f_{im}^{\lambda} \right\rangle_{\lambda\gamma}}_{\text{Surface Flux}} + \underbrace{\mathbf{M}_{jk}^{-1} \left\langle \phi_k s_i^{\lambda} \right\rangle}_{\text{Sources}} . \tag{3.3.11}$$

The surface flux shares information between elements, while the internal flux and source terms are locally evaluated within the element. Equation (3.3.11) can be further simplified by recognizing that the normal vector in the isoparametric space is convetered into the face normal vector in real space $n_i^{\lambda\gamma}$ through the multiplication with the Jacbobian matrix

$$n_i^{\lambda\gamma} = \frac{1}{G_{\lambda\gamma}} J_{ji}^{\lambda} m_j^{\gamma\lambda} \tag{3.3.12}$$

where

$$G_{\lambda\gamma} = \sqrt{J_{ik}^{\lambda} m_i^{\lambda\gamma} J_{jk}^{\lambda} m_j^{\lambda\gamma}} \tag{3.3.13}$$

is a scaling factor related to the geometry of the real element's face. This operation changes Eq. (3.3.11) into

$$\frac{\partial}{\partial t}q_{ij}^{\lambda} = J_{mn}^{\lambda}\mathbf{M}_{jk}^{-1}\left\langle f_{im}^{\lambda}\frac{\partial}{\partial \xi_n}\phi_k\right\rangle - \sum_{\gamma\in\bar{\Gamma}_\lambda} G_{\lambda\gamma}\mathbf{M}_{jk}^{-1}\left\langle \phi_k n_n^{\lambda\gamma} f_{in}^{\lambda}\right\rangle_{\lambda\gamma} + \mathbf{M}_{jk}^{-1}\left\langle \phi_k s_i^{\lambda}\right\rangle. \tag{3.3.14}$$

The normal flux defined in Sec. 3.1.1

$$\mathcal{F}_i^{\lambda\gamma} = n_j^{\lambda\gamma} f_{ij}^{\lambda} \tag{3.3.15}$$

is seen in the surface integral. As was seen in Sec. 3.1, the flux-normal multiplication can be converted into a rotation operation acting on a numerical flux. Again, the numerical flux calculation requires the state on both sides of the interface, implying that an efficient implementation has overlapping facial nodes between elements to minimize communication costs. This is the major advantage of the nodal DGM over the modal method where all modes from each element are required for each numerical flux evaluation. The numerical and internal fluxes can be expanded on a basis $\vec{\theta}$ such that

$$\mathcal{F}_i^{\lambda\gamma} = \mathcal{F}_{ij}^{\lambda\gamma}\theta_j \tag{3.3.16}$$

and

$$f_{ij}^{\lambda} = f_{ijk}^{\lambda}\theta_k. \tag{3.3.17}$$

The reason for having different bases for the solution (ϕ_i) and flux (θ_i) is due to the non-linearity associated with a given equation set. For linear equations, such as the advection equation

$$\frac{\partial}{\partial t}q = -\frac{\partial}{\partial x}q, \tag{3.3.18}$$

the basis used for the flux can be the same as the solution's basis ($\vec{\theta} = \vec{\phi}$). This is not necessarily the case for nonlinear equations sets, such as Burger's equation

$$\frac{\partial}{\partial t} q = -\frac{1}{2} \frac{\partial}{\partial x} q^2, \tag{3.3.19}$$

where the flux $f = q^2/2$ exists on a higher order basis than the solution.

The source term is similarly expanded on an additional basis

$$s_i^\lambda = s_{ij}^\lambda \psi_j. \tag{3.3.20}$$

Applying this multi-basis expansion to Eq. (3.3.14) yields

$$\frac{\partial}{\partial t} q_{ij}^\lambda = J_{ml}^\lambda \Upsilon_{jlk} f_{imk}^\lambda - \sum_{\gamma \in \bar{\Gamma}_\lambda} G_{\lambda\gamma} \Xi_{jk}^{\lambda\gamma} \mathcal{F}_{ik}^{\lambda\gamma} + \Psi_{jk} s_{ik}^\lambda. \tag{3.3.21}$$

with basis arrays

$$\Upsilon_{ijk} = \mathbf{M}_{il}^{-1} \left\langle \phi_l \frac{\partial}{\partial \xi_j} \theta_k \right\rangle, \tag{3.3.22}$$

$$\Xi_{ij}^{\lambda\gamma} = \mathbf{M}_{il}^{-1} \left\langle \theta_j \phi_l \right\rangle_{\lambda\gamma}, \tag{3.3.23}$$

and

$$\Psi_{ij} = \mathbf{M}_{ik}^{-1} \left\langle \phi_k \psi_j \right\rangle. \tag{3.3.24}$$

Note that all of these basis arrays are defined in the isoparametric space, and the same basis arrays can be used for each element. For linear systems, where the flux and source terms are

proportional to the solution (i.e. $\vec{\theta}, \vec{\psi} = \vec{\phi}$), the basis arrays can be simplified to

$$\bar{\Upsilon}_{ijk} = \mathbf{M}_{il}^{-1} \left\langle \phi_l \frac{\partial}{\partial \xi_j} \phi_k \right\rangle, \tag{3.3.25}$$

$$\bar{\Xi}_{ij}^{\lambda\gamma} = \mathbf{M}_{il}^{-1} \left\langle \phi_j \phi_l \right\rangle_{\lambda\gamma}, \tag{3.3.26}$$

and

$$\bar{\Psi}_{ij} = \mathbf{M}_{ik}^{-1} \left\langle \phi_k \phi_j \right\rangle = \delta_{ij}. \tag{3.3.27}$$

This simplification is very important for computational efficiency. For linear systems, the fluxes and sources are calculated at the same nodes that the solution is already known. This greatly reduces communication costs, as the entire basis set for the solution does not need to be used to find the solution at the nodal locations described by $\vec{\psi}$ and $\vec{\theta}$. While the linear basis sets do not accurately capture element scale dynamics in strongly nonlinear systems, the computational efficiency associated with them makes them advantageous for the large equation sets associated with multi-species models.

3.3.1 Gradient and diffusion operators

Diffusion operators are used throughout computational physics models to describe both elliptic and parabolic behavior. The operators used for viscosity and thermal conductivity (Sec. 2.3), diffusive stabilization (Sec. 2.4), and parabolic cleaning of Maxwell's equations (Sec. 2.6.2) all take the generalized form

$$\frac{\partial}{\partial t} q_i = \frac{\partial}{\partial x_j} \left(D_{ijkl} \frac{\partial}{\partial x_k} q_l \right). \tag{3.3.28}$$

The method used to evaluate diffusion operators is based on the work done by Cockburn and Shu[47], Kirby and Karniadakis[48], Arnold et al.[49], and Arnold et al.[50]. To evaluate a diffusion operator of this form using discontinuous Galerkin methods the operator is first converted into the flux form

$$\frac{\partial}{\partial t} q_i = -J^{\lambda}_{jk} \frac{\partial}{\partial \xi_k} f_{ij} \tag{3.3.29}$$

from Sec. 3.3 where

$$f_{ij} = -D_{ijkl} \frac{\partial}{\partial x_k} q_l \tag{3.3.30}$$

is the viscous flux that must be evaluated on the nodal basis $\vec{\phi}$. To solve this, the gradient term is redefined as

$$Y_{ij} = \frac{\partial}{\partial x_j} q_i \tag{3.3.31}$$

such that the viscous flux becomes

$$f_{ij} = -D_{ijkl} Y_{lk}. \tag{3.3.32}$$

As before, the equation is converted to the isometric space $x \to \xi$ to give

$$Y^{\lambda}_{ij} = J^{\lambda}_{jk} \frac{\partial}{\partial \xi_k} q^{\lambda}_i. \tag{3.3.33}$$

The gradient array is expanded on the solution's basis $\vec{\phi}$,

$$Y^{\lambda}_{ij} = Y^{\lambda}_{ijk} \phi_k, \tag{3.3.34}$$

to give

$$Y^{\lambda}_{ijl}\phi_l = J^{\lambda}_{jk}\frac{\partial}{\partial \xi_k}q^{\lambda}_i. \tag{3.3.35}$$

As before, this basis expanded strong form is converted to the weak form by multiplying by the basis and integrating over the isometric element (see Eq. (3.3.6)) to give

$$Y^{\lambda}_{ijl}\left\langle \phi_l\phi_n\right\rangle = J^{\lambda}_{jk}\left\langle \phi_n\frac{\partial}{\partial \xi_k}q^{\lambda}_i\right\rangle. \tag{3.3.36}$$

The mass matrix defined in Eq. (3.3.7) is used to simplify the left hand side

$$Y^{\lambda}_{ijk} = J^{\lambda}_{jl}\mathbf{M}^{-1}_{nk}\left\langle \phi_n\frac{\partial}{\partial \xi_l}q^{\lambda}_i\right\rangle. \tag{3.3.37}$$

Integrating by parts breaks the integral into a surface contribution, and an internal contribution, which is written as

$$Y^{\lambda}_{ijk} = \underbrace{J^{\lambda}_{jl}\sum_{\gamma\in\bar{\Gamma}_\lambda} m^{\lambda\gamma}_l\mathbf{M}^{-1}_{nk}\left\langle \phi_n q^{\lambda}_i\right\rangle_{\lambda\gamma}}_{\text{Surface Flux}} - \underbrace{J^{\lambda}_{jl}\mathbf{M}^{-1}_{nk}\left\langle q^{\lambda}_i\frac{\partial}{\partial \xi_l}\phi_n\right\rangle}_{\text{Internal Flux}}. \tag{3.3.38}$$

By converting the isoparametric face normal frame to the world frame face normal using Eq. (3.3.12), Eq. (3.3.38) can be rewritten as

$$Y^{\lambda}_{ijk} = \sum_{\gamma\in\bar{\Gamma}_\lambda} G_{\lambda\gamma}n^{\lambda\gamma}_j\mathbf{M}^{-1}_{nk}\left\langle \phi_n q^{\lambda}_i\right\rangle^{\partial}_l - J^{\lambda}_{jl}\mathbf{M}^{-1}_{nk}\left\langle q^{\lambda}_i\frac{\partial}{\partial \xi_l}\phi_n\right\rangle \tag{3.3.39}$$

where $G_{\lambda\gamma}$ is defined in Eq. (3.3.13). Similar to the normal flux described in Sec.3.1.1, the evaluation of $\vec{q}$ on the surface of the element is rewritten as

$$\mathcal{Q}^{\lambda\gamma}_{ij} = q^{\lambda}_i n^{\lambda\gamma}_j \tag{3.3.40}$$

and is expanded,

$$\mathcal{Q}_{ij}^{\lambda\gamma} = \mathcal{Q}_{ijk}^{\lambda\gamma}\phi_k, \tag{3.3.41}$$

on the same basis as $\vec{q}_\lambda$. As was seen for the normal flux in the finite volume and discontinuous Galerkin methods, the surface quantity $\mathcal{Q}_{\lambda\gamma}$ must be single valued at the interfaces between elements. The final form for the gradient operator is then

$$Y_{ijk}^{\lambda} = \sum_{\gamma\in\bar{\Gamma}_\lambda} G_{\lambda\gamma}\mathcal{Q}_{ijl}^{\lambda\gamma}\bar{\Xi}_{kl}^{\lambda\gamma} - J_{jl}^{\lambda}\bar{\Upsilon}_{klm}q_{im}^{\lambda} \tag{3.3.42}$$

with basis arrays

$$\bar{\Xi}_{ij}^{\lambda\gamma} = \mathbf{M}_{ik}^{-1}\left\langle \phi_k\phi_j \right\rangle_{\lambda\gamma} \tag{3.3.43}$$

and

$$\bar{\Upsilon}_{ijk} = \mathbf{M}_{il}^{-1}\left\langle \phi_k\frac{\partial}{\partial x_j}\phi_l \right\rangle_{\lambda\gamma}. \tag{3.3.44}$$

The surface value $\mathcal{Q}_{\lambda\gamma}$ can take many forms[49;50], and the form used in this study is

$$\mathcal{Q}_{ij}^{\lambda\gamma} = \left(q_i^{\lambda}\frac{1+\beta}{2} + q_i^{\gamma}\frac{1-\beta}{2}\right)n_j^{\lambda\gamma} \tag{3.3.45}$$

where β is a factor that is generally set in the bounds[49;50] $\beta \in (-1, 1)$. Since this diffusion model is a penalty method, the definition of the numerical flux used in Eq. 3.3.21 must also take the β penalty into consideration such that

$$\mathcal{F}_i^{\lambda\gamma} = \frac{1}{2}\left(f_{ij}^{\lambda} + f_{ij}^{\gamma}\right)n_j^{\lambda\gamma} - \eta_{\lambda\gamma}\left(q_i^{\gamma} - q_i^{\lambda}\right) - \frac{1}{2}\beta D_{ik}^{\lambda\gamma}\left(Y_{kj}^{\lambda} - Y_{kj}^{\gamma}\right)n_j^{\lambda\gamma} \tag{3.3.46}$$

where $D_{ik}^{\lambda\gamma}$ is a equation set dependent matrix that describes how the penalty applies to the

various gradients. The penalty $\eta_{\lambda\gamma}$ is approximated as[50]

$$\eta_{\lambda\gamma} \approx \frac{1}{A_{\lambda\gamma}} \tag{3.3.47}$$

where $A_{\lambda\gamma}$ is the area of the face separating element λ and γ.

3.3.2 Node positioning

The nodal DGM minimizes communication costs by placing nodes at points where numerical fluxes are calculated. The generation of an interpolation polynomial basis from a given set of node positions is a fairly straightforward processes. The interpolation basis functions $\vec{\phi}$, $\vec{\theta}$, and $\vec{\psi}$ can all be generated from a different set of node positions, and this section generalizes the process for the basis $\vec{\phi}$. Given a set of node positions $\vec{x}_i$, a set of interpolation basis functions are defined by the identity

$$\phi_i(\vec{\xi_j}) = \delta_{ij} \tag{3.3.48}$$

which states that the interpolation polynomial has a value of 1 at its associated node ($j = i$), and 0 at all other nodes ($j \neq i$). The first step in deriving an interpolation basis is to expand the nodal basis,

$$\phi_i = \phi_{ij}\varphi_j, \tag{3.3.49}$$

on an arbitrary basis $\vec{\varphi}$. For large sets of basis nodes (e.g. a primitive containing 20 nodes), it is helpful to use an orthogonal basis[37] which obeys the condition

$$\langle \varphi_i \varphi_j \rangle \propto \delta_{ij}. \tag{3.3.50}$$

For small node sets, a simple monomial series may be used instead, such as the one described in Sec. 3.2.1. Combining Eq. (3.3.49) with Eq. (3.3.48) gives

$$\phi_i(\vec{\xi_j}) = \phi_{ik}\varphi_k(\vec{\xi_j}) = \delta_{ij}, \tag{3.3.51}$$

which describes the interpolation basis $\vec{\phi}$ in terms of the known basis $\vec{\varphi}$. Defining a matrix

$$V_{ij} = \varphi_i(\vec{\xi_j}) \tag{3.3.52}$$

allows for solving for the expansion variables

$$\phi_{ik}V_{kj}V_{lj}^{-1} = \delta_{ij}V_{lj}^{-1} \tag{3.3.53}$$

which leads to the definition

$$\phi_{ij} = V_{ji}^{-1}. \tag{3.3.54}$$

For this research, the discontinuous Galerkin method is implemented for unit line, triangle, and tetrahedral primitives as described in Appendix A. The basis $\vec{\varphi}$ used in this research was defined separately for the three primitives. In one dimension, the basis is given by

$$\varphi_i(\xi_0) = P_i(2\xi_0) \tag{3.3.55}$$

with Legendre polynomials $P_i(x)$. The basis used for two dimensions are designed for unit right angle triangles and is given by Hesthaven and Warburton[37]

$$\varphi_{m_{ij}} = 2^{i+1/2}(1-\xi_1)^i P_i\left(\frac{2\xi_0}{1-\xi_1} - 1\right) J_j^{2i+1,0}(2\xi_1 - 1) \tag{3.3.56}$$

where $J_i^{jk}(x)$ is the i^{th} order Jacobi polynomial. The composite index

$$m_{ij} = j + (p+1)i + 1 - \frac{i}{2}(i-1) \tag{3.3.57}$$

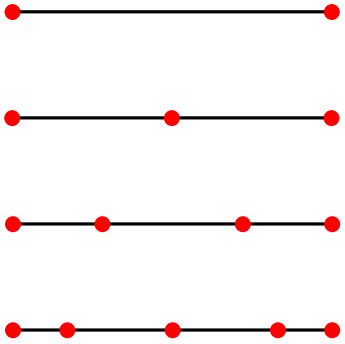
combines the indexes $i, j \geq 0$ into a single index using the highest basis order $p \geq i + j$. For right angle, right-handed, unit tetrahedra, the orthogonal basis is also related to Jacobi polynomials

$$\begin{aligned} \varphi_{m_{ijk}} = 2^{2i+j+3/2} (1-\xi_1-\xi_2)^i (1-\xi_2)^j P_i\left(\frac{2\xi_0}{1-\xi_1-\xi_2} - 1\right) \\ J_j^{2i+1,0}\left(\frac{2\xi_1}{1-\xi_2} - 1\right) J_k^{2i+2j+2,0}(2\xi_2 - 1). \end{aligned} \tag{3.3.58}$$

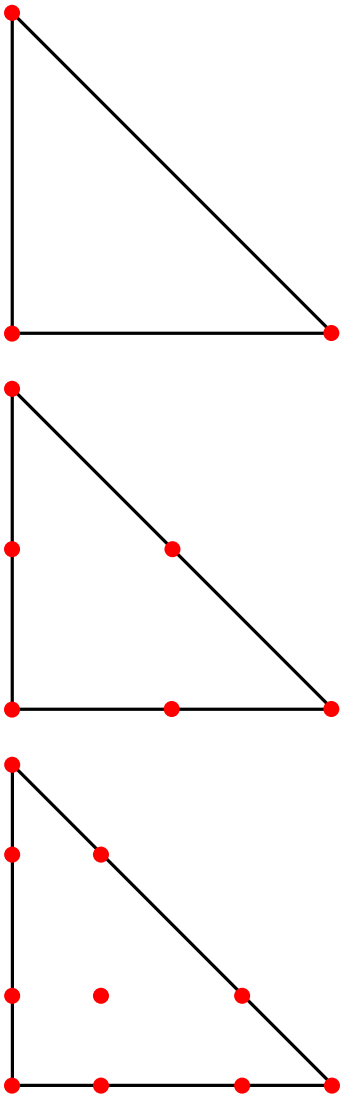
The composite index

$$m_{ijk} = 1 + \frac{1}{6}\left(11 + 12p + 3p^2\right) i + \frac{1}{2}(2p+3) j + k - \frac{1}{2}(2+p) i^2 - ij - \frac{1}{2}j^2 + \frac{1}{6}i^3 \tag{3.3.59}$$

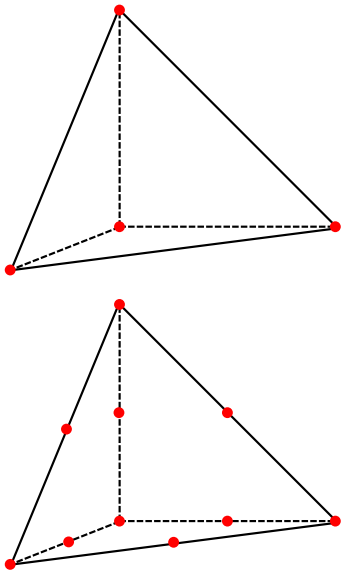
is again used to convert from indexes $i, j, k \geq 0$ to a single index based on the highest basis order $p \geq i+j+k$. The choice of node positioning is determined by minimizing the condition number of the Vandermonde matrix using Lebesgue constants[37]. This becomes important when increasing the number of nodes in a primitive; however, this study focused more on the development of small computationally efficient node sets. In one dimension the node locations are given by the Gauss-Lobatto quadrature node locations, given in Tab. 3.1, as advised in Hesthaven and Warburton[37], which gives nodes both on the boundary of the line element and within the line element. For the triangular and tetrahedral elements the basis is approximated by a tensor form Gauss-Lobatto quadrature. These node configurations are seen in Fig. 3.3.2.



(a) One dimensional element node sets. From top to bottom: Piecewise linear, quadratic, cubic, quartic, and quintic interpolation polynomial nodes.



(b) Two dimensional element node sets. From top to bottom: Piecewise linear, quadratic, and cubic interpolation polynomial nodes.



(c) Two dimensional element node sets. From top to bottom: Piecewise linear and quadratic interpolation polynomial nodes.

Figure 3.3.2: Discontinuous Galerkin node positions based on Gauss-Lobatto quadrature positions, as specified in Table 3.1.

Table 3.1: Gauss-Lobatto abscissa x_i for 2, 3, 4, and 5 point node sets. Additional abscissa can be found in Hesthaven and Warburton[37] for higher order node sets.

m	x_i
2	± 1
3	$0, \pm 1$
4	$\pm\sqrt{\frac{1}{5}}, \pm 1$
5	$0, \pm\sqrt{\frac{3}{7}}, \pm 1$

3.3.3 Monotonicity enforcement

As was seen with the finite volume method in Sec. 3.2.2, the discontinuous Galerkin method, being discontinuous, requires a monotonicity enforcement scheme to remain stable. To enforce monotonicity, the CENO method, implemented for the finite volume method in Sec. 3.2.2, is extended to work with the discontinuous Galerkin method. The implementation first calculates the volume average quantity for each element using the relation

$$Q_\lambda = \frac{1}{V_\lambda} \int_\lambda q_\lambda \, dV = q_i^\lambda \left\langle \phi_i \right\rangle , \tag{3.3.60}$$

where i refers to a node in an element. Given the volume averages for all elements, a piecewise linear k-exact reconstruction is applied to the element and its nearest neighbors. The first-order polynomial solution is then limited using the Barth-Jesperson limiter of Sec. 3.2.2. The limited solution is then sampled at the nodal locations to rebuild the discontinuous Galerkin solution. There are also higher-order monotonicity enforcement methods based on WENO schemes[51]. WENO limiting schemes retain high-order accuracies in the presence of discontinuities making them popular for modern discontinuous Galerkin applications. The main issue with WENO limiters is that they are slow and computationally expensive.

3.3.4 Boundary conditions

Discontinuous Galerkin methods are locally defined finite element methods, meaning that, unlike the finite volume method, the high-order solution is both stored and evolved directly without requiring a reconstruction scheme. As such, each element is evolved through two operations, the internal flux operator which describes the flow of the solution within an element, and the surface flux operator which captures the flow of the solution between elements. Boundary conditions for DG methods are then implemented by setting a surface flux at the edge of a domain. Dirichlet boundary conditions are implemented by projecting a solution onto the outside of a boundary face which effectively sets the desired numerical flux. For example, in the case of a 5-moment model interacting with a solid wall traveling at some velocity $\vec{v}_w = \begin{bmatrix} v_x^w & v_y^w & v_z^w \end{bmatrix}$, the solution on the inboard $(-)$ side of the faces defining the domain boundary,

$$\vec{q}_- = \begin{bmatrix} \rho_- & p_x^- & p_y^- & p_z^- & \epsilon_- \end{bmatrix}^T, \tag{3.3.61}$$

is used to define the solution on the outboard $(+)$ side of the domain boundary

$$\vec{q}_+ = \begin{bmatrix} \rho_+ \\ p_x^+ \\ p_y^+ \\ p_z^+ \\ \epsilon_+ \end{bmatrix} = \begin{bmatrix} \rho_- \\ \rho_- v_x^w \\ \rho_- v_y^w \\ \rho_- v_z^w \\ \epsilon_- + \frac{1}{2}\left(\rho_- v_w^2 - p_-^2/\rho_-\right) \end{bmatrix}. \tag{3.3.62}$$

This particular boundary condition species a force balance across the interface such that no mass, momentum, or energy transits the boundary. Naturally this boundary condition is only applicable to systems where the wall velocity is purely in the wall's plane. Neumann boundary conditions are set in a similar manner where the solution's gradient on the outboard side of the domain's boundary is set explicitly.

3.4 Temporal discretization

Plasma models can generally be written in terms of a first-order, ordinary differential equation (ODE),

$$\frac{\partial}{\partial t} q = \mathcal{L}\left(q, t\right), \tag{3.4.1}$$

where q is the evolving solution vector and the $\mathcal{L}$ operator represents all spatial and temporal dependencies specified by the finite volume (Sec. 3.2) or discontinuous Galerkin (Sec. 3.3) discretizations. The discretization of the time derivative operator in Eq. (3.4.1) is known as a time integration scheme.

The time integration schemes used in this study are a form of explicit methods where the known solution q^n at a time step n is used to generate the right-hand-side variable $\mathcal{L}(q^n, t_n)$ for a particular time t_n. Explicit methods are highly parallelizable schemes capable of high-order convergence accuracies. Additionally, the locality of explicit operators makes them easy to parallelize, however, this locality leads to a stability constraint on the maximum allowable time step size $\Delta t = t_{n+1} - t_n$. For example, when explicit schemes are applied to hyperbolic conservation laws the stability of the algorithm is limited by the Courant number C. The Courant number defines a maximum stable time step

$$\Delta t < C\frac{\Delta x}{v} \tag{3.4.2}$$

allowed in a system given a maximum hyperbolic wave speed v and a local element width Δx. For linear equation sets, such as the advection equation, the finite volume method can operate with a Courant number of $C < 1$. The stability restrictions for discontinuous Galerkin methods are more restrictive, and depend on the basis order. A higher order basis requires a smaller time step for the same element size. This is further complicated in the case of unstructured, or general geometry meshes, where defining an appropriate Δx

becomes difficult. Additional time step constraints can be derived for source terms and diffusive operators. For example, the most restrictive time step constraint for multi-species plasma simulations is rarely set by the speed of light or fluid viscosity, but by the electron plasma frequency, which must be resolved ($\Delta t \ll 1/\omega_p^e$).

This study implemented a set of explicit Runge-Kutta (RK) methods to handle time integration. RK methods are a class of high-order accurate ordinary differential equation solvers. The first-order accurate time integration scheme, mainly used for debugging purposes, is given by the forward Euler method

$$q^{n+1} = q^n + \Delta t \cdot \mathcal{L}(q^n, t_n). \tag{3.4.3}$$

The second-order time integration is given by the midpoint method

$$q^* = q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}\left(q^n, t_n\right) \tag{3.4.4a}$$

$$q^{n+1} = q^n + \Delta t \cdot \mathcal{L}\left(q^*, t_n + \frac{1}{2}\Delta t\right). \tag{3.4.4b}$$

A third-order time integration is given by the RK3 scheme

$$q^* = q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}\left(q^n, t_n\right) \tag{3.4.5a}$$

$$q^{**} = 3q^n - 2q^* + 2\Delta t \cdot \mathcal{L}\left(q^*, t_n + \frac{1}{2}\Delta t\right) \tag{3.4.5b}$$

$$q^{n+1} = -\frac{1}{3}q^n + q^* + \frac{1}{3}q_i^{**} + \frac{1}{6}\Delta t \cdot \mathcal{L}\left(q^{**}, t_n + \Delta t\right). \tag{3.4.5c}$$

The highest-order accurate scheme used is the fourth-order accurate RK4 scheme

$$q^* = q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}\left(q^n, t_n\right) \tag{3.4.6a}$$

$$q^{**} = q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}\left(q^*, t_n + \frac{1}{2}\Delta t\right) \tag{3.4.6b}$$

$$q^{***} = q^n + \Delta t \cdot \mathcal{L}\left(q^{**}, t_n + \frac{1}{2}\Delta t\right) \tag{3.4.6c}$$

$$q^{n+1} = \frac{1}{3}\left(-q^n + q^* + 2q^{**} + q^{***}\right) + \frac{1}{6}\Delta t \cdot \mathcal{L}\left(q^{***}, t_n + \Delta t\right). \tag{3.4.6d}$$

Alternative Runge-Kutta schemes used for this research include a second-order, strong stability preserving Runge-Kutta scheme[37], known as the Heun method,

$$q^* = q^n + \Delta t \cdot \mathcal{L}\left(q^n, t_n\right) \tag{3.4.7a}$$

$$q^{n+1} = \frac{1}{2}q^* + \frac{1}{2}q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}\left(q^*, t_n + \Delta t\right), \tag{3.4.7b}$$

and the third-order, strong stability preserving Runge-Kutta scheme

$$q^* = q^n + \Delta t \cdot \mathcal{L}\left(q^n, t_n\right) \tag{3.4.8a}$$

$$q^{**} = \frac{3}{4}q^n + \frac{1}{4}q^* + \frac{1}{4}\Delta t \cdot \mathcal{L}\left(q^*, t_n + \Delta t\right) \tag{3.4.8b}$$

$$q^{n+1} = \frac{1}{3}q^n + \frac{2}{3}q^{**} + \frac{2}{3}\Delta t \cdot \mathcal{L}\left(q^{**}, t_n + \frac{1}{2}\Delta t\right). \tag{3.4.8c}$$

All of these explicit time integration schemes are made up of stages, each of which constructs a right-hand-side variable $\mathcal{L}$ from the solution generated by the previous stage. This iterative structure means that the computational efficiency of an explicit method is entirely dependent on the efficiency of the spatial solver.

Chapter 4

COMPUTATIONAL FRAMEWORK

The models and numerical methods discussed in this dissertation are implemented in the WARPXM code (Washington Approximate Riemann Plasma eXtended modeling platform - Many-core version). WARPXM is designed to model large scale continuum systems containing the many characteristic scales associated with multi-species plasma models. The numerical solvers making up WARPXM are designed for evolving nonlinear equation sets, such as those described in Sec. 3, using a modular construction. A simulation is initialized using an input file that describes model in terms of a temporal integration scheme linked to a spatial discretization scheme. The temporal and spatial discretization schemes can be swapped out at will. The spatial discretization scheme is made up of a collection of subsolvers. For example, a plasma simulation contains separate subsolvers to handle advective flow, collisional processes, electromagnetic interactions, and Maxwell's equations. Swapping the modular subsolvers is as simple as modifying a few lines in the input file. Parsing the input file into WARPXM results in a collection of interlinked programs that are dynamically generated and compiled at runtime.

The generalized and modular approach of the WARPXM lends itself to fast numerical model development. Writing new modules for WARPXM is a simple process of copying a preexisting module, making the required modifications, and recompiling. The registry system within WARPXM automatically allows the module to be identified in an input file. The output from WARPXM is written to a proprietary hdf5 datafile format. The hdf5 files can be directly read into python or MATLAB, or can be folded into a XDMF translation file using a python script. The XDMF interface file allows the hdf5 files to be read into data visualization

programs including Visit or Paraview.

This chapter overviews the main features of WARPXM that have been developed over the course of this research. Section 4.1 overviews the underlying parallelization schemes and computational framework used in WARPXM. A discussion of the unstructured mesh framework implemented in WARPXM is found in Sec. 4.2. The constraints governed by the computational and unstructured frameworks required a new configuration for datasets, which is described in Sec. 4.2. Finally, Sec. 4.4 overviews the method used to implement the discontinuous Galerkin finite element method.

4.1 Parallelization in WARPXM

The WARPXM code is designed to target modern high-performance compute (HPC) environments, where the addition of compute accelerators, such as GPUs, has become commonplace. Compute accelerators greatly increase the effectiveness of HPC systems which is beneficial considering the high computational costs of plasma models.

Modern compute environments, such as clusters, employ high levels of parallelism. Parallelization takes place not only at the cluster level with multiple compute nodes, but also within the compute node where multiple compute devices (e.g. CPUs and GPUs) may exists. Each compute device is further subdivided into processing elements (e.g. CPU logical cores or GPU streaming processors). For the purposes of this study a compute environment is subdivided into three abstract levels, each with a task management and threading mechanism implemented using separate application programming interfaces (API).

- **Cluster level:**

  The cluster level represents a collection of compute nodes. In the case of a workstation there is only a single compute node, however in general, many compute nodes exist with the ability to pass messages to each other over the cluster's interconnect. Communication between a cluster's compute nodes is handled using the message passing

interface (MPI) standard, or more specifically MPI-2 as implemented by OpenMPI [1] or other commercial products. Each individual compute node has a single instantiation of WARPXM to enforce proper compute device management. A compute node's instantiation of WARPXM is known as the host level.

- **Host level:**

  Each compute node is made up of a collection of compute devices including CPUs and GPUs. To prevent overlapped use of compute devices, only a single instantiation of WARPXM should ever exist for a given compute node. The host level's management system is divided into task management and device management.

  A task is an abstract reference to a set of instructions implemented on a dataset such as applying a boundary condition to a particular boundary or saving data to an output file. Task management on the host level includes initialization, communication, and synchronization between tasks, and is handled using the Boost threadpool API [2]. When the threadpool library is given a task, it schedules the task for a thread within its pool of threads. When a thread becomes available, the threadpool runs the task on the thread. The threadpool library is only designed to run on CPU-like devices so while each thread can potentially complete a task itself, it cannot be directly extended to GPU compute devices.

  Management of compute devices (memory management and device synchronization) is handled by the OpenCL API [3]. The OpenCL API is split into control and compute environments. The control environment, handled within the host level's task manager, controls explicit global memory management and task assignment to compute devices. This interface is used in conjunction with the cluster level manager to handle global synchronization of datasets. The implementation of the numerical methods and plasma

[1] www.open-mpi.org

[2] www.boost.org

[3] www.khronos.org/opencl

models are found in the compute level.

- **Compute level:**

  The compute level is where the actual numerical solvers exist. The purpose of the host and cluster levels is to ensure memory is in the correct place at the correct time and that the numerical solvers are properly synchronized. As such the management systems in the cluster and host levels are computationally inexpensive. The bulk of computational expense in WARPXM exists in the compute level.

  Each compute device is made up of a collection of processing elements. A CPU generally contains anywhere from a single to a few dozen logical cores, while a GPU can contain thousands of streaming processors. Within each compute device, processing elements are grouped into compute units in which memory can be shared and synchronized.

  Within OpenCL, all devices can be thought of as single-instruction multiple-data (SIMD) parallelized frameworks. This means that the same set of instructions are applied on a collection of processing elements in lockstep to different datasets. The simplicity of the SIMD model greatly decreases computational overhead making them ideal for modeling parallelizable systems such as the numerical methods discussed in Sec. 3.

  OpenCL instruction sets are written in kernels using an OpenCL kernel language based on the C standard. A kernel can be thought of as a function call existing within a for loop. OpenCL unwraps this for loop and applies the function (kernel) individually on all available processing elements. In general multiple kernels are compiled for a given simulation and it is the host level's responsibility to properly sequence them. The implementation of the kernel code on a processing element is known as a work item, which is analogous to how a task is assigned to a thread in the host level's threadpool. Work items are collected into work groups capable of locally sharing and

synchronizing memory. The only synchronization available within OpenCL's compute level is between work items in a common work group. Synchronization between work groups must be handled on the host level. Work groups are assigned a compute unit on which they can run. Ideally, the number of work items in each work group should be a multiple of the number of processing elements in the compute unit. This allows the SIMD lockstep processing method to run a subset of the work items while other work items are accessing memory. Hiding memory access latency behind computation operations is the basis of the OpenCL methodology.

The compute level has four separate memory zones accessible to the compute processors, as shown in Fig. 4.1.1, with varying levels of latency and size: global, constant, local, and private.

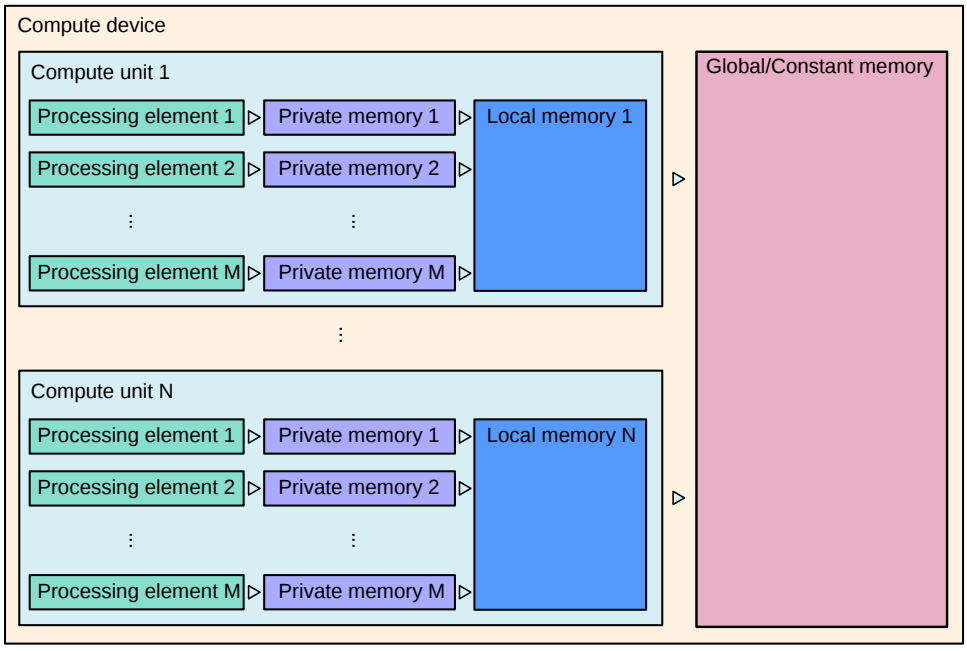


Figure 4.1.1: Hardware overview of an OpenCL compute device. Processing elements have a private memory space and are grouped into compute units with a shared local memory space. Compute units are grouped together with a shared global/constant memory space.

Global memory represents the compute device main memory used for holding large allocations of data ($\sim 10GB$ per device). For CPUs this is the compute node's RAM, while for GPUs this is the on-board VRAM. Global memory is the slowest for processing elements to access, but is the only storage persistent beyond the lifetime of a kernel. In practice this memory space should only be used for data storage between kernels. Constant memory is effectively global memory that is optimized to be loaded quickly onto all processing elements (i.e. broadcast operations). The exact location of constant memory is device dependent, and is severely restricted in terms of storage size ($\sim 100kB$ per device). Local memory is shared within the compute units on a compute device. It is much faster to access than global memory, but is also much smaller ($\sim 10kB$ per compute unit). While local memory is mainly used to share memory between work items in a work group, it is also good practice to use it to prevent overflow of private memory. Private memory represents the register space for a given processing element. It is usually very small ($\sim 1kB$ per processing element), but has the fastest access speed.

This brief overview of the compute level of the OpenCL model is a bit of an oversimplification, but it is enough for this dissertation's purposes.

4.2 Unstructured mesh framework

Numerical models are generally solved by projecting onto a mesh, the resolution of which determines how well the dynamics of interest are resolved. The previous generations of WARPXM (e.g. WARPX[52]) worked with structured meshes. Structured meshes line their elements up on a rectilinear grid, which can be used to greatly simplify the math and communication required for numerical implementation. While structured mesh codes are inherently easier to work with than unstructured mesh codes, structured mesh codes have difficulties in capturing complex geometries in that they tend to over-resolve some regions and under-resolve others. Alternatively, unstructured meshes can be mapped to any geometry without

resolution issues.

Before getting into the mechanics of meshes, some nomenclature is reviewed. A problem, such as a plasma simulation, is abstractly mapped to a mesh that spans a set of spaces, including configuration space (x, y, z) and possibly velocity space (v_x, v_y, v_z). The mesh is made up of a collection of elements, each of which is described by a primitive archetype defined by a set of vertexes known as the connectivity. For example, a triangle is a primitive made up of three vertexes. The vertexes describe the element geometry and, by comparing vertexes between elements, define how an element connects to other elements. The set of elements connected to an element is known as the element's neighborhood.

Prior to running a simulation, a user must generate a mesh using a mesh generation toolkit. The CUBIT toolkit was used in this study, which is designed for generating unstructured meshes from CAD files. The unstructured framework was developed to work with most mesh generation toolkits, with a focus on the CUBIT toolkit. Meshes are read into WARPXM using the ABAQUS [4] file format, which is then converted into and exported as a proprietary hdf5 format. The exported mesh file is designed to be used with the implemented XDMF plotting framework and for restarting simulations. Initial conditions can be generated within WARPXM, or through an external script using the exported mesh file.

While generating a mesh, the user has the option to subdivide the domain into subdomains. Subdomains define areas of the domain on which variables and subsolvers (i.e. plasma models) can exist. The goal of this mechanic is to allow the existence of multiple solvers in a domain. This is important for plasma simulations where it is feasible to have parameter regime transitions that are best resolved by different models. For example, when modeling electrode dynamics, a kinetic representation may be most accurate near the electrode, however further away, a fluid representation may be more computationally efficient. The meshing interface is also able to construct nodesets, or collections of nodes, that are used in

[4] http://www.simulia.com/

WARPXM to define the perimeter of a domain where boundary conditions are applied.

In general, attempting to solve an entire problem domain on a single compute device requires too much memory and/or runtime. Therefore, the mesh must be broken down into a collection of compute zones, known as patches, which are spread out among the available compute devices. The cost of communicating data between patches on a cluster represents the most expensive process in most numerical frameworks. To help alleviate inter-patch communication expenses required for handling boundary conditions between patches, a layered structure is imposed on a patch's element layout. An element layer is a collection of elements that share a minimum number of jumps to reach the boundary. The concept is expressed in Fig. 4.2.1 where element layer 0 describes elements on the inside edge of the boundary. Positive element layers exist external to the patch while negative layers exist within the patch.

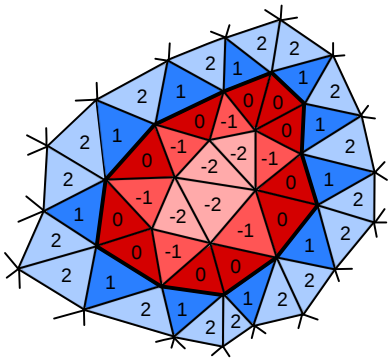


Figure 4.2.1: Triangular mesh patch decomposed into layers, identified by the numbers -2 through 2, based on distance between an element and the boundary (thick black line). External elements (blue) have positive layer values while the three internal element layers (red) are in layers ≤ 0.

A simple example of inter-patch communication is shown in Fig. 4.2.2 where two patches must synchronize each other's element layers. The practice of layering elements is important when considering mesh partitioning and memory management in the following sections.

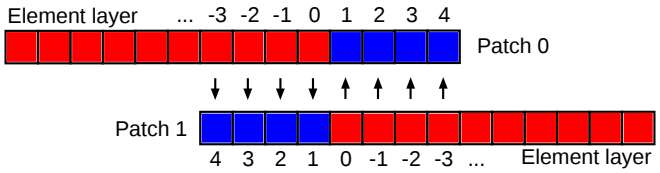
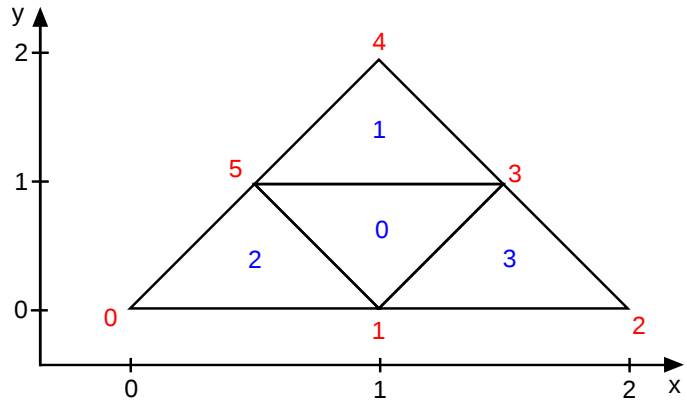


Figure 4.2.2: Shared boundary condition for two adjacent patches with overlapping elements used for communication. In this case patches 0 and 1 must send their internal (red) element layers to fill out their neighbor patch's external 'ghost' (blue) element layers.

The compute layer in WARPXM is designed around a SIMD parallelization scheme where each processing element is processing in lockstep with potentially every other processing element in a compute unit. To prevent the wasted computational effort brought on by conditional statements, the unstructured meshes used in WARPXM are limited to a single primitive type per mesh. For example, a two dimensional mesh can only be made up of entirely triangles or quadrilaterals. To simplify things further, WARPXM is designed for polygonal isoparametric primitives, which are fully described by a minimal number of vertexes (e.g. triangles have 3 vertexes and flat faces). This reduces the complexity and computational expense of rotation and integration operations.

Since all elements are described by a single primitive type, only coordinate arrays, defining the complete set of node positions, and connectivity arrays, defining the node indexes that make up each element, are required to describe a mesh. The ordering of node indexes in the connectivity array is specific for each primitive type, and is used to define the geometry and neighborhood for each element. The relation between connectivity array layout and the geometry for each element is outlined in Appendix A. Figure 4.2.3 gives an example of the coordinate, connectivity, and neighborhood arrays for a simple triangular mesh. Currently, neighborhood arrays are generated using a face comparison/sort algorithm, where the nodal make up of individual elements are compared in order to find matching faces.



$$\text{Coordinate Array} = [(0,0),(1,0),(2,0),(1.5,1),(1,2),(0.5,1)]$$
$$\text{Connectivity Array} = [(1,3,5),(5,3,4),(0,1,5),(1,2,3)]$$
$$\text{Neighborhood Array} = [(3,1,2),(0,\text{-}1,\text{-}1),(\text{-}1,0,\text{-}1),(\text{-}1,\text{-}1,0)]$$

Figure 4.2.3: Triangle mesh representation by coordinate, connectivity and neighborhood arrays for 6 nodes (red) describing 4 elements (blue). The neighborhood array uses "-1" for non-existent elements. Connectivity and neighborhood array layouts for other primitives are discussed in Appendix A.

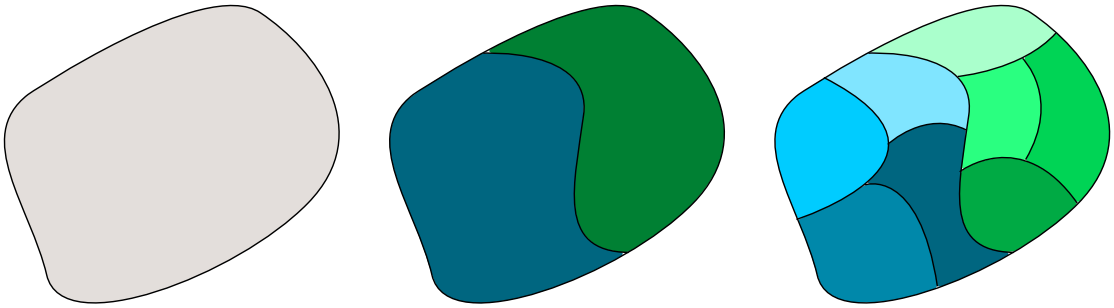
Element primitives are identified by the number of primitive faces N_f, vertexes N_n, face vertexes N_n^f, and dimensions N_d. Table 4.1 describes the primitives currently available in WARPXM along with their primitive identification number (PID). All element primitives in the current unstructured framework refer to three dimensional geometries. Lower primitive dimensionality refer to slab geometries in higher dimensions. For instance a slab triangle is a two dimensional primitive that describes a three dimensional triangular prism of unit height. This distinction is important for geometric quantities, such as volumes, surface areas, and Jacobian matrices.

Table 4.1: Primitive types allowed in WARPXM. Primitives are described by primitive ID (PID) and number of primitive faces N_f, vertexes N_n, face vertexes N_n^f, and dimensionality N_d.

Primitive	PID	N_f	N_n	N_d	N_n^f
Slab Line	0	2	2	1	1
Slab Triangle	1	3	3	2	2
Slab Quadrilateral	2	4	4	2	2
Tetrahedron	3	4	4	3	3
Hexahedron	4	6	8	3	4

4.2.1 Mesh partitioning

Since it is often impractical or slower for a single processor core in a cluster to handle computations over all subdomain elements, each subdomain is broken down into groups of interconnected elements known as patches. Partitioning the subdomains into patches is done entirely within WARPXM using a graph partitioning library, and the user has no direct control over the process. Each patch generated within WARPXM is fully described by a closed set of coordinate, connectivity, and neighborhood arrays and composition can range from a single element to the entire subdomain. Patches are designed as a compute interface which conforms to the compute device to which they are assigned. Multiple patches can exist on a compute device; however, to minimize communication between compute devices, each patch is only assigned to a single compute device. Figure 4.2.4 shows an example decomposition of a domain into subdomains and patches.



(a) Domain (b) Subdomains (c) Patches

Figure 4.2.4: Decomposition of domain into patches. (a) Domain of simulation. (b) Subdomain decomposition of domain into blue and green subdomains as defined by user. (c) Patch decomposition of subdomains into shades of blue and green as defined by partitioning scheme.

In WARPXM, mesh partitioning is designed to optimally break down a mesh into blocks based on criteria derived from the available compute device performances. Partitioning the patches among a cluster's compute devices is handled by the graph partitioning library METIS. Graph partitioning is the process of grouping a collection of interconnected points based on a set of criteria. For meshes, partitioning schemes attempt to maximize the compute volume defined by the number of elements in a patch, while minimizing the communication area, which is represented by the number of element faces separating the patches. WARPXM assigns weights to the compute devices it finds on a cluster based on their performance, which is used by METIS to impose a rough form of static load balancing.

4.3 Dataset configuration

Memory management is an important factor when developing efficient numerical solvers. In general, processing elements are extremely fast at floating point arithmetic compared to fetching memory. In order to maximize the computational efficiency of an algorithm, the dataset must be configured for quick access and update operations. All variables allocated in

WARPXM are held in datasets. Each dataset is mapped to a mesh, such that each element in a mesh relates to a contiguous chunk in the dataset, known as the dataset's element array. In the case of the nodal DGM, the element array would store the value of a solution for each node in the solution. For finite volume, each element array contains a single value representing the volume average in the element.

Given a dataset with an element array of size N_p (e.g. finite volume $N_p = 1$, or 2D discontinuous Galerkin method with six nodes per triangle $N_p = 6$), each element λ with index i_λ, the contiguous chunk in the dataset would be in the range $[i_\lambda \times N_p, (i_\lambda + 1) \times N_p)$. In terms of a solution q_p^n, where p is the node index and n is the element index the layout is

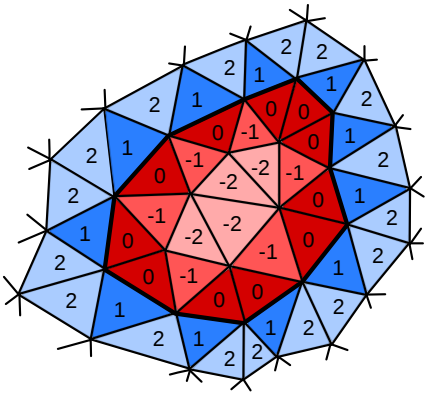
$$\underbrace{\left[\underbrace{\left[\underbrace{q_0^0}_{\text{Node 0}}, \underbrace{q_1^0}_{\text{Node 1}}, \cdots\right]}_{\text{Element 0}}, \underbrace{\left[\underbrace{q_0^1}_{\text{Node 0}}, \underbrace{q_1^1}_{\text{Node 1}}, \cdots\right]}_{\text{Element 1}}, \cdots\right]}_{\text{Dataset}}.$$

The order of the nodes in the dataset's element array is dependent on the element primitive and the numerical method.

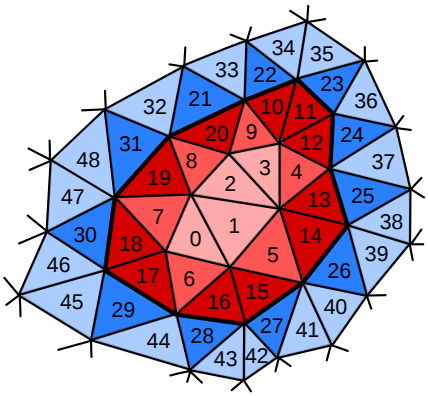
The dataset for a variable is broken into components, such as the mass density ρ in a moment model, which are held in separate component arrays. A component array is subdivided into subdomain arrays based on what subdomains in a domain the component exists on. Each subdomian array is further subdivided into patch arrays. Each individual patch array is associated with a patch on a mesh, and contains internal elements representing patch specific data, external elements used for representing data in neighboring patch, and virtual elements representing elements outside of the domain. Virtual elements do not have a physically relevant geometry associated with them, and are used as storage space for boundary condition data. Since each patch is associated with a single compute device, it allows the patch to exist in a stationary allocation on the compute device.

The order of the elements within the patch array is important when considering data access and update operations. The unique caching method within OpenCL does not directly ensure that having any two elements nearby one another within the dataset increases performance. Instead, the largest bottleneck in memory management is in accessing compute device datasets from the host level, which is required for synchronizing data between patches (i.e. ghost sync) and file I/O operations. The key restriction in dataset design is that the OpenCL API only allows contiguous access operations to datasets from the host level. For explicit time integration schemes, this ghost sync happens multiple times per time step. The bulk copy operations to and from host memory required for file I/O are generally uncommon, happening after hundreds or thousands of time steps. Optimizing the ghost sync operations are then the driving force in dataset layout.

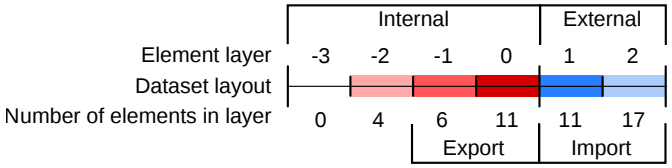
Ghost sync operations pull data from datasets to send to external datasets and push data gathered from external datasets. To take advantage of this operational layout, a layering structure is imposed which groups elements into import, export, and interior element sets based on their layer as depicted in Fig. 4.3.1. Internal elements represent all elements internal to the boundary and are required for file I/O operations. Export elements are a subset of internal elements that are required by external datasets. Import elements, or ghost elements, are elements that are updated by external datasets. Virtual elements, not shown in Fig. 4.3.1, are appended to the end of the patch array.



(a) Element layering by distance from boundary.



(b) Element indexes are grouped by layer, and begin in the lowest element layer.



(c) Dataset sectioning based on element layering

Figure 4.3.1: Dataset layout based on element layering using two layers. Elements are grouped based on distance from boundary (thick black line). Internal elements (red - element layers ≤ 0) represent the majority of elements in large meshes. Import/external elements (blue - element layers > 0) are elements existing in external patches. Export elements are a subset of internal elements that exist as import elements in external patches. Element layer -2 contains all internal elements that are not export elements. Element indexes are assigned based on layering in order to generate contiguous chunks in the dataset.

This layout means that only a single contiguous access operation is required to read/write the export/import elements from/to the host level. For this study, grouping elements into layers is done using a recursive path finding algorithm that starts from the boundary and counts the minimum number of elements it encounters as it travels to each element. As such, this algorithm becomes increasingly expensive as the number of element layers increases. To

counter the expense, only a small number of layers are ever generated. In this manner, the lowest layer specified contains the elements of the specified layer as well as all elements in lower layers. A possible optimization of this dataset layout would be to apply a space filling curve indexing scheme to the element indexes within each layer to promote element adjacency in the dataset, however this was not implemented for this research.

4.4 Solver discretization

Solvers in WARPXM represent a collection of modules that describe the evolution of an equation set. A Runge-Kutta time integration scheme tied to a discontinuous Galerkin numerical method is an example of a solver. Solvers are designed to operate on user specified subdomains in the mesh. Since a subdomain is broken into multiple patches, as shown in Fig. 4.2.4, a solver must handle each patch separately.

Solvers are designed to hide memory movement with computations. To facilitate efficient communication practices, patches are abstractly split up into interior and periphery regions as shown in Fig. 4.4.1. The periphery region contains the surface of the patch represented by the outer element layers (e.g. element layer 0) along with all faces connected to said element layers. The elements making up the interior layers (e.g. element layers ≤ -1) including all faces connected to said elements, but not connected to periphery elements, are collectively known as the interior. Breaking solvers into periphery and interior solvers allows for the costs associated with ghost sync processes to be hidden by the solver acting on the patch interior.

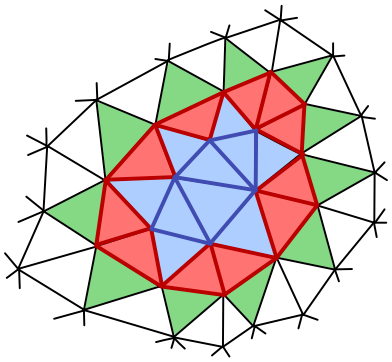


Figure 4.4.1: Breakdown of patch elements into interior (blue), periphery (red), and external (green) elements. Interior faces (blue lines) and periphery faces (red lines) are separated to expedite data transfers between patches.

In general, a time step from time t to time $t + \Delta t$ within an explicit time integration scheme is broken into stages. Each stage takes a solution $q(t + \alpha_i \Delta t)$ and uses it to predict a new solution $q(t + \alpha_{i+1} \Delta t)$, where α_i represents the time associated with a stage. Runge-Kutta (RK) time integration methods, discussed in Set. 3.4, can be generalized to the form

$$q(t + \alpha_{i+1} \Delta t) = \sum_{n=0}^{n \leq i} \beta_n^i q\left(t + \alpha_n \Delta t\right) + \gamma_i \Delta t \mathcal{L}\left(q\left(t + \alpha_i \Delta t\right)\right) \tag{4.4.1}$$

where γ_i and β_n^i represents the coefficients associated with a particular stage in an time step, and $\mathcal{L}$ represents the spatial discretization generated during each RK stage. Within WARPXM, each RK stage is further broken up into individual modules, outlined in Fig. 4.4.2, that handle boundary conditions, gradient solvers, limiters, and the time advance that generates q used in the next RK stage. The modular solver design is currently only implemented for the discontinuous Galerkin method, however, nothing in its design precludes it from being implemented for the finite volume method. The modules described in Fig. 4.4.2 are defined as follows:

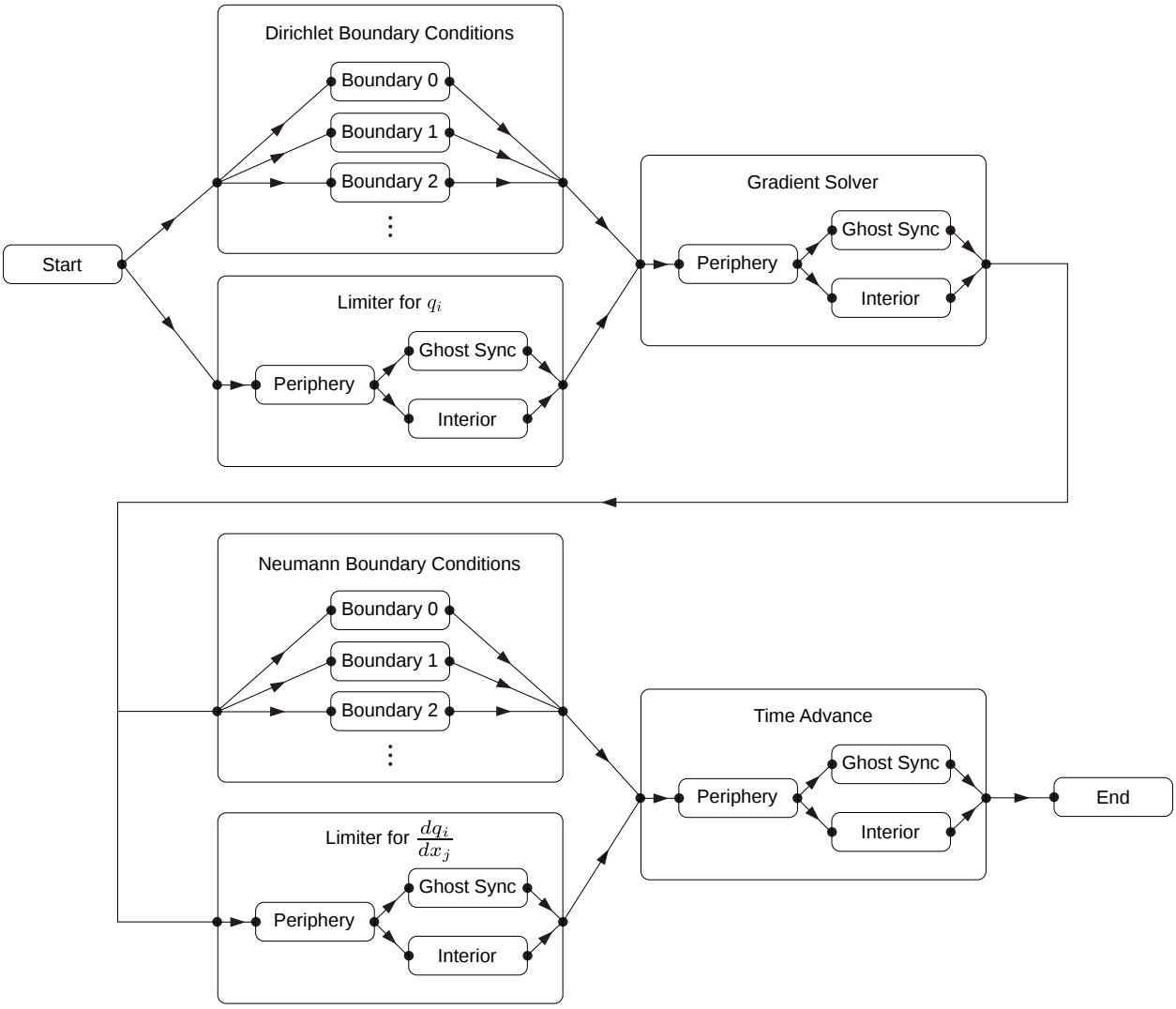


Figure 4.4.2: Dependency graph of solvers required for a single stage of an explicit Runge-Kutta time integration step.

- **Boundary conditions**:

  Boundary conditions for the discontinuous Galerkin method are handled using virtual elements constructed for each unconnected face on a patch. A virtual element has no geometry associated with it outside of the face geometry connecting the virtual element to the patch. This face geometry is used to describe a face normal into the

virtual element which can be used to specify boundary conditions. For example, if the boundary condition specifies that the flow normal to a face is zero, then the normal projection of the fluid's momentum is reversed across the face.

Dirichlet boundary conditions are directly set in the dataset as values found on the outside of the face. These boundary conditions are directly read into the numerical flux calculation for the discontinuous numerical method. Neumann boundary conditions are similarly constructed such that the value of the gradient is directly set on the outside edge of the domain's boundary. In the case where no boundary condition is specified, WARPXM is designed to automatically copy the value found on the inside of the face to the outside of the face. Since boundary conditions only act on virtual elements found outside of either the periphery or interior of a patch, each boundary condition can be applied asynchronously.

- **Limiting**:

  Applying limiters to a solution is a process required by discontinuous numerical methods to enforce monotonicity, as described in Sec. 3.2.2 and Sec. 3.3.3. There exist many monotonicity enforcement schemes, and while the method discussed herein is based on Sec. 3.3.3, the limiter module is designed to be swappable so that alternative monotonicity enforcement schemes can be implemented in the future. For the discontinuous Galerkin scheme, limiters can be run asynchronously with the boundary conditions module, however this may not be applicable for a finite volume method.

  Limiting a solution is handled in two stages. The first stage applies the limiting scheme to the values of q while the second stage applies limiters to the gradient of q. Whether the limiter needs to be applied to either q or the gradient of q is left up to the user. Since limiting schemes are generally expensive processes, the limiter module is broken into interior and periphery submodules. The limiter module used for each of the interior and periphery applications is shown in Fig. 4.4.3. The module is broken into two

kernels which first calculate the solution's volume average in each element and then uses the volume averages to generate a limited solution within each element. The two kernel formulation is required in order to synchronize the volume average calculation over the entire patch.

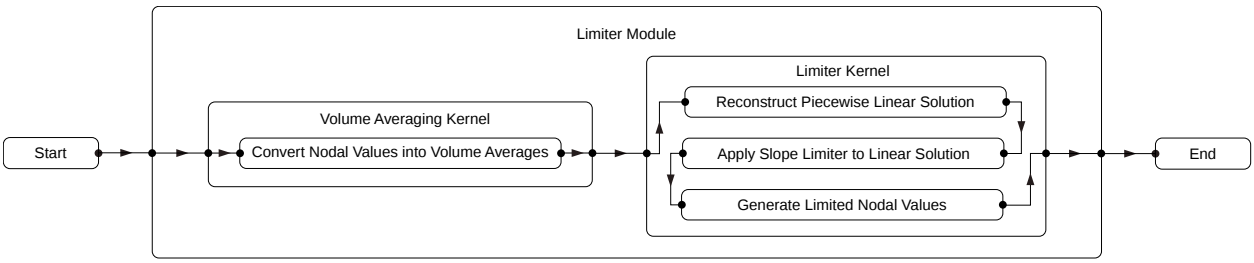


Figure 4.4.3: Dependency graph of limiter module made up of two separate kernels.

It is important to note that in order to apply this limiting scheme to elements in layer 0 (i.e. the periphery), the volume average solution must be generated for elements in layers -1, 0, and 1, which means that at least two element layers must exist in the mesh. This is due to how the piecewise linear reconstruction scheme is implemented.

- **Gradient solver**:

The gradient of the solution is generated from knowledge of the solution, and the geometry of the mesh. In the case of the finite volume method, gradients can be generated directly from the k-exact reconstruction (see Sec. 3.2.1), however for discontinuous Galerkin methods, the gradient operator (see Sec. 3.3.1) must be implemented using multiple kernels as shown in Fig. 4.4.4.

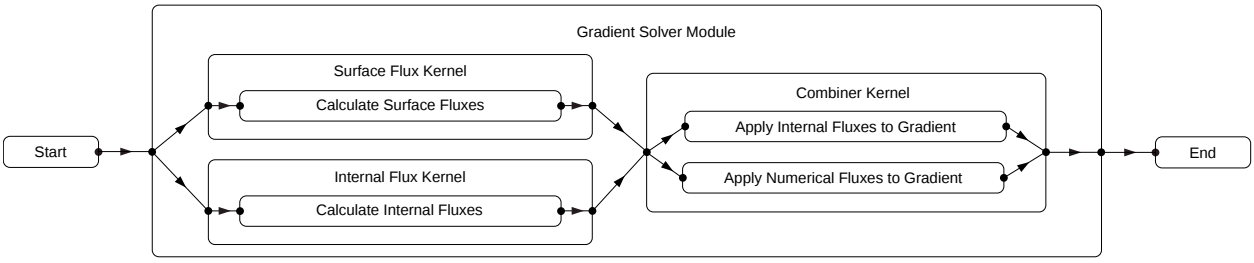


Figure 4.4.4: Dependency graph of gradient module made up of three separate kernels.

In terms of Sec. 3.3.1, the gradient of the solution at each element node, Y_{ij}^{λ}, is constructed from both a surface flux and an internal flux contribution. The surface flux kernel is designed to iterate over each face in its assigned range (i.e. periphery or interior), and generate the surface flux $\mathcal{Q}_{ijk}^{\lambda\gamma}$ which is stored in global memory. The internal flux kernel iterates through a range of elements, generating the internal fluxes q_{ij}^{λ}, and storing them in global memory. Once both the internal and surface fluxes are held in global memory, the coalescence kernel iterates through each element and combines the surface and internal fluxes as described in Eq. 3.3.42 to generate the solution's gradient. Since the internal flux calculation is independent of the surface flux calculation, the kernels can be run asynchronously. Unlike the limiter module, the gradient module only requires a single element layer.

- **Time advance**:

  The time advance module takes the limited solution, and its gradients, at a given time and predicts the solution at a later time. The methods discussed for this module are designed around explicit time integration methods, however the majority of the material can be adapted to implicit time integration methods. The time advance process is broken into two stages. The first stage, known as the spatial solver, generates the right hand side variable Γ from a solution q, while the second stage, known as the temporal solver, uses a combination of the solution q and the right hand side variable to

generate the solution at a later time, as described in Eq. (4.4.1). The functional layout of the discontinuous Galerkin method time advance module is pictured in Fig. 4.4.5, which is applied to the interior and periphery of the patch separately. The current finite volume method in WARPXM does not use this layout, but it can be adapted for it. The spatial solver is made up of three kernels, similar to gradient module, which are directly related to the formulation in Sec. 3.3.

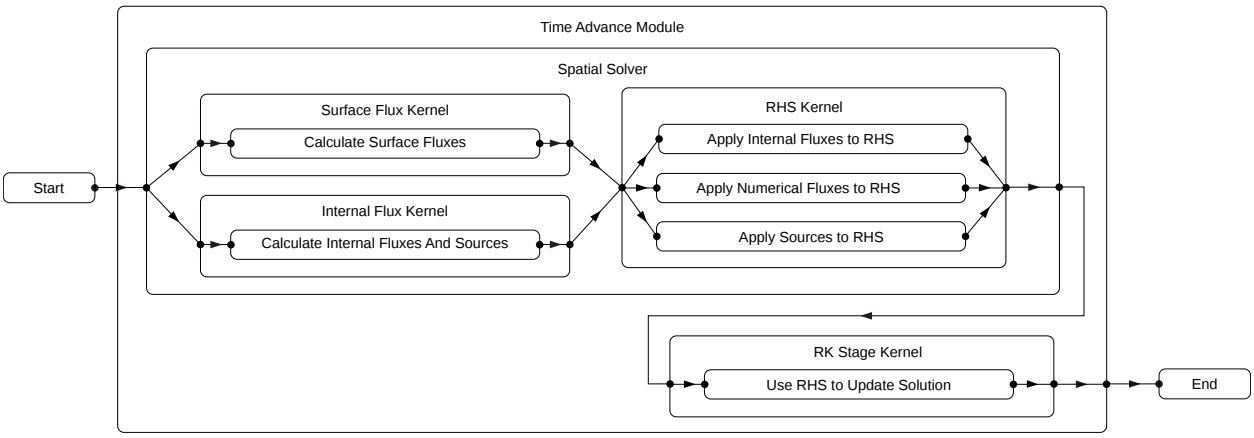


Figure 4.4.5: Dependency graph of time advance module made up of four separate kernels.

The surface flux kernel iterates over the faces in its prescribed region of the mesh (i.e. interior or periphery), and generates and stores the numerical fluxes $\mathcal{F}_{ij}^{\lambda\gamma}$. The face iteration is handled at the workgroup level, meaning that the individual kernels are applied to individual or groups of face nodes making up the face. Linking the workgroup to a face allows the local memory space to contain face centric data required by the numerical flux solvers, such as the face normal vector, which is then shared among the individual kernels acting on the face's nodes. This greatly reduces the memory latency for surface flux calculations on multidimensional meshes.

The internal flux kernel iterates over the elements in a mesh to generate and store the internal fluxes f_{ij}^{λ} as well as the source terms s_i. Similar to the surface flux workgroups,

the internal flux workgroups are associated with the elements, and the kernel is applied to groups of element nodes. The combination of the source and internal flux terms in the internal flux kernel attempts to make the most of cache locality within an element, with the drawback of requiring the sources and fluxes to be expanded on the same basis ($\psi_i = \theta_i$).

The equation sets defining a model are implemented sequentially within the surface and internal kernels. For example, consider a two-fluid plasma model containing a 5-moment description of ions and electrons as well as Maxwell's equations. In this case, the surface flux kernel iterates through the basis nodes within an element and, at each node, calculate the flux associated with the ions, then the electron flux, then the flux associated with Maxwell's equations. Once the flux calculations are complete for a given node, the Lorentz force source terms for the ions are calculated, then for the electrons, and finally the current density source terms are generated for Maxwell's equations. The flux/source terms are combined within the kernels, as opposed to having a separate kernel for each flux/source term, to reduce overhead associated with loading face or element specific geometry into local and private memory. This design also increases cache reusability for cases where both multiple flux/source terms require globally stored data. For example, both the 5-moment model flux and Lorentz force operator require knowledge of the fluid's local mass and momentum densities. The main drawback to this layout is that the OpenCL compiler may fail to properly vectorize large equation sets, which increases simulation runtime.

The right-hand-side (RHS) kernel combines the fluxes and source terms together into a right-hand-side variable as described in Eq. 3.3.21. As before the RHS workgroups are associated with elements, which allows for the local storage of internal fluxes, numerical fluxes, and source terms prior to their application in Eq. 3.3.21. While this is an optimal design for small equation sets such as MHD, the local storage size requirements for large multidimensional equation sets such as multi-species 13-moment model acting with a

high order basis containing dozens of nodes is a major limitation. For CPUs this is generally not an issue as there is plenty of memory available, however most GPUs have a limited cache size, and the RHS kernel can fail to compile and/or run.

The RK stage kernel uses the RHS variable generated by the spatial solver to generate the solution for the next RK stage. Unlike the spatial solver kernels, there is no geometry associated with time integration, meaning that instead of iterating over elements or faces, the RK stage kernel iterates over the nodes in its prescribed region and solves Eq. (4.4.1). Since this is a purely a multiply-and-add operation, it is easy for the OpenCL compiler to optimize the kernel. Similar to the gradient module, only a single element layer is required for the time advance module.

Chapter 5

RESULTS

This section overviews a collection of results obtained from the plasma models discussed in Sec. 2 using the finite volume and discontinuous Galerkin methods discussed in Sec. 3. Since the 13-moment fluid model is still largely untested in literature, the first set of test cases discussed in Sec. 5.2 check how well the base neutral-fluid model captures collisional transport. The full 13-moment plasma model is tested in Sec. 5.3 with the goal of identifying where further development is needed. The simulation input files listed in the following sections are designed to work in WARPXM in versions prior to June, 2016.

The parabolic cleaning operator used for Maxwell's equations is compared against the hyperbolically cleaned Maxwell's equations in Sec. 5.4. The goal of the cleaning tests is to show the locality of parabolic operator, and how it can be implemented for problems where divergence errors are difficult to control. Section 5.5 summarizes the results and discusses paths for further development. Before discussing any results obtained from the WARPXM code, it is important to test the convergence accuracy of its numerical models.

5.1 Convergence analysis

The convergence accuracy for both the discontinuous Galerkin and temporal discretizations are tested in the following sections. Convergence accuracy in *hp* style methods, such as finite element methods or predictor corrector schemes, is specified by the dependency of deviation of a simulation solution $\bar{q}$ from a known solution q based on the spatial/temporal step size

h. The error takes the approximate form

$$\epsilon = |q - \bar{q}|_2 \approx \epsilon_0 h^p, \tag{5.1.1}$$

where $|\cdot|_2$ represents an L_2 norm and p is the order of convergence. To test convergence order, errors are calculated from simulations run with various values of h, and the convergence accuracy is defined by the slope

$$p = \frac{\Delta \log(\epsilon)}{\Delta \log(h)}. \tag{5.1.2}$$

5.1.1 Discontinuous Galerkin method

The convergence order for the discontinuous Galerkin method was tested against the linear advection equation

$$\frac{\partial}{\partial t} q = -\frac{\partial}{\partial x_i}(v_i q). \tag{5.1.3}$$

The velocity was set to $\vec{v} = [1, 0, 0]$, while the solution was initialized with a single wavelength of a sine wave

$$q(\vec{x}, t = 0) = \sin\left(\frac{2\pi}{L} x\right) \tag{5.1.4}$$

over a domain length of $L = 1$ such that $x \in \left[-\frac{1}{2}, \frac{1}{2}\right]$. Periodic boundaries are used to ensure that the wave propagation is smooth across the boundaries. The solution to the linear advection problem, assuming a constant velocity, is

$$q(\vec{x}, t) = q(\vec{x} - \vec{v}t, 0) = \sin\left(\frac{2\pi}{L}(x - v_x t)\right). \tag{5.1.5}$$

The simulations were run from $t = 0$ to $t = T$ using a consistent time step for every node

set. The RK4 time integration scheme was used due for its high-order convergence accuracy which ensures that the L_2 norm is dominated by spatial discretization errors. Since the spatial solver's convergence accuracy should be invariant to the number of time steps in a simulation, only a single time step is taken for the convergence test, such that $T = \Delta t$.

The spatial discretization error associated with the nodal discontinuous Galerkin method,

$$\epsilon = \sqrt{\frac{1}{N_n} \sum_{n=0}^{n<N_n} \left(q(\vec{x}_n, T) - \sin\left(\frac{2\pi}{L}(x_n - v_x T) \right) \right)^2}, \tag{5.1.6}$$

is based on a relative L_2 norm where N_n is the total number of nodes in the domain, and $\vec{x}_n$ is the position of node n. The convergence results for the nodesets described in Sec. 3.3.2 are shown in Fig. 5.1.1 for slab line elements, Fig. 5.1.2 for slab triangle elements, and Fig. 5.1.3 for tetrahedral elements. Tests are run for piecewise linear, quadratic, cubic, and quartic polynomial bases which are described by two, three, four, and five nodes along each dimension respectively. The figures show the expected increase in convergence order depending on the basis polynomial order.

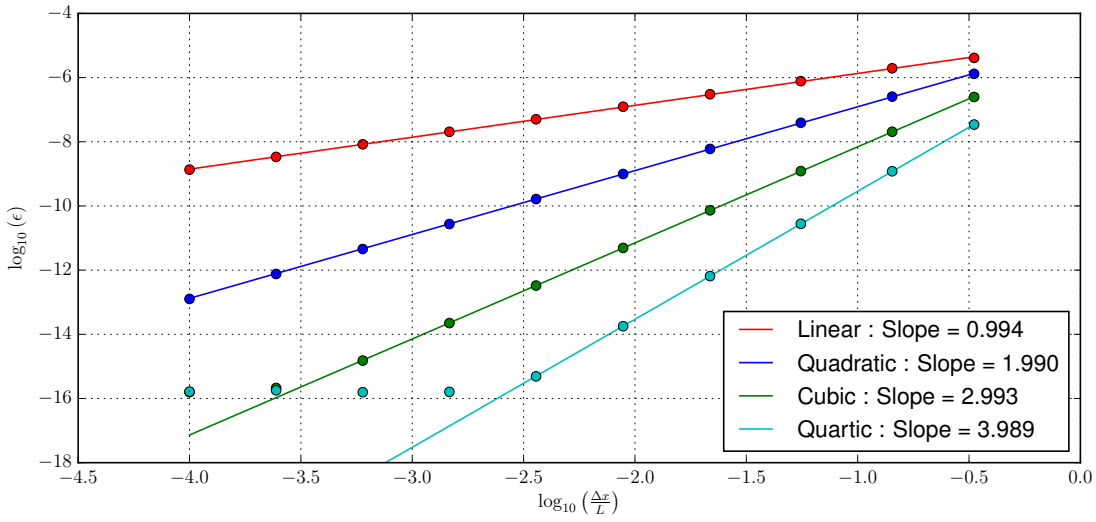


Figure 5.1.1: Linear advection convergence tests for nodal arrangements using the discontinuous Galerkin method with slab line elements. As expected, each additional node increases the convergence accuracy by a single order.

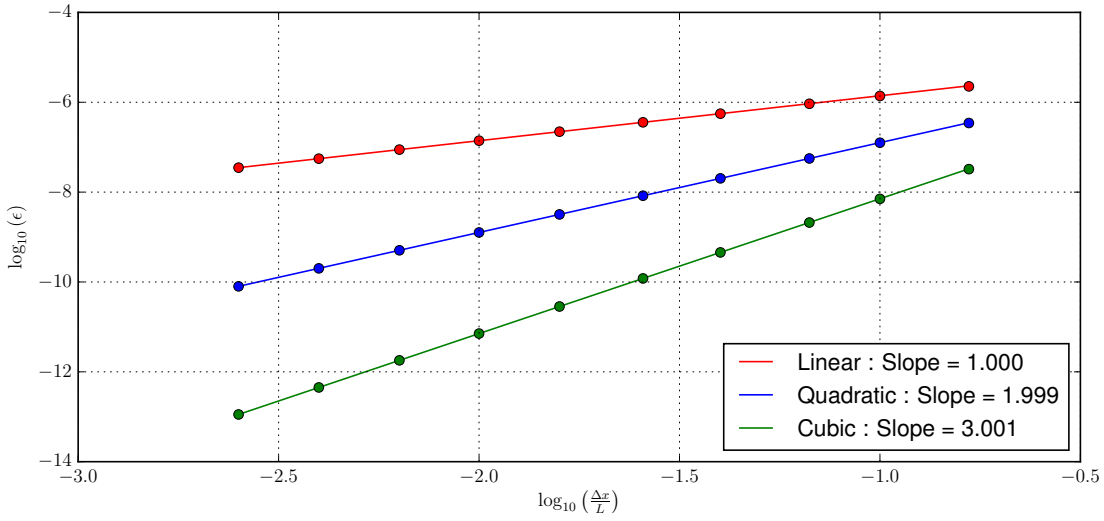


Figure 5.1.2: Linear advection convergence tests for nodal arrangements using the discontinuous Galerkin method with slab triangle elements. Increasing the number of nodes along each dimension increases the convergence accuracy by one order.

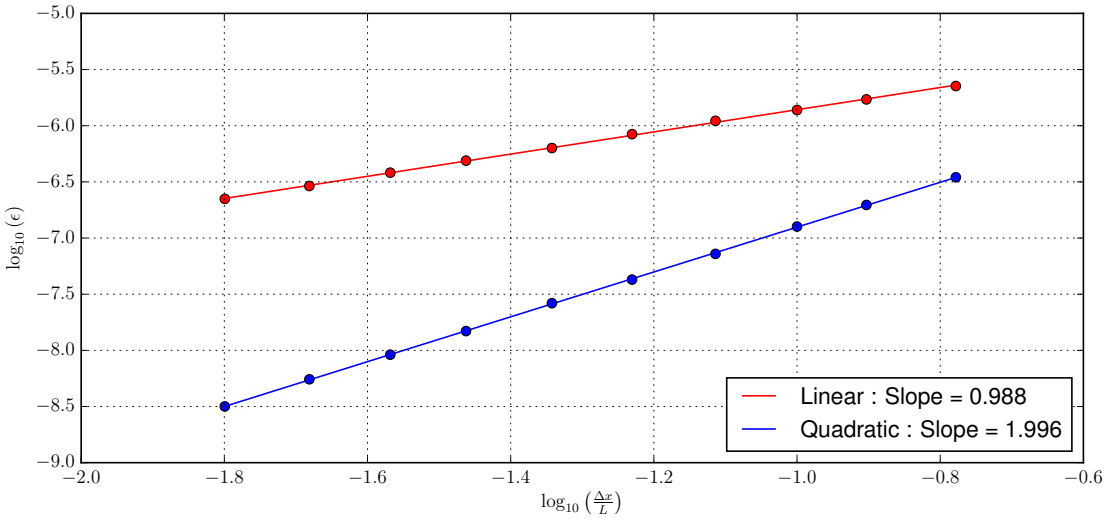


Figure 5.1.3: Linear advection convergence tests for nodal arrangements using the discontinuous Galerkin method with tetrahedral elements. Increasing the number of nodes along each dimension increases the convergence order by one.

5.1.2 Explicit time integration

The Runga-Kutta methods discussed in Sec. 3.4 are designed to increase their convergence order depending on the number of stages involved. To test the convergence order for the Runge-Kutta schemes implemented in WARPXM, a decay function,

$$\frac{d}{dt}q = -q, \tag{5.1.7}$$

is solved with an initial condition $q(t = 0) = 1$. The analytical solution to the decay problem for any time t is then

$$q(t) = e^{-t}. \tag{5.1.8}$$

The convergence test is run from $t = 0$ to $t = T$, and the error is calculated as

$$\epsilon = |q(T) - e^{-T}|. \tag{5.1.9}$$

The data given in Fig. 5.1.4 is given for an end time of $T = 10$. The time step is staggered by

$$\Delta t = 2^{-n}T, \tag{5.1.10}$$

with positive integer n, to ensure that the simulation ends at the proper time. The discontinuous Galerkin method is used as the spatial solver in this test, however, the error associated with the spatial directization can be ignored since the solution q is just negated and sent to the time integration solver. The results show the expected convergence orders for the implemented Runge-Kutta schemes, however it is interesting to note that the SSPRK3 method and RK4 methods have abnormally high noise floors. The reason for the high noise floors is unknown.

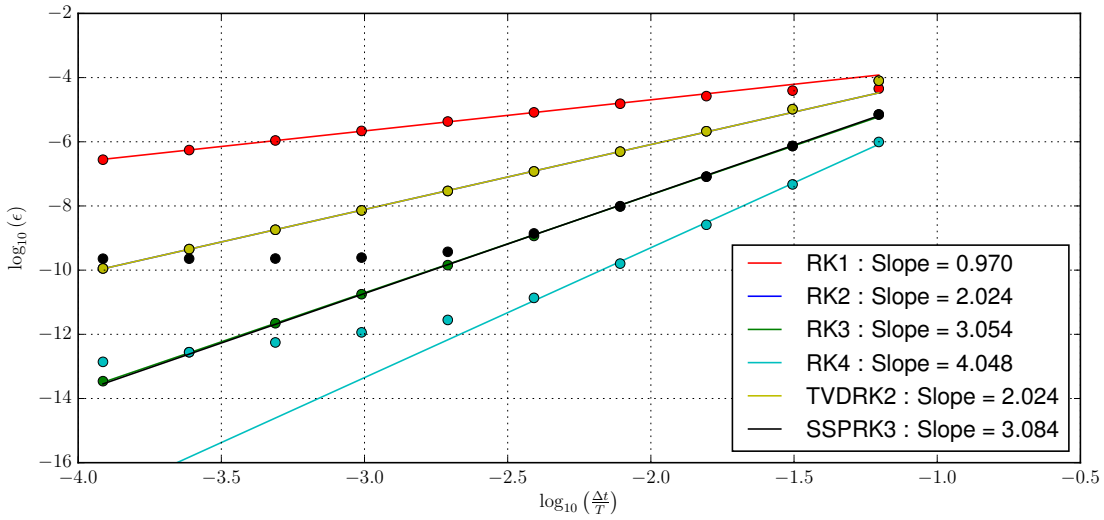


Figure 5.1.4: Temporal convergence tested against the decay function using the discontinuous Galerkin method. Each RK method shows its expected convergence order, however the noise floor for the RK4 method and the SSPRK3 method are higher than expected. Note that the RK2 and TVDRK2 shemes overlap perfectly.

5.2 Neutral flow results

5.2.1 Shock tube

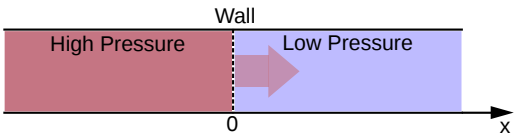


Figure 5.2.1: Representation of shock tube problem. High pressure fluid moves into low pressure area generating rarefaction wave in high pressure region and shock wave in low pressure region. The resulting shock profile is highly dependent on collisional interactions.

The isothermal shock tube is a benchmark for analyzing the effect of collisionality on shock structures. The test is initialized with a high pressure region separated from low pressure region with a wall as shown in Fig. 5.2.1. When the wall is removed, the high pressure fluid drives a shock wave into the low pressure region. For highly-collisional fluids, the particles congregate into three steep interfaces known as a shock profile. As the collisionality is lowered, particles travel further before thermalizing and the shock profile becomes diffuse.

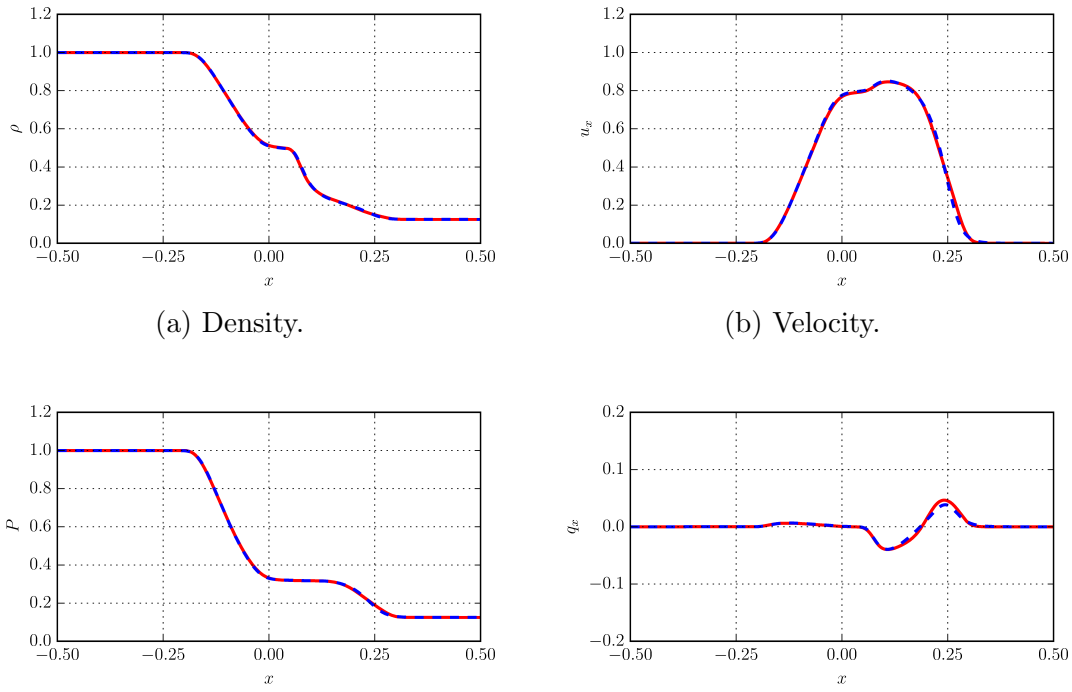
The shock tube for this study is initialized as a single, thermally equilibrated neutral fluid with a uniform temperature of $T = 1$ across a domain of $x \in [-\frac{1}{2}, \frac{1}{2}]$. The domain is split into a left (l) and right (r) regions by an interface at $x = 0$. The initial density conditions for the left and right state are given by $\rho_l = 8\rho_r = 1$ with an initial velocity $\vec{v} = 0$. The wall separating the pressurized regions is removed at time $t = 0$, and the simulation is run to a time $t = 0.12$. The collisionality is controlled by the Knudsen number $\mathrm{Kn} = (\nu_p\tau)^{-1}$ and a normalized collision frequency,

$$\nu = \frac{n}{T^{3/2}}, \tag{5.2.1}$$

related to Eq. (2.1.10). This definition for the collision frequency splits the domain into two collisional zones. In the left state, where the density is larger, the effective rate of thermalization, $(\nu_p\tau)\nu$, is a factor of 8 higher than in the right state. This implies that the shock wave, which moves to the right, exists in a state of lower collisionality than is set by the Knudsen number. The reason for doing this is to show the limitations of the 13-moment model's closure, and the reason behind the diffusive stabilization operator. These results were generated in the scripts **examples/dg/5-moment/shock_tube.py** for the 5-moment model and **examples/dg/13-moment/shock_tube.py** for the 13-moment model. The Boltzmann-BGK model was solved outside of WARPXM.

For highly collisional systems, the 13-moment model's closure is dominated by the BGK collision operator which drives convergence to the 5-moment fluid model. Figure 5.2.2 shows

an example for a collisionality of Kn $= 10^{-3}$ which is below the collisional transition regime. For this example, the 13-moment solution (red) does not include diffusive stabilization, and shows good agreement with the kinetic result (dashed blue) derived from the Boltzmann-BGK model. The reduced collisionality in the $x > 0$ region results in a small deviation in the 13-moment heat flux compared to the 5-moment model.



(a) Density.

(b) Velocity.

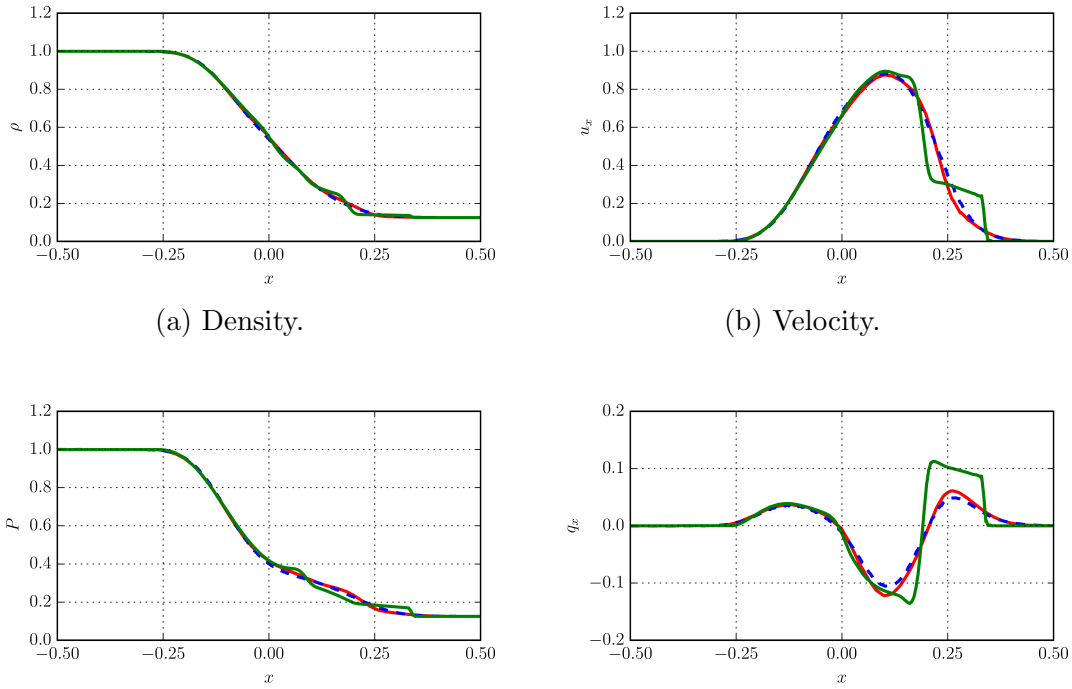
(c) Pressure.

(d) Heat flux.

Figure 5.2.2: Single-species isothermal shock tube comparison between 13-moment model (red) and Boltzmann-BGK model (dashed blue) for a collisionality of Kn $= 10^{-3}$ at normalized time $t = 0.12$. Strong agreement is seen for the density, velocity and pressure profiles, while the heat flux shows slight deviations due to the decreased collisionality in the $x > 0$ domain.

At weaker collisionalities the effect of the BGK collision operator is reduced which reveals the artificial waves of the underlying Pearson-IV closure. This effect is shown in the green trace of Fig. 5.2.3 for a moderately collisional test case of Kn $= 10^{-2}$ which is on the border

of the collisional transition regime. When the 13-moment fluid model includes the diffusive stabilization operator (red) it better matches the Boltzmann-BGK model (dashed blue) in the weaker collisionality region. These results express the need of the diffusive stabilization operator in moderately and weakly collisional systems undergoing strong, nonlinear dynamics in order to damp the artificial waves of the Pearson-IV closure.



(a) Density.

(b) Velocity.

(c) Pressure.

(d) Heat flux.

Figure 5.2.3: Single-species isothermal shock tube comparison between 13-moment model and Boltzmann-BGK model (dashed blue) for a moderate collisionality of Kn $= 10^{-2}$ at normalized time $t = 0.12$. The green line represents the 13-moment model solution without diffusive stabilization, while the red line shows the corrective behavior of the diffusive stabilization operator when the collisionality is reduced.

5.2.2 Subsonic flow past cylinder

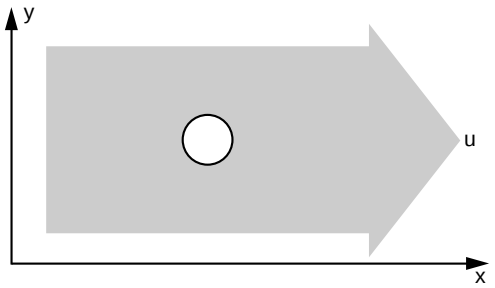


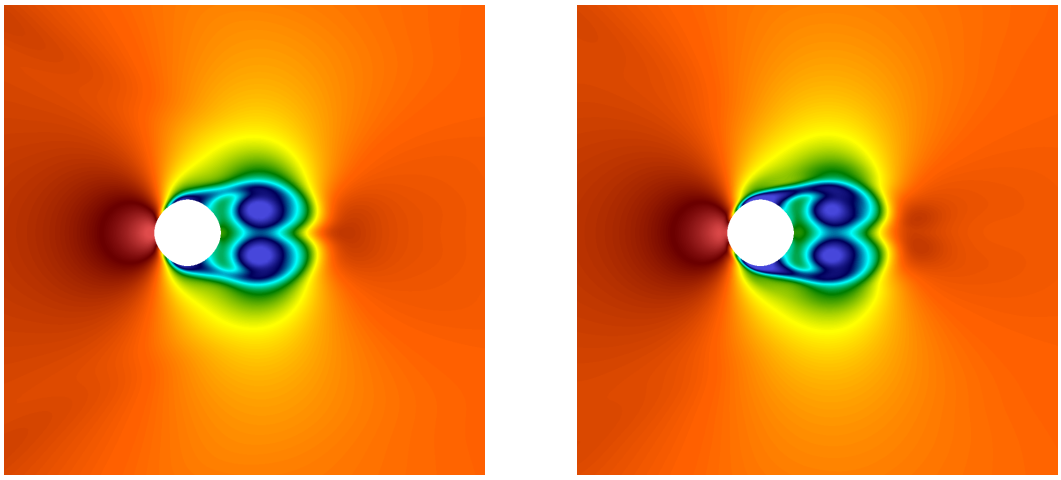
Figure 5.2.4: Fluid traveling in positive x direction at speed u interacts with a no-slip cylinder.

Subsonic flow past a cylinder, depicted in Fig. 5.2.4, is a two dimensional viscous CFD problem that tests the interaction between a subsonic fluid flow against a no-slip, adiabatic boundary condition. In this test the 13-moment fluid model is compared to the 5-moment fluid model from Sec. 2.3. As the flow passes the cylinder, vortexes form of the top and bottom of the lee side of the cylinder, which eventually interact with each other, resulting in vortex shedding[53]. The goal of the test is to show that the fully hyperbolic collisional transport of the 13-moment fluid model is accurate and stable in multiple dimensions.

The cylinder has diameter of 1 and is centered at position $(-6, 0, 0)$ in a domain of $x \in [-12, 12]$ by $y \in [-6, 6]$. The fluid is initially in thermal equilibrium ($T = 1$) with $\rho = 1$ and $\vec{u} = \begin{bmatrix} 0.5 & 0 & 0 \end{bmatrix}^T$. These initial conditions define a Mach 0.38 flow. The right, top, and bottom boundary conditions are zero-normal gradient for all fluid components, while the left boundary is an inflow boundary condition consistent with the initial conditions. The mesh is not designed to fully resolve the collisional dynamics ($\Delta x \approx 10^{-2}$), but to give a comparison between fluid models.

Figure 5.2.5 shows the relative state of the different models at time $t = 8$ for a Knudsen number of $\mathrm{Kn} = 2 \cdot 10^{-3}$ and a collision frequency of $\nu = 1$. The effective Reynolds number is $\mathrm{Re} = 250$. Both models give approximately the same result in terms of the dimensions of the

vortexes and the magnitude of the density. The sharper vortex structures seen with the 13-moment model is attributed to the 13-moment closure's artificial wave structures. Without the diffusive stabilization operator, the stability of the 13-moment fluid model was found to be dependent on the collisionality. As the collisionality is lowered the artificial waves become sharper which, under transonic flow conditions, result in a loss of hyperbolicity.



(a) 5-moment model (b) 13-moment model

Figure 5.2.5: Density comparison between (a) 5-moment and (b) 13-moment models for vortex development in 0.38 Mach flow past cylinder at time $t = 8$ and collisionality Kn $= 2 \cdot 10^{-3}$. Vortex profiles are comparable signifying that the hyperbolic collisional transport of the 13-moment model is physically accurate in multidimensional simulations. Slight deviations between profiles are attributed to the artificial wave structure of the 13-moment model's closure.

5.3 Charged flow results

5.3.1 Electromagnetic shock tube

The two-fluid electromagnetic shock tube is a generalization of the MHD Brio-Wu problem. The problem is initialized similar to the shock tube in Sec. 5.2.1 with additional magnetic

field components:

$$
\begin{aligned}
n_e &= n_i = \phi(x) \\
\vec{u}_e &= \vec{u}_i = 0 \\
T_e &= T_i = 1 \\
\vec{E} &= 0 \\
\vec{B}_l &= \begin{bmatrix} \frac{3}{4} & 1 & 0 \end{bmatrix}^T \\
\vec{B}_r &= \begin{bmatrix} \frac{3}{4} & -1 & 0 \end{bmatrix}^T
\end{aligned}
$$

The spatially dependent function

$$\phi(x) = \left(\frac{7}{8}\right) \frac{1 - \tanh\left(\frac{2\pi}{\lambda}x\right)}{2} + \frac{1}{8} \tag{5.3.1}$$

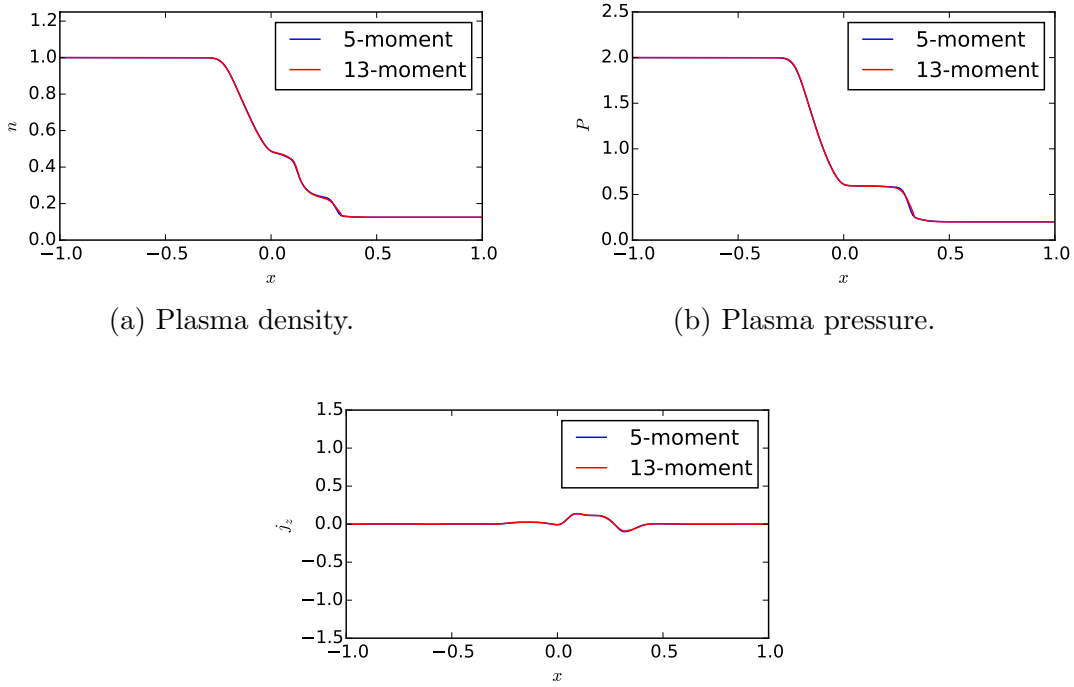
acts to smooth the interface, making shock transition easier to resolve. The interface transition width λ is set smaller than the size of the domain. The initial magnetic field in the y directions has a discontinuity at $x = 0$ which shocks along the x-axis at a speed dependent on the level of magnetization ($\omega_c\tau$). As B_y travels outward from the center of the domain it generating a large electric field in the z direction which acts to drive current. For weakly magnetized plasmas, the fluids ignore the magnetic field altogether, and the problem becomes an electrostatic shock tube.

For the following tests, the ions represent a proton species ($A_i = 1$, $Z_i = 1$), while the electrons have a non-realistic mass ($A_e = 10^{-2}$, $Z_e = -1$) to help reduce runtime. The level of magnetization ranges from $(\omega_c\tau) = 1$, representing a weakly magnetized plasma, to $(\omega_c\tau) = 10$, representing a moderately magnetized plasma. The reference collision frequency is set at $(\nu_p\tau) = 10^3$ to keep the Hall parameter small ($\omega_c \ll \nu_p$) which helps reduce the magnetoviscous effects captured in the 13-moment model that are ignored in the 5-moment model. The speed of light $c/v_0 = 100$ is kept small to decrease runtime, however it is well

above the electron thermal velocity. Collisions between species are ignored to make the results more comparable to the findings in Srinivasan and Shumlak [15]. These results were generated in the scripts **examples/dg/plasma/brio_wu/brio_wu_5moment.py** for the 5-moment model and **examples/dg/plasma/brio_wu/brio_wu_13moment.py** for the 13-moment model.

The simulation is run in a one dimensional domain of size $x \in [-\frac{3}{2}, \frac{3}{2}]$ and is run for a time span of $t \in [0, 0.12]$. The plasma density $n = (\rho_e + \rho_i)/(A_e + A_i)$, plasma pressure $P = P_e + P_i$, and current density $j_z = Z_e n_e u_z^e + Z_i n_i u_z^i$ are compared between the 5-moment and 13-moment plasma models.

The results for a weakly magnetized plasma, $(\omega_c \tau) = 1$, are shown in Fig. 5.3.1. The figure shows good agreement between the 5-moment model and 13-moment model. The weakly magnetized results are comparable to the neutral flow problem since the electrostatic coupling between the ions and electrons do not drive strong anisotropic effects.



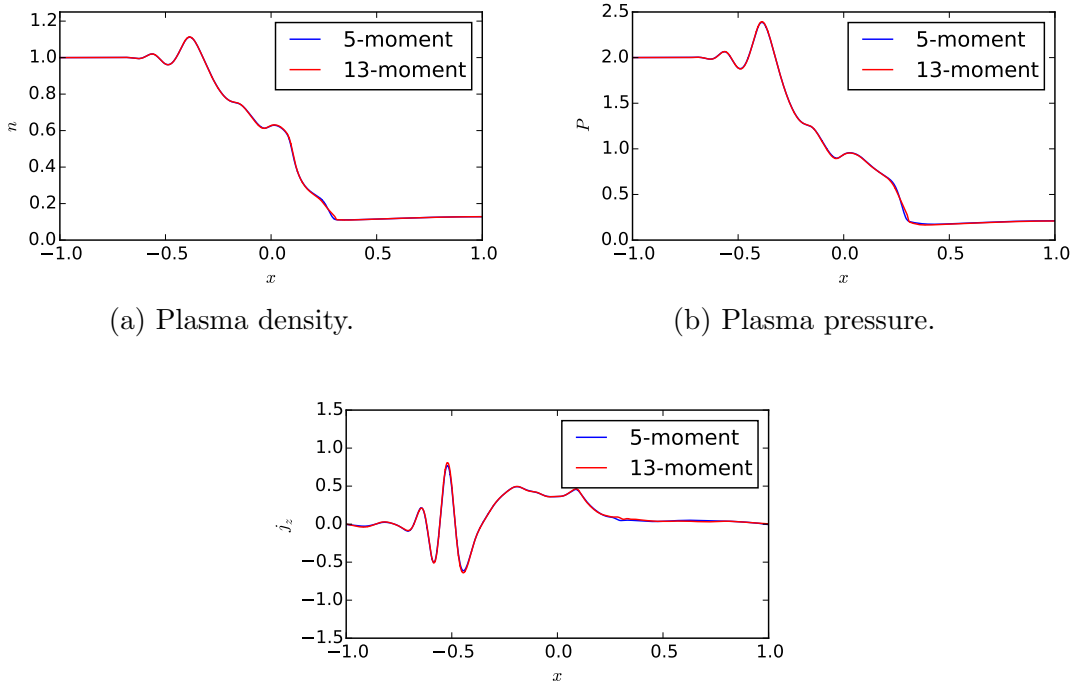
(a) Plasma density.

(b) Plasma pressure.

(c) Current density.

Figure 5.3.1: Electromagnetic shock tube with parameters $(\nu_p\tau) = 10^4$, $(\omega_c\tau) = 1$, $(c/v_0) = 10^2$, and $(\omega_p\tau) = 10^4$. Ion have mass $A_i = 1$ and charge $Z_i = 1$, while electrons have mass $A_e = 10^{-2}$ and charge $Z_e = 1$. Interspecies collisions are ignored and transition width is set at $\lambda = 0.01$. Simulation is run to time $t = 0.12$. Results show good agreement between the 5-moment and 13-moment plasma models.

Increasing the magnetization to $(\omega_c\tau) = 10$, see Fig. 5.3.2, also shows good agreement between the 5-moment model and 13-moment model. At this point, whistler waves are seen both upstream and downstream of the shock interfaces, which is consistent with the findings of Srinivasan and Shumlak [15]. Note that these results differ from those found in Srinivasan and Shumlak [15] purely due to the inclusion anisotropic pressure and heat flux effects, which is minimal due to the large collision frequency $(\nu_p\tau) = 10^4$.



(a) Plasma density.

(b) Plasma pressure.

(c) Current density.

Figure 5.3.2: Electromagnetic shock tube with parameters $(\nu_p\tau) = 10^4$, $(\omega_c\tau) = 10$, $(c/v_0) = 10^2$, and $(\omega_p\tau) = 10^4$. Ion have mass $A_i = 1$ and charge $Z_i = 1$, while electrons have mass $A_e = 10^{-2}$ and charge $Z_e = 1$. Interspecies collisions are ignored and transition width is set at $\lambda = 0.01$. Simulation is run to time $t = 0.12$.Results show good agreement between the 5-moment and 13-moment plasma models.

5.3.2 Langmuir dispersion

Both the 5-moment and 13-moment plasma models are used to describe a species behavior in the presence of an electromagnetic field. Langmuir oscillations are an example of electrostatic plasma oscillations that are generated from the electrons in a plasma oscillating around a background ion species due to an initial density perturbation. Since the Langmuir oscillation problem is designed around a static ion species, the ion species is initialized in thermal

equilibrium, and is be evolved in time.

The ions and electrons are initializes in a quasineutral state ($n = n_e = Z_i n_i$) in thermal equilibrium $T = T_e = T_i$, without any initial flow $\vec{u}_e = \vec{u}_i = 0$. The density is initialized with a smooth transition from a high density region to a low density region, similar to what was used for the electromagnetic shock problem in Sec. 5.3.1,

$$n = n_0 \left((1-\beta) \frac{1 - \tanh\left(\frac{2\pi}{\lambda} x\right)}{2} + \beta \right), \tag{5.3.2}$$

which transitions from $n = n_0$ in the $x < 0$ region to $n = \beta n_0$ in the $x > 0$ region, over a smoothing region of width approximated by λ. Since this initial condition is not in pressure balance, the electrons move along the pressure gradient, generating an electrostatic field due to their charge separation from the static ions. This electrostatic field pulls the electrons back toward the ions, creating Langmuir oscillations. The dispersion relation for Langmuir oscillations can be found in most electrostatic plasma literature[28] as

$$\omega^2 = \left(\omega_p^e\right)^2 + \gamma k_x^2 \bar{u}_e^2 \tag{5.3.3}$$

where $\gamma = 5/3$ is the adiabatic index, $\bar{u}_e^2 = v_0 T_e / A_e$ is the electron thermal velocity, and $(\omega_p^e)^2 = \omega_p^2 n_e / A_e$ is the normalized electron plasma frequency. This can be rewritten as

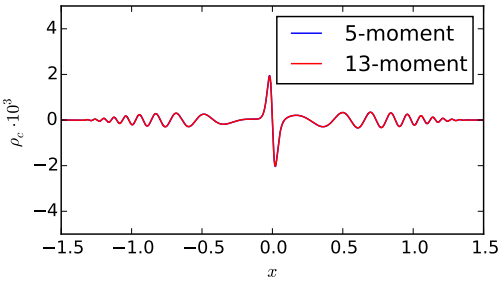
$$\left(\frac{\omega}{\omega_p^e}\right)^2 = 1 + \gamma \left(k_x \lambda_d^e\right)^2 \tag{5.3.4}$$

where $\lambda_d^e = \bar{u}_e / \omega_p^e$ is the normalized electron Debye length. Since the plasma frequency is density dependent, the initial jump in plasma density in Eq. (5.3.2) described by β must be kept small, meaning $\beta \approx 1$. For all cases in this section, the jump factor is kept at $\beta = 0.95$. All of the following test cases have a set plasma frequency $(\omega_p \tau) = 100$ and electron mass $A_e = 10^{-2}$. The speed of light is set at $c/v_0 = 100$ and the reference cyclotron frequency is $(\omega_c \tau) = 1$, however since this is an electrostatic problem, setting these

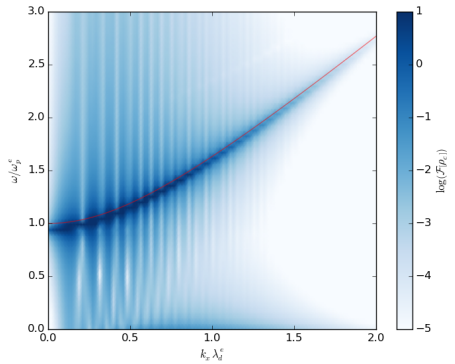
values has no impact on the results. These results were generated in the scripts **examples/dg/plasma/dispersion/langmuir_oscillation_static_ions_5moment.py** for the 5-moment model and **examples/dg/plasma/dispersion/langmuir_oscillation_static_ions_13moment.py** for the 13-moment model.

Note that assuming $\gamma = 5/3$ in Eq. (5.3.4) is a isotropic fluid approximation to the Langmuir dispersion relation, and in the more physically relevant collisionless case, where $\nu_p \ll \omega_p$, the dispersion relation assumes $\gamma = 3$. Even though the highly-collisional Langmuir oscillation regime does not reflect physical plasma dynamics, it is a good test to compare the electrostatic interactions between the Lorentz force operators and the hyperbolic fluxes of the 5-moment and 13-moment plasma models. Since the goal is to compare the 5-moment and 13-moment models using electrostatic waves, interspecies collisions and diffusive stabilization are not used.

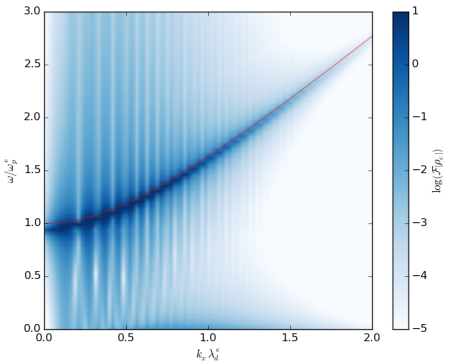
Figure 5.3.3 shows that both the 5-moment and 13-moment plasma models agree with the Bohm-Gross law for highly-collisional fluids of $(\nu_p \tau) = 10^4$. This is expected since the 13-moment model is designed to converge to the 5-moment model in this collisional regime.



(a) Comparison of 5-moment and 13-moment charge densities.



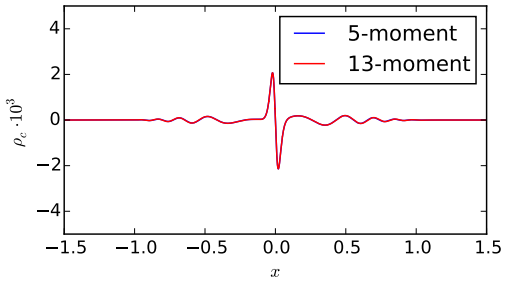
(b) Comparison of 5-moment results (blue) to theory (red).



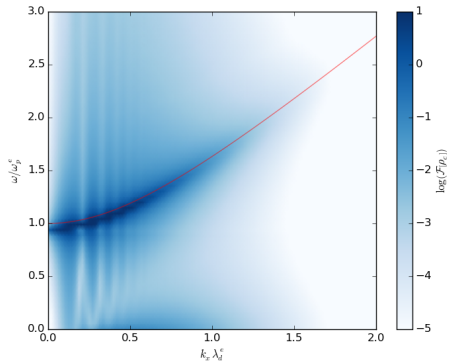
(c) Comparison of 13-moment results (blue) to theory (red).

Figure 5.3.3: Comparison between 5-moment and 13-moment plasma models for the Langmuir oscillation problem with $(\nu_p \tau) = 10^4$. Both the 5-moment and 13-moment models show good agreement with the theoretical results.

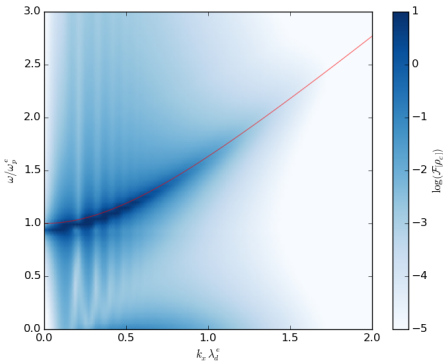
Figure 5.3.4 shows a comparison between the 5-moment model and the 13-moment model for a reduced collisionality of $(\nu_p \tau) = 10^3$. The result is similar to the increased collisional case, however the vertical striations in Fig. 5.3.3 are largely damped due to the viscous effects in the fluid models.



(a) Comparison of 5-moment and 13-moment charge densities.



(b) Comparison of 5-moment results (blue) to theory (red).

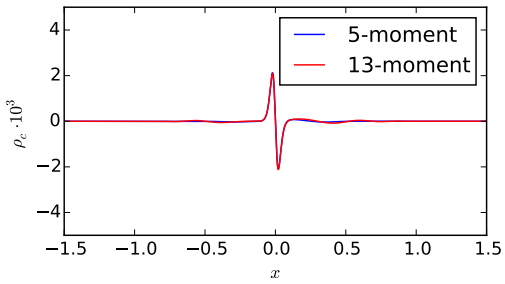


(c) Comparison of 13-moment results (blue) to theory (red).

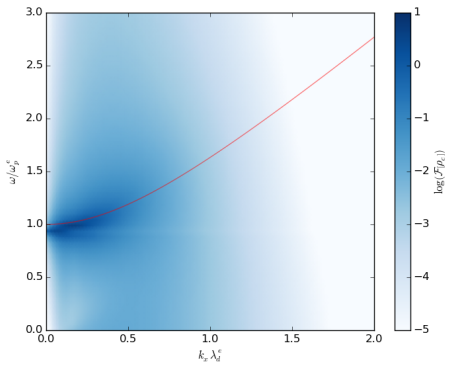
Figure 5.3.4: Comparison between 5-moment and 13-moment plasma models for the Langmuir oscillation problem with $(\nu_p \tau) = 10^3$. The 5-moment and 13-moment models show good agreement with the theoretical results, with less background noise in comparison to the increased collisionality case in Fig. 5.3.3.

Figure 5.3.5 compares the models at an even lower collisionality of $(\nu_p \tau) = 10^2$, which is the border of the weakly-coupled plasma regime where $\omega_p \geq \nu_p$. Here it is difficult to judge whether the 5-moment model is agreeing with the theoretical results, however the 13-moment model has noticeably deviated from the expected solution. This is due to the hyperbolic nature of the closure, where the electrostatic waves are decoupled from

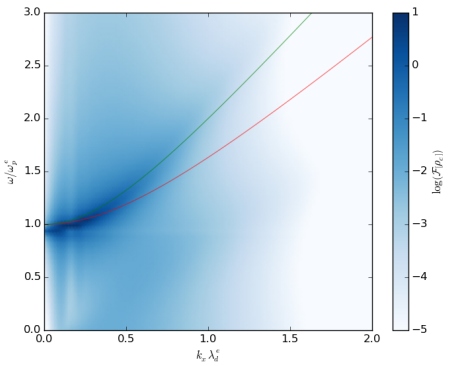
the electron's isotropic pressure, and are now dependent only on the anisotropic pressure P_{xx}. This agrees with kinetic theory[28], which is given by Eq. 5.3.4 when assuming $\gamma = 3$. The collisionless dispersion relation shows better agreement with the 13-moment model in Fig. 5.3.4, however a more detailed benchmark is required for further analysis.



(a) Comparison of 5-moment and 13-moment charge densities.



(b) Comparison of 5-moment results (blue) to theory (red).



(c) Comparison of 13-moment results (blue) to theory (red).

Figure 5.3.5: Comparison between 5-moment and 13-moment plasma models for the Langmuir oscillation problem with $(\nu_p \tau) = 10^2$. While it is difficult to state that the 5-moment results agree with the expected dispersion relation, it is interesting to note that the 13-moment model does show a rough agreement with the collisionless version of Eq. (5.3.4) where $\gamma = 3$, which is shown in green.

5.3.3 Multi-species Hartmann flow

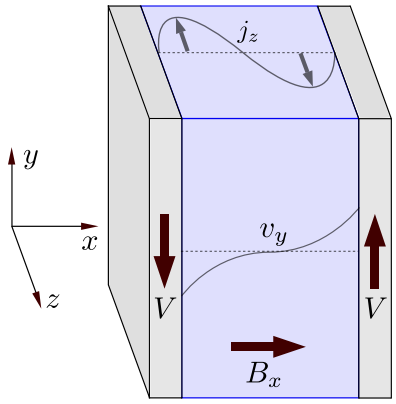


Figure 5.3.6: Hartmann flow diagram showing geometry of shear flow and imposed magnetic field as well as the resultant current density.

The Hartmann flow problem is a classic benchmark[54] for magnetoviscous, resistive plasmas. The problem setting, shown in Fig. 5.3.6, is similar to a Couette flow, where two infinite conducting plates are used to drive shear flow along the y-direction. The shear, along with an imposed magnetic field along x, drives a current in the z-direction which is balanced by resistive forces. The flow generated along z interacts with B_x to drive a force along y, suppressing, or flattening, the viscous boundary layer.

For the MHD model, the shear velocity profile control is directly dependent on the Hartmann number[54]. For the presented normalization, the profile is largely controlled by $\omega_c\tau$, but must also take into consideration a complex combination of the masses, charges, and thermalization rates. For this derivation, the plasma is assumed to be incompressible and the walls are no-slip and adiabatic. The multi-species Hartmann flow problem is derived from the 5-moment multi-species plasma model where the continuity equation,

$$\frac{\partial}{\partial t} n_\alpha = -\frac{\partial}{\partial x_i} \left(n_\alpha u_i^\alpha \right), \tag{5.3.5}$$

and momentum equation,

$$\begin{aligned}\frac{\partial}{\partial t}\left(A_\alpha n_\alpha u_i^\alpha\right) = &-\frac{\partial}{\partial x_j}\left(A_\alpha n_\alpha u_i^\alpha u_j^\alpha + P_\alpha \delta_{ij} + \Pi_{ij}^\alpha\right) \\ &+ Z_\alpha(\omega_p\tau)^2 n_\alpha E_i + Z_\alpha(\omega_c\tau) n_\alpha \epsilon_{ijk} u_j^\alpha B_k \\ &- \sum_\beta (\nu_p\tau)\nu_{\alpha\beta} A_\alpha n_\alpha \left(u_i^\alpha - u_i^\beta\right)\end{aligned} \tag{5.3.6}$$

use the closure described in Sec. 2.3

$$\Pi_{ij}^\alpha = -\frac{P_\alpha}{(\nu_p\tau)\nu_\alpha}\left(\frac{\partial}{\partial x_i}u_j^\alpha + \frac{\partial}{\partial x_j}u_i^\alpha - \frac{2}{3}\delta_{ij}\frac{\partial}{\partial x_k}u_k^\alpha\right). \tag{5.3.7}$$

Note that the accuracy of this model is limited to small Hall parameters where $\omega_c \ll \nu_p$. The goal of the derivation is to develop a steady state equilibrium. This simplifies the continuity equation

$$\frac{\partial}{\partial x_i}\left(n_\alpha u_i^\alpha\right) = 0 \tag{5.3.8}$$

and momentum equation

$$\begin{aligned}A_\alpha n_\alpha u_j^\alpha \frac{\partial}{\partial x_j}u_i^\alpha = &-\frac{\partial}{\partial x_i}P_\alpha + Z_\alpha(\omega_p\tau)^2 n_\alpha E_i + Z_\alpha(\omega_c\tau) n_\alpha \epsilon_{ijk} u_j^\alpha B_k \\ &+ \frac{\partial}{\partial x_j}\left(\frac{P_\alpha}{(\nu_p\tau)\nu_\alpha}\left(\frac{\partial}{\partial x_i}u_j^\alpha + \frac{\partial}{\partial x_j}u_i^\alpha\right)\right) \\ &- \frac{\partial}{\partial x_j}\left(\frac{2P_\alpha\delta_{ij}}{3(\nu_p\tau)\nu_\alpha}\frac{\partial}{\partial x_k}u_k^\alpha\right) \\ &- \sum_\beta (\nu_p\tau)\nu_{\alpha\beta} A_\alpha n_\alpha \left(u_i^\alpha - u_i^\beta\right).\end{aligned} \tag{5.3.9}$$

Since the problem is designed for a slab geometry, there are no gradients in the y or z directions. Applying the slab geometry to Eq. (5.3.8),

$$\frac{\partial}{\partial x}\left(n_\alpha u_x^\alpha\right) = 0, \tag{5.3.10}$$

with the boundary condition of no flow normal to the walls, results in the condition

$$n_\alpha u_x^\alpha = 0. \tag{5.3.11}$$

Assuming that the plasma has a density implies $u_x^\alpha = 0$. Steady state Faraday's Law in a slab geometry,

$$\frac{\partial}{\partial x} E_y = \frac{\partial}{\partial x} E_z = 0, \tag{5.3.12}$$

is applied in conjunction with the conducting wall boundary condition, $\hat{n} \times \vec{E} = 0$, to give

$$E_y = E_z = 0. \tag{5.3.13}$$

Applying these conditions to the momentum equation leads to

$$\frac{\partial}{\partial x}^2 u_y^\alpha = -\frac{1}{P_\alpha}(\frac{\partial}{\partial x} P_\alpha)(\frac{\partial}{\partial x} u_y^\alpha) - \lambda_\alpha u_z^\alpha + \sum_\beta \gamma_{\alpha\beta} \left(u_y^\alpha - u_y^\beta\right) \tag{5.3.14}$$

for the shear flow and

$$\frac{\partial}{\partial x}^2 u_z^\alpha = -\frac{1}{P_\alpha}(\frac{\partial}{\partial x} P_\alpha)(\frac{\partial}{\partial x} u_z^\alpha) + \lambda_\alpha u_y^\alpha + \sum_\beta \gamma_{\alpha\beta} \left(u_z^\alpha - u_z^\beta\right) \tag{5.3.15}$$

for the orthogonal flow with parameters

$$\lambda_\alpha = (\nu_p \tau)(\omega_c \tau) \frac{Z_\alpha \nu_\alpha n_\alpha B_x}{P_\alpha} \tag{5.3.16}$$

and

$$\gamma_{\alpha\beta} = (\nu_p \tau)^2 \frac{A_\alpha n_\alpha \nu_{\alpha\beta} \nu_\alpha}{P_\alpha}. \tag{5.3.17}$$

The pressure gradient

$$\frac{\partial}{\partial x}P_\alpha = Z_\alpha(\omega_p\tau)^2 n_\alpha E_x + Z_\alpha(\omega_c\tau)n_\alpha\left(u_y^\alpha B_z - u_z^\alpha B_y\right), \tag{5.3.18}$$

scales with the imposed electric field E_x, and the magnetic force from the shear and orthogonal flows. By removing the imposed electric field ($E_x = 0$), and assuming a very low Mach flow ($u_\alpha^2 \ll T$), the gradient of the pressure is minimized with respect to the background pressure, and can be ignored. This assumption on the pressure simplifies the equations to

$$\frac{\partial}{\partial x}^2 u_y^\alpha = -\lambda_\alpha u_z^\alpha + \sum_\beta \gamma_{\alpha\beta}\left(u_y^\alpha - u_y^\beta\right) \tag{5.3.19}$$

and

$$\frac{\partial}{\partial x}^2 u_z^\alpha = \lambda_\alpha u_y^\alpha + \sum_\beta \gamma_{\alpha\beta}\left(u_z^\alpha - u_z^\beta\right). \tag{5.3.20}$$

Note that the small pressure gradient implies that the parameters λ_α and $\gamma_{\alpha\beta}$ are roughly constant with respect to x.

To solve for u_y^α and u_z^α, Eqs. (5.3.19) and (5.3.20) are written in the matrix form $\vec{u}'' = \mathbf{A}\cdot\vec{u}$ where $u' = \frac{\partial}{\partial x}u$. For example, if the multi-species velocity vector is given by

$$\vec{u} = \begin{bmatrix} u_y^0 & u_z^0 & u_y^1 & u_z^1 & \cdots \end{bmatrix}^T$$

then the coupling matrix is given by

$$\mathbf{A} = \begin{bmatrix} \sum_{\beta\neq 0}\gamma_{0\beta} & \lambda_0 & -\gamma_{01} & 0 & \cdots \\ -\lambda_0 & \sum_{\beta\neq 0}\gamma_{0\beta} & 0 & -\gamma_{01} & \cdots \\ -\gamma_{10} & 0 & \sum_{\beta\neq 1}\gamma_{1\beta} & \lambda_1 & \cdots \\ 0 & -\gamma_{10} & -\lambda_1 & \sum_{\beta\neq 1}\gamma_{1\beta} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

To solve, the coupling matrix undergoes an eigendecomposition $\mathbf{A} = \mathbf{R}\cdot\mathbf{D}\cdot\mathbf{R}^{-1}$ such that $\vec{u}'' = \mathbf{R}\cdot\mathbf{D}\cdot\mathbf{R}^{-1}\cdot\vec{u}$. Defining an eigenspace velocity $\vec{s} = \mathbf{R}^{-1}\cdot\vec{u}$ converts the problem into an orthogonal equation set $\vec{s}'' = \mathbf{D}\cdot\vec{s}$ with diagonal eigenvalue matrix $\mathbf{D}$. This means that each component of $\vec{s}$ must obey the boundary value problem $s_i'' = D_i s_i$ which has a general solution

$$s_i(x) = C_i^0 e^{\sqrt{D_i}x} + C_i^1 e^{-\sqrt{D_i}x}. \tag{5.3.21}$$

Given a wall velocity V, the boundary conditions are $u_y^\alpha(x_{L/R}) = \mp V$ and $u_z^\alpha(x_{L/R}) = 0$ at boundary positions $x_{L/R}$. The solution coefficients C_i^0 and C_i^1 are then found by solving Eq. (5.3.21) using the boundary values $\vec{s}(x_{L/R}) = \mathbf{R}^{-1}\cdot\vec{u}(x_{L/R})$. The real space velocity solution is retrieved using $\vec{u} = \mathbf{R}\cdot\vec{s}$. In general $\{\vec{s}, \mathbf{R}, \mathbf{D}\} \in \mathbb{C}$ making analytical solutions impractically complicated, and for this study Wolfram *Mathematica*® was used to find the velocity profiles numerically.

The solution to the magnetic field is given by integrating steady state Ampere's laws

$$\frac{\partial}{\partial x}B_y = \frac{(\omega_p\tau)^2}{(\omega_c\tau)\left(\frac{c}{v_0}\right)^2}\sum_\alpha Z_\alpha n_\alpha u_z^\alpha, \tag{5.3.22}$$

and

$$\frac{\partial}{\partial x}B_z = -\frac{(\omega_p\tau)^2}{(\omega_c\tau)\left(\frac{c}{v_0}\right)^2}\sum_\alpha Z_\alpha n_\alpha u_y^\alpha, \tag{5.3.23}$$

with the condition that the plasma is held in a flux conserver

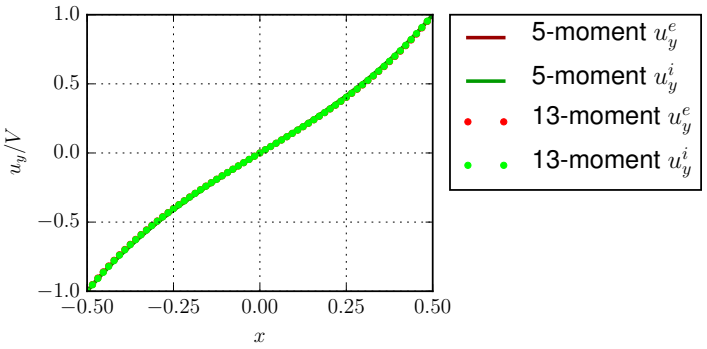
$$\int B_y\,dx = \int B_z\,dx = 0. \tag{5.3.24}$$

Since the 5-moment Hartmann flow problem is being used to verify the 13-moment model, a careful examination of how the individual model's parameter regimes overlap must be taken into account.

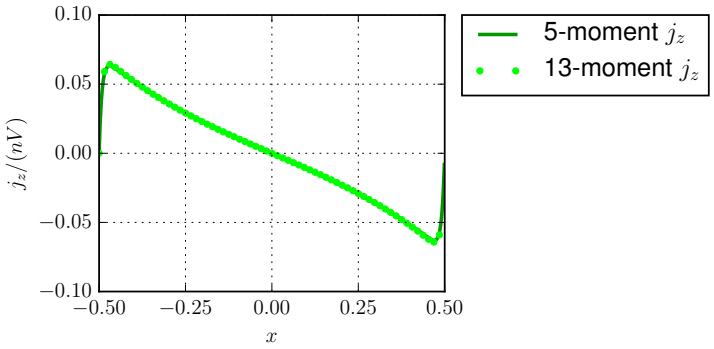
- The magneto-collisional constraint requires $\omega_c \ll \nu_p$ to avoid strongly magnetoviscous 'Braginskii effects' which are ignored by the 5-moment model.
- The collisionality constraint requires $\nu_p\tau \geq 10^2$ to keep 5-moment model valid, but also requires $\nu_p\tau \ll 10^5$ to prevent low mass particles from becoming inviscid and thus numerically stiff in the 13-moment description.
- Keeping the lightest mass species subluminal requires $c/v_0 \gg \max_\alpha\left(\sqrt{T_\alpha/A_\alpha}\right)$ which must also be minimized to reduced runtime.
- To keep the Alfvén velocities subluminal requires $\omega_p \gg \omega_c$, but larger mass ions must be magnetized $\omega_c\tau \geq 1$.
- To avoid large gains in temperature due to viscous heating from the shearing plates, the plate velocity is limited to $V \ll \min_\alpha\left(\sqrt{T_\alpha/A_\alpha}\right)$.

Three comparisons were generated for the Hartmann flow problem for which the diffusive stabilization operator is ignored in the 13-moment model. The first, shown in Fig. 5.3.7, represents a weakly magnetized system where the magnetic forces do not drive a strong current density in the z direction. Since the Hall parameter $\omega_c/\nu_p = 10^{-2}$ is small, the effect

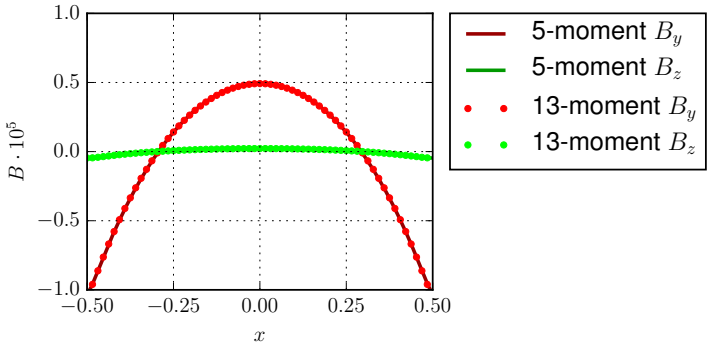
of magnetization on the collisional transport is minimal, which means that the 13-moment model can agree well with the analytical model derived from the 5-moment model. These 13-moment model results were generated in the **examples/dg/plasma/hartmann_flow/hartmann_flow_13moment.py** script.



(a) Shear velocity.



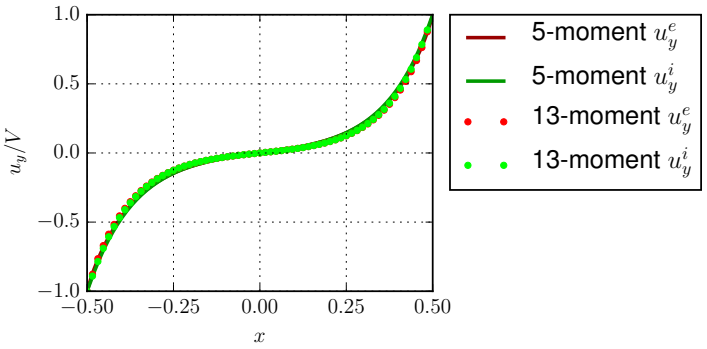
(b) Current density.



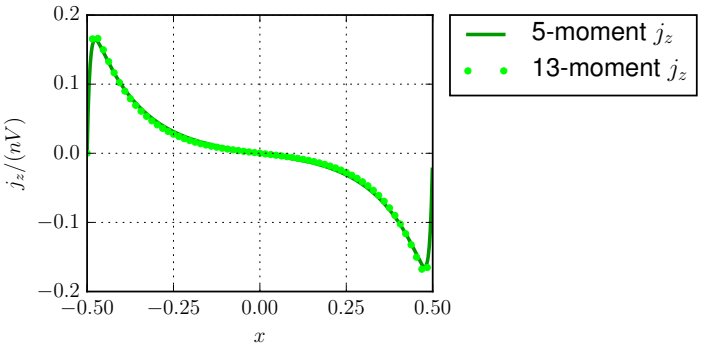
(c) Magnetic fields.

Figure 5.3.7: Two-species Hartmann flow problem with parameters $(\nu_p\tau) = 10^2$, $(\omega_c\tau) = 1$, $(c/v_0) = 10^2$, and $(\omega_p\tau) = 10^2$. Ion have mass $A_i = 1$ and charge $Z_i = 1$, while electrons have mass $A_e = 10^{-2}$ and charge $Z_e = -1$. Simulation is run to time $t = 10$. Results show good agreement between the 5-moment analytical solution and numerical solution from the 13-moment model.

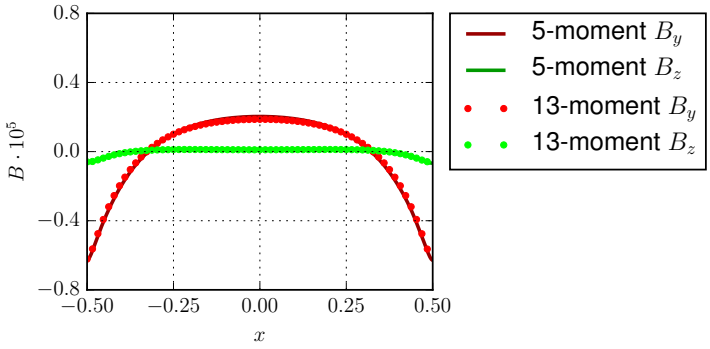
Figure 5.3.8 has an increased magnetization of $(\omega_c\tau) = 3$, which increases the Hall parameter to $\omega_c/\nu_p = 3 \cdot 10^{-2}$. While this is still a fairly weakly magnetized system, there are small deviations between the 13-moment solution and the analytical solution due to the interaction between the collision operators and the magnetic field (i.e. magnetoviscosity). This deviation is expected since the 13-moment model directly couples the magnetic field to the anisotropic pressure tensor (see Eq. (2.4.3)), while the analytical solution derived from the 5-moment model ignores magnetoviscous effects.



(a) Shear velocity.



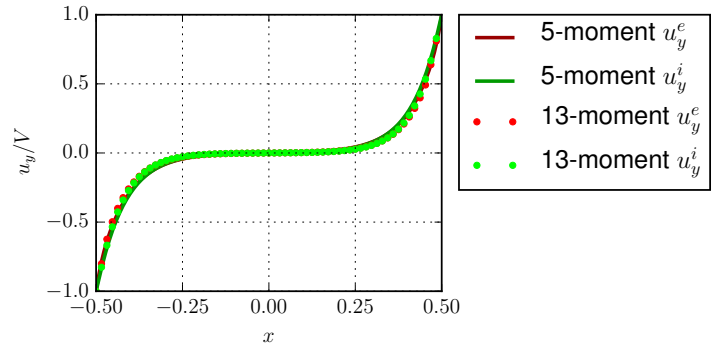
(b) Current density.



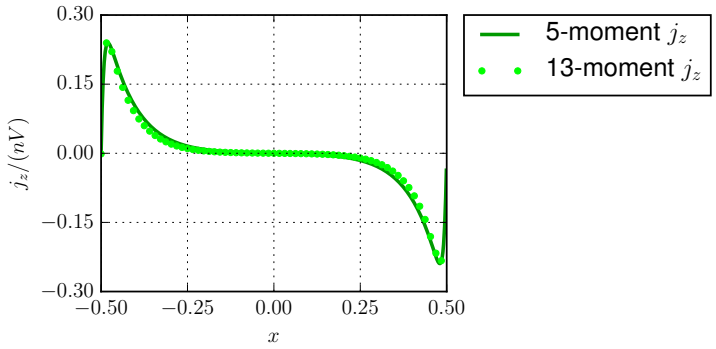
(c) Magnetic fields.

Figure 5.3.8: Two-species Hartmann flow problem with parameters $(\nu_p\tau) = 10^2$, $(\omega_c\tau) = 3$, $(c/v_0) = 10^2$, and $(\omega_p\tau) = 10^2$. Ion have mass $A_i = 1$ and charge $Z_i = 1$, while electrons have mass $A_e = 10^{-2}$ and charge $Z_e = -1$. Simulation is run to time $t = 10$. Results show good agreement between the 5-moment analytical solution and numerical solution from the 13-moment plasma model.

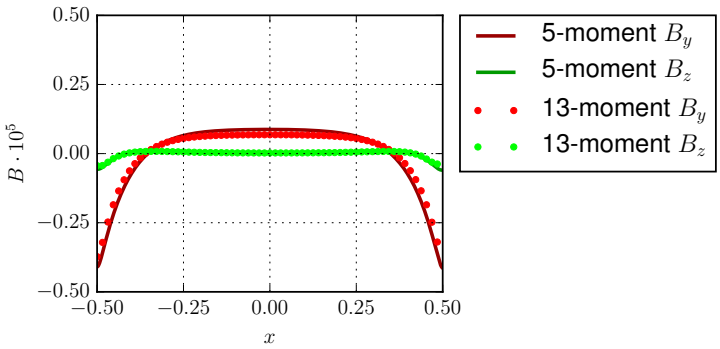
Figure 5.3.9 compares the 5-moment analytical solution to the 13-moment model for a higher level of magnetization of $(\omega_c\tau) = 5$ where the Hall parameter is $\omega_c/\nu_p = 5{\cdot}10^{-2}$. As expected, increasing the Hall parameter has increased the deviation between the 13-moment model and the analytical solution.



(a) Shear velocity.



(b) Current density.



(c) Magnetic fields.

Figure 5.3.9: Two-species Hartmann flow problem with parameters $(\nu_p\tau) = 10^2$, $(\omega_c\tau) = 5$, $(c/v_0) = 10^2$, and $(\omega_p\tau) = 10^2$. Ion have mass $A_i = 1$ and charge $Z_i = 1$, while electrons have mass $A_e = 10^{-2}$ and charge $Z_e = -1$. Simulation is run to time $t = 10$. While the results show decent agreement, the moderate level of magnetization is beginning to drive deviations between the 5-moment analytical solution and numerical solution from the 13-moment plasma model.

These results have validated the collision operators of the 13-moment model against the analytical solution derived from the 5-moment model for weakly magnetized plasmas. Additionally, the interspecies collision operators and electromagnetic operators in the 13-moment model have been shown to be consistent for the parameter regime chosen for this problem. For moderately magnetized plasmas, small deviations have arisen due to magnetization affects captured by the 13-moment model that are ignored by the 5-moment model. For future analysis of strongly magnetized collisional transport in the 13-moment model, it would be advantageous to extend the multi-species Hartmann flow analytical solution to include the magnetoviscous Braginskii closure[11] terms.

5.3.4 GEM reconnection challenge

The GEM reconnection challenge is designed to capture fast magnetic reconnection time scales. Generally this problem is applied to a collisionless parameter regime, requiring a kinetic representation. The problem has also been studied using moment models. While moment models do not capture much of the intricate plasma behavior during fast magnetic reconnection, these models can capture some of the bulk plasma response, such as the reconnection flux rate. A detailed description and analysis of the GEM challenge can be found in Birn et al.[1] as well as Ricci et al.[55]. Additional studies on the GEM challenge can be found in Srinivasan[52], Wang et al.[22], Hakim[56], and Reddell[57]. The concept behind the GEM challenge is to use a slab current sheet, impose a perturbation to the magnetic field, and analyze the resulting reconnection event. The growth rate of the reconnected flux has been shown to have good agreement between Hall-MHD[1;52], multi-species 5-moment plasma models[22;52], 10-moment plasma models[22;56], particle-in-cell models[1;22], and continuum kinetic models[57].

The derivation of the GEM reconnection problem begins with the steady state pressure balance relationship between the fluid's pressure gradient and the magnetic force on the

plasma

$$\frac{\partial}{\partial y}\left(n_\alpha T_\alpha\right) = (\omega_c\tau) Z_\alpha n_\alpha u_z^\alpha B_x. \tag{5.3.25}$$

The GEM challenge ignores collisions between species ($R_i^\alpha = 0$). When applied to moment models, the GEM challenge assumes each species in the plasma to exist in local thermal equilibrium, meaning zero viscosity, which is not a physically accurate assumption. For kinetic representations, the GEM challenge derivation can have more liberty in how it treats effects associated with non-thermal equilibrium plasmas (see Reddell[57]). The relationship between the magnetic field and the current density is given by Ampere's law

$$j_i = \left(\frac{\delta_p}{L}\right)^2 (\omega_c\tau)\epsilon_{ijk}\frac{\partial}{\partial x_j}B_k. \tag{5.3.26}$$

which, for this geometry, becomes

$$\sum_\alpha Z_\alpha n_\alpha u_z^\alpha = -\left(\frac{\delta_p}{L}\right)^2 (\omega_c\tau)\partial_y B_x. \tag{5.3.27}$$

The pressure balance in Eq. (5.3.25) can be rewritten in terms of the magnetic field to give a constraint on all species

$$(\omega_c\tau)B_x = \frac{\partial_y\left(n_\alpha T_\alpha\right)}{Z_\alpha n_\alpha u_z^\alpha} = \frac{\partial_y\left(n_\beta T_\beta\right)}{Z_\beta n_\beta u_z^\beta}. \tag{5.3.28}$$

The density profile of the GEM challenge is given by

$$n_\alpha = n_0 \mathrm{sech}\left(\frac{y}{\lambda}\right)^2 + n_1, \tag{5.3.29}$$

with peak density $n_0 + n_1$ and background density n_1, in order to enforce quasineutrality in $|Z_\alpha| = 1$ plasmas. The temperature profile is assumed constant. This simplifies in species

constraint in Eq. (5.3.28) into

$$\frac{T_\alpha}{Z_\alpha u_z^\alpha} = \frac{T_\beta}{Z_\beta u_z^\beta}. \tag{5.3.30}$$

Clasically, the GEM challenge uses a magnetic field of the form

$$B_x = B_0 \tanh\Big(\frac{y}{\lambda}\Big). \tag{5.3.31}$$

Combining the solution for the magnetic field and number density with the pressure balance requirement in Eq. (5.3.25) gives the solution for the flow velocity

$$\begin{aligned} u_z^\alpha &= \frac{T_\alpha}{Z_\alpha(\omega_c\tau)}\frac{1}{B_z}\frac{\partial_y n_\alpha}{n_\alpha} \\ &= \frac{T_\alpha}{Z_\alpha(\omega_c\tau)B_0}\frac{1}{\tanh\left(\frac{y}{\lambda}\right)}\frac{n_0\partial_y\left(\mathrm{sech}\left(\frac{y}{\lambda}\right)^2\right)}{\left(n_0\mathrm{sech}\left(\frac{y}{\lambda}\right)^2 + n_1\right)} \\ &= -\frac{2T_\alpha}{Z_\alpha(\omega_c\tau)B_0\lambda}\frac{1}{\tanh\left(\frac{y}{\lambda}\right)}\frac{n_0\left(\mathrm{sech}\left(\frac{y}{\lambda}\right)^2\tanh\left(\frac{y}{\lambda}\right)\right)}{n_0\mathrm{sech}\left(\frac{y}{\lambda}\right)^2 + n_1}. \end{aligned} \tag{5.3.32}$$

Assuming $n_0 \neq 0$, this can be rearranged into

$$u_z^\alpha = -\frac{T_\alpha}{Z_\alpha}\frac{2}{(\omega_c\tau)B_0\lambda}\frac{1}{\left(1+\frac{n_1}{n_0}\cosh\left(\frac{y}{\lambda}\right)^2\right)} \tag{5.3.33}$$

which has an extrema of

$$u_z^\alpha(y=0) = -\frac{T_\alpha}{Z_\alpha}\frac{2}{(\omega_c\tau)B_0\lambda}\left(1+\frac{n_1}{n_0}\right)^{-1}. \tag{5.3.34}$$

Given these solution profiles, the current density is given by

$$\begin{aligned}\sum_\alpha Z_\alpha n_\alpha u_z^\alpha &= -\frac{2}{(\omega_c\tau)B_0\lambda}\frac{1}{\left(1+\frac{n_1}{n_0}\cosh\left(\frac{y}{\lambda}\right)^2\right)}\left(n_0\text{sech}\left(\frac{y}{\lambda}\right)^2+n_1\right)\sum_\alpha T_\alpha \\ &= -\frac{2n_0\sum_\alpha T_\alpha}{(\omega_c\tau)B_0\lambda}\text{sech}\left(\frac{y}{\lambda}\right)^2\end{aligned} \tag{5.3.35}$$

while the right hand side of Ampere's law becomes

$$\begin{aligned}-\left(\frac{\delta_p}{L}\right)^2(\omega_c\tau)\partial_y B_x &= -\left(\frac{\delta_p}{L}\right)^2(\omega_c\tau)B_0\partial_y\tanh\left(\frac{y}{\lambda}\right) \\ &= -\frac{\left(\frac{\delta_p}{L}\right)^2(\omega_c\tau)B_0}{\lambda}\text{sech}\left(\frac{y}{\lambda}\right)^2.\end{aligned} \tag{5.3.36}$$

Combining the two sides of Ampere's law given in Eq. (5.3.35) with Eq. (5.3.36) results in a condition on the parameter space

$$(\omega_c\tau)^2\left(\frac{\delta_p}{L}\right)^2 = \frac{2n_0}{B_0^2}\sum_\alpha T_\alpha. \tag{5.3.37}$$

The GEM challenge applied to a ion-electron plasma reduces the parameter space for this problem by setting the number densities $n_0 = 1$ and $n_1 = 1/5$, and charges $Z_e = -1$ and $Z_i = 1$. The number density for each species is given by

$$n_e = n_i = \text{sech}\left(\frac{y}{\lambda}\right)^2 + \frac{1}{5}. \tag{5.3.38}$$

The drift velocities are in opposite directions,

$$\frac{u_y^i}{T_i} = -\frac{u_y^e}{T_e} = -\frac{2}{(\omega_c\tau)B_0\lambda}\frac{1}{\left(1+\frac{1}{5}\cosh\left(\frac{y}{\lambda}\right)^2\right)}, \tag{5.3.39}$$

and the condition on the parameter space simplifies to

$$\frac{\delta_p}{L} = \frac{\sqrt{2(T_i + T_e)}}{(\omega_c \tau) B_0}, \tag{5.3.40}$$

which can also be written in terms of the ion-electron temperature ratio as

$$T_i = \left(\frac{\delta_p}{L}\right)^2 (\omega_c \tau)^2 \frac{B_0^2}{2\left(1 + \frac{T_e}{T_i}\right)}. \tag{5.3.41}$$

The GEM challenge is designed around a domain of size $x \in \{-4\pi, 4\pi\}$ by $y \in \{-2\pi, 2\pi\}$ with a skin depth of $\delta_p/L = 1$ and a sheet width of $\lambda = 1/2$. For convenience in comparing with literature, the magnetization parameter is set to $\omega_c \tau = 1$, which puts the time in terms of ion cyclotron periods. The temperature ratio is set by the species mass ratio $T_i/T_e = \sqrt{A_i/A_e}$. The combination of the temperature ratio, density, and parameter space can be used in Eq. (5.3.41) to retrieve the ion temperature

$$T_i = \frac{1}{2\left(1 + \sqrt{\frac{A_e}{A_i}}\right)} \tag{5.3.42}$$

and electron temperature

$$T_e = \frac{1}{2\left(1 + \sqrt{\frac{A_i}{A_e}}\right)}. \tag{5.3.43}$$

Velocities are set at

$$u_z^i = -\frac{1}{2\left(1 + \sqrt{\frac{A_e}{A_i}}\right)} \frac{1}{1 + \frac{1}{5}\cosh(y)^2}, \tag{5.3.44}$$

and

$$u_z^e = \frac{1}{2\left(1+\sqrt{\frac{A_i}{A_e}}\right)} \frac{1}{1+\frac{1}{5}\cosh\left(y\right)^2}. \tag{5.3.45}$$

While the equilibrium magnetic field is $\vec{B} = \tanh\left(y\right)\hat{x}$, the actual field is perturbed with a vector potential of the form

$$A_z^1 = \frac{1}{10}\cos\left(k_x x\right)\cos\left(k_y y\right) \tag{5.3.46}$$

such that

$$\vec{B} = \tanh\left(\frac{y}{\lambda}\right)\hat{x} + \nabla \times \vec{A^1} = \begin{bmatrix} \tanh\left(\frac{y}{\lambda}\right) - \frac{1}{10}k_y \cos\left(k_x x\right)\sin\left(k_y y\right) \\ \frac{1}{10}k_x \sin\left(k_x x\right)\cos\left(k_y y\right) \\ 0 \end{bmatrix}. \tag{5.3.47}$$

The GEM challenge specifies the perturbation with $k_x = k_y = 1/4$. The plasma frequency is not explicitly set in the GEM challenge; however, to keep the Alfven velocity v_A and thermal velocities well below the speed of light, the reference plasma frequency is set to $\omega_p \tau = c/v_0 = c/v_A = 20$. For this study, the mass ratio between the ions and electrons is set to $A_i = A_e = 1$ as to make this a positron-electron plasma. This mass ratio is chosen since it will decrease simulation runtime compared to the traditional GEM challenge choice of $A_i = 25A_e = 1$, and has little impact on the reconnected flux rate, as was shown by Ricci et al.[55]. Boundary conditions for this problem are periodic across the left and right walls, while two ideal conducting plates confine the plasma on the top and bottom wall.

These initial conditions are shown in Fig. 5.3.10 in terms of the plasma density $n = (\rho_i + \rho_e)/(A_i + A_e)$, plasma pressure $P = P_i + P_e$, and current density $j_z = Z_e p_z^e/A_e + Z_i p_z^i/A_i$. The simulation was carried out for both the the 5-moment model and the 13-moment model. The 5-moment model assumed no viscosity, thermal conductivity, or resistivity.

The 13-moment model inherently assumes there is some viscosity and thermal conductivity in the fluid, which is limited by the BGK collision operator at a collision frequency of $\nu_p\tau = 10^2$. This level of collisionality is used to help stabilize the 13-moment closure around the strong viscous forces at the beginning of the simulation due to the shear (represented by $\frac{\partial}{\partial y}u_z^\alpha$) which will act to make the current sheet larger. After this initial viscous effect, the solutions of the 5-moment and 13-moment models are comparable. The models are run on a mesh of 4096 equally sized triangular elements with a cubic polynomial basis. The explicit time integration is done using the TVDRK2 method. The simulation scripts are found in **examples/dg/plasma/gem_challenge/gem_5moment.py** for the 5-moment model and **examples/dg/plasma/gem_challenge/gem_13moment.py** for the 13-moment model.

(a) Plasma number density.

(b) Current density.

(c) Plasma pressure.

Figure 5.3.10: Initial conditions for the plasma density, pressure, and current density. Current sheet is uniform along x, with periodic boundary conditions on the left and right wall and ideal conducting walls on the top and bottom.

The evolution of the 5-moment model and 13-moment model were found to be consistent for this parameter space. Unfortunately, the simulation itself becomes unstable at a time of $t = 22$, when the pressure profiles undergo a large amount of compression. The source of this instability is currently under study. Since the instability affects the 5-moment model and the 13-moment model in the same fashion, in both the instability's onset time and spatial structure, the instability is likely due to either an error in the initial conditions (e.g. stronger than necessary magnetic field) or with the discontinuous Galerkin method not properly balancing the strong pressure gradient with the strong Lorentz force.

Comparisons between the 5-moment and 13-moment models are given in Fig. 5.3.11, 5.3.12, and 5.3.13 for the density, current density, and pressure respectively. The results show that the response of the 5-moment model and 13-moment model to the reconnection event are almost exactly the same, with small variations due to the initial viscous response of the 13-moment model. The rate of reconnection, shown in Fig. 5.3.14, shows that up to the point of the instability, the reconnected flux, calculated as

$$\phi = \int_{-4\pi}^{4\pi} |B_y(x, y = 0)| \, dx, \tag{5.3.48}$$

aligns well with the results found in literature. The results show that the 13-moment model has potential as a multi-dimensional base for modeling magnetized plasma transport.

(a) 5-moment model.

(b) 13-moment model.

Figure 5.3.11: Comparison of number densities between 5-moment model and 13-moment model at time $t = 20$. Both 5-moment and 13-moment models give a consistent peak in the density at this time, with a thin sheet existing between the peaks.

(a) 5-moment model.

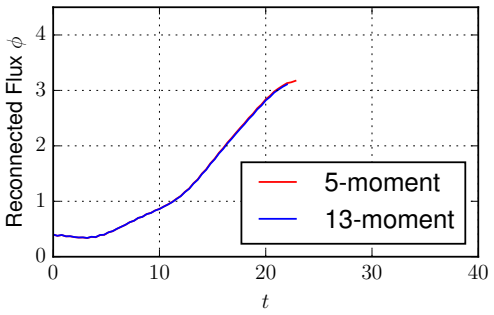
(b) 13-moment model.

Figure 5.3.12: Comparison of current densities between 5-moment model and 13-moment model at time $t = 20$. Both 5-moment and 13-moment models give consistent results at this point in time.

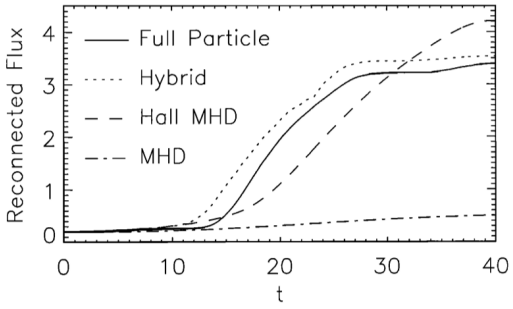
(a) 5-moment model.

(b) 13-moment model.

Figure 5.3.13: Comparison of plasma pressure between 5-moment model and 13-moment model at time $t = 20$. Both 5-moment and 13-moment models give consistent results at this point in time.



(a) Simulated reconnected flux.



(b) Literature reconnected flux.

Figure 5.3.14: Comparison of reconnected flux ϕ between simulated 5-moment and 13-moment models to literature results from Birn et al.[1]. Good agreement is shown up to the point of instability.

5.4 Maxwell's equations

5.4.1 Static cleaning response

The static cleaning response test is designed to monitor how an initial divergence error is cleaned out of a solution over time. The goal of the test is to show the benefits of the local divergence error cleaning operator of PCMaxwell. In this test, a one dimensional domain of $x \in [-1, 1]$ is initialized with

$$\vec{E} = 0 \tag{5.4.1}$$

$$B_x = \sum_{i=1}^{i<10} \cos\left(2\pi(50i + \bar{W}_i)x\right) \tag{5.4.2}$$

$$B_y = B_z = 0 \tag{5.4.3}$$

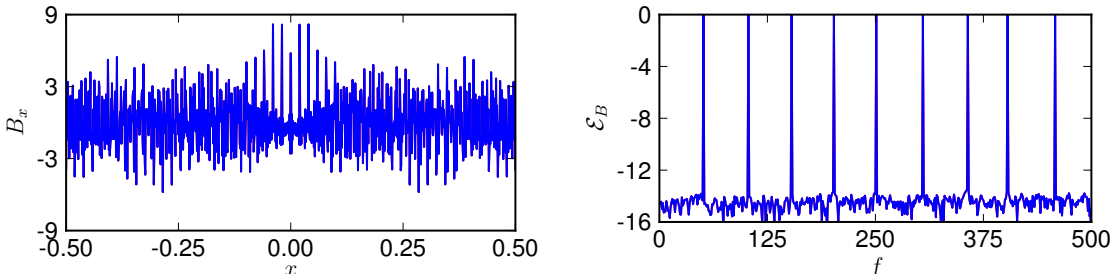
where $0 < \bar{W}_i < 10$ is a noise source used to prevent repeated modal structures in the domain. The cleaning coefficients are chosen to keep a common time step between both models such that $\gamma = \chi = 1$ for PHMaxwell implies $\chi_E = \chi_B = \frac{c\Delta x}{2}$ for PCMaxwell. The

divergence errors were cleaned from a domain with mesh resolution $\Delta x = 10^{-3}$ using the high-order finite volume method over a time of 0.12. The analytical solution for this system is a $B_x = 0$ which is enforced by the boundary conditions.

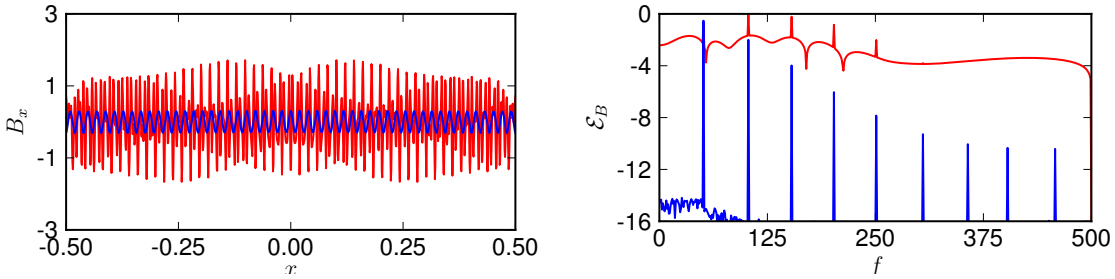
The static cleaning response of PCMaxwell and PHMaxwell is shown in Fig.5.4.1 with magnetic field error

$$\mathcal{E}_B = \ln |\frac{1}{500}\mathcal{F}(B_x)| \tag{5.4.4}$$

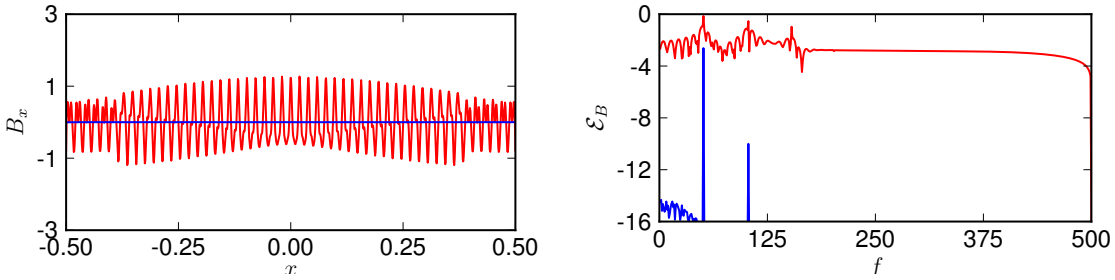
where $\mathcal{F}$ is a Fourier transform. The magnetic field error has arbitrary units and only expresses a comparison of B_x error between the models. Note that PCMaxwell is better at cleaning short wavelength noise from the solution, however the longest wavelength mode (20 elements wide) is nearly ignored by the parabolic operator for the time scale of the simulation. Since PHMaxwell advects the divergence error throughout the domain using a numerical flux, it introduces wide-band noise in k space. These results show that the local divergence error removal of PCMaxwell is fast at removing short-wavelength errors from a system. Long-wavelength errors are removed over a longer time scale, however, in practice it is unlikely that a system would be initialized with low-frequency divergence errors.



(a) $t = 0.000$



(b) $t = 0.024$



(c) $t = 0.120$

Figure 5.4.1: Cleaning comparison and error frequency spectrum for an initially perturbed system for PCMaxwell (blue) and PH-Maxwell (red). The time step Δt is the same for both methods. Time increments are (a) $t = 0$, (b) $t = 0.024$, (c) $t = 0.12$. Results show that PCMaxwell cleans mesh scale divergence errors quickly from the domain, while large wavelength divergence errors are slow to dissipate.

5.4.2 Dynamic cleaning response

The dynamic response problem tests the ability of PHMaxwell and PCMaxwell to remove divergence errors in the presence of the plasma dynamics similar to those found in Sec. 5.3.1. The plasma is represented by ideal 5-moment model ($(\nu_p\tau) \to \infty$) to generate sharp shock profiles which leads to discontinuities in the charge density. The boundary conditions are simple zero-normal-gradient for all fluid and field components. The boundary conditions here are important as they introduce electric field divergence errors when the Langmuir waves transit the boundary.

The shock tube is setup in a domain $x \in \left[-\frac{1}{2}, \frac{1}{2}\right]$ with a wall located at $x = 0$. The initial conditions for the left (l) and right (r) states are given by,

$$n_l = 8n_r = 1, \tag{5.4.5}$$

$$T_e = T_i = 1, \tag{5.4.6}$$

$$\vec{E} = 0, \tag{5.4.7}$$

$$\vec{B}_l = \begin{bmatrix} \frac{3}{4} & 1 & 0 \end{bmatrix}^T, \tag{5.4.8}$$

$$\vec{B}_r = \begin{bmatrix} \frac{3}{4} & -1 & 0 \end{bmatrix}^T, \tag{5.4.9}$$

with ion mass $A_i = 25$, ion charge $Z_i = 1$, electron mass $A_e = 1$, electron charge $Z_i = -1$, plasma densities $n = n_e = n_i$, and no initial electric field $\vec{E} = 0$. The plasma frequency is set at $(\omega_p\tau) = 20$ to allow for long wavelength charge separation, while the cyclotron frequency is set at $(\omega_c\tau) = 4$ which allows for moderate magnetization effects. The speed of light is set at $c/v_0 = 50$ to reduce runtime. The simulation is evolved to time $t = 0.6$ to allow for the ion species to propagate across the domain. The cleaning coefficients remain the same as in Sec. 5.4.1, and the simulation is run using the finite volume method. Figure 5.4.2 compares the cleaning of PCMaxwell to PHMaxwell at three times. The electric field error is given

by

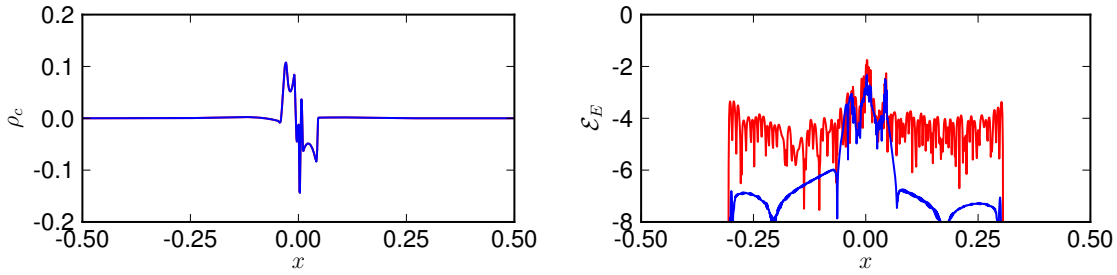
$$\mathcal{E}_E = |\partial_x E_x - \rho_c|. \tag{5.4.10}$$

Initially, in Fig. 5.4.2a, as the electrons first shock outward, the discontinuous charge density interface at the center of the domain is the primary generator of divergence errors. PC-Maxwell keeps the errors local to the shock, while PHMaxwell spreads them out in error waves traveling at the speed of light.

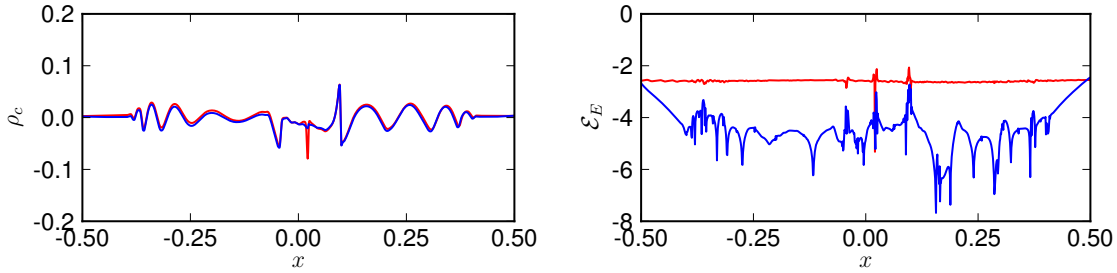
In Fig. 5.4.2b, the first electron waves have exited the boundary inducing a violation of Gauss's law. The error waves of PHMaxwell have imposed a near constant divergence error across the domain which has pushed the electrons into a slightly different configuration than PCMaxwell.

In Fig. 5.4.2c the divergence errors from the boundary conditions have greatly affected the electric field for the PHMaxwell model. PCMaxwell is also showing a large error source at the boundary, but the locality of the cleaning operator is minimizing the divergence error's effect on the internal solution.

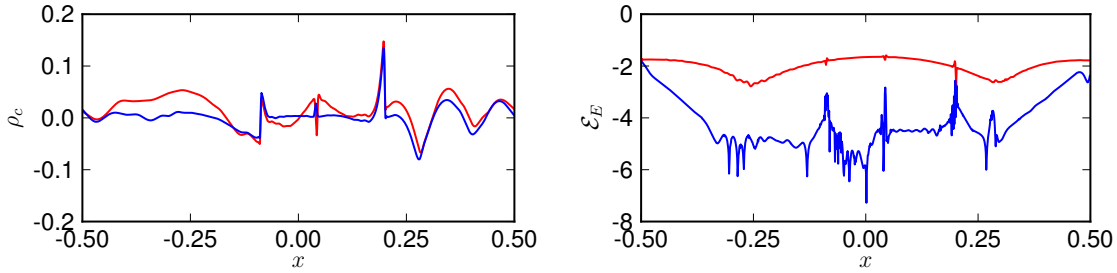
The example shows that even though the simulation is designed to induce large divergence errors into the domain, PCMaxwell is able to stably model accurate plasma dynamics without impacting the runtime.



(a) $t = 0.03$



(b) $t = 0.30$



(c) $t = 0.60$

Figure 5.4.2: Cleaning comparison for a dynamically perturbed system with parabolic cleaning (blue) and hyperbolic cleaning (red) of Maxwell's equation applied to the electromagnetic shock tube of Sec. 5.3.1. Time increments are (a) $t = 0.03$ after initial electron shock, (b) $t = 0.3$ after divergence errors are generated at the boundary, and (c) $t = 0.6$ after boundary divergence errors have propagated into interior of domain. PCMaxwell is shown to clean the solution the same or better than PHMaxwell for the same time step size.

5.5 Summary of results

The results have shown that the 13-moment plasma model is consistent with the 5-moment plasma model. Both the neutral fluid and charged fluid results showed good agreement for strongly to moderately collisional systems. The charged fluid results showed good agreement with the 5-moment model for weakly to moderately magnetized plasmas, however, additional benchmarks are required to analyze the 13-moment model in strongly magnetized, moderately collisional systems.

PCMaxwell was shown to be a adept at locally cleaning short wavelength divergence errors from Maxwell's equations compared to the PHMaxwell model. This effect is beneficial in simulations where divergence errors are primarily generated from discontinuous charge separation effects or boundary conditions. For systems initialized with long wavelength divergence errors, such as systems containing electrode or magnetic coil feedback boundary conditions, PCMaxwell reacts too slowly to be adequate. In such cases, a combination of the locality of a parabolic cleaning operator for short wavelength errors and the speed of the hyperbolic cleaning for long wavelength errors would be ideal.

Chapter 6

CONCLUSION

The research described in this dissertation provides the framework and justification for developing computationally efficient modeling techniques for collisional transport processes in multi-species plasmas. Contributions of this work provide a more robust foundation supporting further research into multi-species, non-equilibrium plasma dynamics. The following sections summarizes the current state of the plasma model development as well as the tools used to study the plasma dynamics including the numerical methods implemented, and the supporting computational framework.

6.1 Plasma models

The plasma models developed and implemented in this research have been designed to capture collisional transport in plasmas. Initially a non-equilibrium, multi-species 5-moment plasma model was implemented, which included weakly-magnetized collisional transport effects both within and between species. The 5-moment model is an established model in the plasma physics community, with well-known validity restrictions based on collisional mean-free-path scales.

A 13-moment model was developed around the 5-moment model and used its additional moments to describe physical behavior relating to collisional transport that was ignored by the 5-moment model. The 13-moment closure problem, inherent to all moment models, was resolved using a Pearson type-IV distribution. A diffusive stabilization operator was added to the model to stabilize the closure scheme in the presence of strongly anisotropic flows. A BGK

collision operator was used to approximate scattering collisional effects within a species, and an interspecies collision operator was developed to incorporate resistive drag effects into the plasma model. The interspecies operator enforces the conservation of total mass, momentum, energy (both isotropic and anisotropic), and energy flux in a multi-species plasma. Finally, the 13-moment model includes an electromagnetic force operator that directly couples the higher moments, relating to the heat flux vector and anisotropic pressure tensor, to the magnetic field, similar to the Braginskii equations.

The multi-species model was developed to incorporate Maxwell's equations to describe the electric and magnetic fields. Maxwell's equations include to constraints relating to divergence conditions on the electric and magnetic field as described by Gauss's laws. Two cleaning methods were implemented to reduce these errors in the solution. A hyperbolic method was implemented which propagates the divergence errors through the plasma until the errors exit the domain through the boundary conditions. Since this method tends to drive global divergence errors from a local source, an alternative parabolic cleaning method was developed. The parabolic cleaning method was shown to have better cleaning behavior than the hyperbolic scheme for short wavelength error sources generated from boundary conditions and charge density shock structures.

The multi-species 13-moment plasma model presented in this dissertation has many advantages over its 5-moment counterpart. The 13-moment model directly captures the effects of magnetization and collisional interactions acting on the full pressure tensor and heat flux vector. The addition of these higher moments to describe anisotropic plasma behavior represents a step towards a higher-fidelity representation of physics. For neutral species modeling, the 13-moment model has been shown to be consistent with both 5-moment and kinetic models. The neutral model was also shown to be stable and accurate in multiple dimensions when modeling viscous flows. The full 13-moment plasma model has been shown to be consistent with the presented 5-moment plasma model for moderate to high collisionalities in the presence of a weak to moderate magnetic fields.

The current multi-species 13-moment plasma model could be further enhanced by adding two modules. The first module would replace the current closure scheme, including its diffusive stabilization operator, with a closure scheme more consistent with magnetized plasma transport. Doing so would increase the stability and accuracy of the 13-moment model in weakly-collisional and/or highly-magnetized systems. Modifying higher-moment model closures is difficult, especially when basing the closure on a Pearson type-IV distribution; however, existing regularized Grad methods may make an ideal base for a closure consistent with plasma transport dynamics. Additionally, to make the 13-moment model more applicable to low-temperature, partially-ionized plasmas, the model could be extended to include reactive collision operators to capture ionization, recombination, and charge-exchange effects.

6.2 Discontinuous numerical methods

Explicit discontinuous numerical methods are highly parallelizable methods for solving the larges equation sets associated with multi-species plasma physics. To simulate the plasma models used in this research, an unstructured, high-order finite volume method and discontinuous Galerkin method were developed and implemented in conjunction with explicit Runge-Kutta time integration schemes. The methods were designed to solve large sets of nonlinear equation sets in a computationally efficient manner.

The high-order finite volume method was developed and implemented as an initial spatial solver for testing the general-geometry, unstructured mesh framework. The method was implemented in one, two, and three dimensions using slab line, slab triangle, slab quadrilateral, tetrahedral, and hexahedral primitive elements. The finite volume method uses a k-exact reconstruction method, which, in theory, can resolve arbitrary convergence accuracies; however, orders above fourth order were not tested. Monotonicity enforcement for the finite volume method, which was required for strong dynamics in nonlinear equation sets, was handled using a central essentially-non-oscillatory (CENO) method, which was de-

signed to be a more computationally efficient method than the alternative weighted ENO methods.

The nodal discontinuous Galerkin method was developed and implemented for line, triangle, and tetrahedral primitive element geometries. The method was designed to work with any user supplied node arrangements. The convergence accuracy for the implemented node arrangements have been shown to increase with the number of nodes in an element for one, two, and three dimensions, which is consistent with theory. Monotonicity enforcement for the discontinuous Galerkin method was implemented using the CENO method that was implemented for the finite volume method. Diffusion operators in the discontinous Galerkin method were applied using a local discontinuous Galerkin approach.

The time integration schemes implemented in WARPXM are designed around efficient parallelization; however, there are alternative methods, such as implicit, semi-implicit, and implicit-explicit methods that can be used to alleviate the stringent time step restrictions of explicit methods. Continuous methods, such as the continuous Galerkin method, would be a good addition to WARPXM once more advanced time integration schemes are in place. Continuous methods coupled to implicit time integration are more capable in the presence of the parabolic behavior associated with charge separation and diffusive collisional transport. Ideally, these implicit continuous methods would be used for electron and electromagnetic dynamics, while heavier ion species would use the existing explicit discontinuous framework. This approach would greatly reduce runtime by removing the stability condition associated with the electron plasma frequencies and the speed of light.

6.3 Computational framework

The WARPXM code has been developed as a effective and easy-to-use interface for developing multi-species plasma models. The goal of the code is to provide the developer to tools for implementing plasma models in a well-known language which can be optimized for any

available compute devices in a given compute environment.

The primary focus of this research has been to develop an OpenCL interface for solving discontinuous numerical methods. The OpenCL threading interface allows for plasma model operators, such as Lorentz force operators or fluid viscosity operators, to be written in a single language, that work on both CPU and GPU devices. This language is based on the C standard which allows the models to be reused in any C based parallelization library, including OpenMP or CUDA. While these parallelization libraries are not currently implemented, WARPXM is designed to allow their use.

The OpenCL modules in WARPXM have been designed around a low memory footprint. Memory allocations are reused as frequently as possible to reduce the overall load on the compute device, which makes the solvers more applicable to GPU devices. Shared memory spaces are used in the kernels to optimize cache reusability, while the variable datasets are optimized for fast read and write operations used in ghost sync and file input/output processes. The OpenCL interface is compiled at runtime which allows for runtime optimization for the different compute devices found in heterogeneous compute environments.

Over the course of this research, WARPXM was adapted to work with unstructured meshes. This unstructured mesh interface was designed to import meshes written in the ABAQUS mesh format. These ABAQUS files can be generated from CAD models using mesh generation toolkits, such as CUBIT, Trelis, or gmsh. This gives WARPXM the capability to model the complex geometries associated with modern plasma confinement systems. The datasets generated by WARPXM are written in a proprietary file format based on the hdf5 format which can be post-processed in scripting languages, such as Python or MATLAB, or analyzed in data visualization software such as VisIt or Paraview using an XDMF translation file.

Solvers in WARPXM are modular, meaning that different spatial solvers (e.g. finite volume or discontinuous Galerkin) can be implemented in conjunction with different time integration

schemes. The models used to describe physics are also modular, meaning that models can be mixed and matched as specified by the WARPXM input file. Solvers are designed to hide memory movement with computations. This has been accomplished by breaking the solver's domain into periphery and interior regions. The solver begins with the periphery region, which is used in the ghost sync process. While the ghost elements are synchronizing, the solver is running on the interior region. This effectively hides the latency associated with the host level and cluster level synchronization latencies, which can be a large burden for multi-species models containing hundreds of variables.

WARPXM has been developed for modeling plasmas exhibiting a wide variety of behaviors, and one of its objectives is to handle multiple plasma representations concurrently. This modification to the framework would allow plasma models to be swapped out during run-time depending on the local plasma parameter regime. WARPXM has various multi-species models of both the 5-moment and 13-moment variety, and is currently being extended to include non-equilibrium MHD and Boltzmann models. Ideally, these models would be applied in different areas of the domain in order balance physical accuracy with computational cost. For example, a high-energy particle beam or laser may drive a plasma out of thermal equilibrium around a localized point. Instead of modeling the entire domain using a kinetic representation, only the area around the source would be represented kinetically, while everywhere else in the domain could be represented by a more computationally inexpensive model, such as a 13-moment or MHD model. Such a framework would be advantageous for the transitional, high-energy density plasmas found in tokamaks, where collisionality transitions from collisionless in the core region to highly collisional in the divertor, and z-pinches, where the plasma is unmagnetized on-axis and highly magnetized at the edge of the pinch.

The WARPXM code continues to be a work in progress, and is designed to be adaptable to emerging compute environments. The adaptability of WARPXM to the newer generation of parallelization libraries makes it a productive platform for developing both numerical methods and plasma models.

REFERENCES

[1] J. Birn, J. F. Drake, M. A. Shay, B. N. Rogers, R. E. Denton, M. Hesse, M. Kuznetsova, Z. W. Ma, A. Bhattacharjee, A. Otto, et al. Geospace environmental modeling (gem) magnetic reconnection challenge. *Journal of Geophysical Research: Space Physics*, 106 (A3):3715–3719, 2001.

[2] E. T. Meier and U. Shumlak. A general nonlinear fluid model for reacting plasma-neutral mixtures. *Physics of Plasmas*, 19(7):072508, 2012.

[3] E. T. Meier. *Modeling plasmas with strong anisotropy, neutral fluid effects, and open boundaries*. PhD thesis, University of Washington, 2011.

[4] M. P. Bachynski. Electromagnetic wave penetration of reentry plasma sheaths. *Journal of Research of the National Bureau of Standards Section D-Radio Science*, 69(2):147, 1965.

[5] C. He, T. C. Corke, and M. P. Patel. Plasma flaps and slats: an application of weakly ionized plasma actuators. *Journal of Aircraft*, 46(3):864–873, 2009.

[6] Yuri P Raizer and John E Allen. *Gas discharge physics*. Springer Berlin, 1997.

[7] U. Shumlak, B. A. Nelson, R. P. Golingo, S. L. Jackson, E. A. Crawford, and D. J. Den Hartog. Sheared flow stabilization experiments in the zap flow z pinch. *Physics of Plasmas*, 10(5):1683–1690, 2003.

[8] J. Wesson and D.J. Campbell. *Tokamaks*. OUP Oxford, 2011.

[9] J. P. Freidberg. *Plasma Physics and Fusion Energy*. Cambridge University Press, 2007.

[10] Y. Sone. *Molecular gas dynamics: theory, techniques, and applications*. Springer Science & Business Media, 2007.

[11] S. I. Braginskii. Transport processes in a plasma. *Reviews of plasma physics*, 1:205, 1965.

[12] C. D. Levermore. Moment closure hierarchies for kinetic theories. *Journal of Statistical Physics*, 83(5-6):1021–1065, 1996.

[13] A. Hakim, J. Loverich, and U. Shumlak. A high resolution wave propagation scheme for ideal two-fluid plasma equations. *Journal of Computational Physics*, 219(1):418–442, 2006.

[14] J. Loverich, A. Hakim, and U. Shumlak. A discontinuous galerkin method for ideal two-fluid plasma equations. *Communications in Computational Physics*, 9(02):240–268, 2011.

[15] B. Srinivasan and U. Shumlak. Analytical and computational study of the ideal full two-fluid plasma model and asymptotic approximations for hall-magnetohydrodynamics. *Physics of Plasmas*, 18(9):092113, 2011.

[16] U. Shumlak, R. Lilly, N. Reddell, E. Sousa, and B. Srinivasan. Advanced physics calculations using a multi-fluid plasma model. *Computer Physics Communications*, 182 (9):1767–1770, 2011.

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

[18] C. P. T. Groth and J. G. McDonald. Towards physically realizable and hyperbolic moment closures for kinetic theory. *Continuum Mechanics and Thermodynamics*, 21(6): 467–493, 2009.

[19] M. Torrilhon. Hyperbolic moment equations in kinetic gas theory based on multivariate pearson-iv-distributions. *Communications in Computational Physics*, 7(4):639–673, 2010.

[20] Z. Cai, Y. Fan, and R. Li. Globally hyperbolic regularization of grad's moment system. *Communications on Pure and Applied Mathematics*, 67(3):464–518, 2014.

[21] M. Torrilhon. Regularization of grad's 13-moment-equations in kinetic gas theory. Technical report, DTIC Document, 2011.

[22] L. Wang, A. H. Hakim, A. Bhattacharjee, and K. Germaschewski. Comparison of multi-fluid moment models with particle-in-cell simulations of collisionless magnetic reconnection. *Physics of Plasmas*, 22(1):012108, 2015.

[23] J. Ng, Y. Huang, A. Hakim, A. Bhattacharjee, A. Stanier, W. Daughton, L. Wang, and K. Germaschewski. The island coalescence problem: Scaling of reconnection in extended fluid models including higher-order moments. *Physics of Plasmas*, 22(11):112104, 2015.

[24] P. M. Bellan. *Fundamentals of Plasma Physics*. Cambridge University Press, 2008.

[25] F. L. Hinton. Collisional transport in plasma. *Handbook of Plasma Physics*, 1:147, 1983.

[26] O. Pezzi, F. Valentini, and P. Veltri. Collisional relaxation: Landau versus dougherty operator. *Journal of Plasma Physics*, 81(01):305810107, 2015.

[27] H. Fehske, R. Schneider, and A. Weiße. *Computational Many-Particle Physics*. Lecture Notes in Physics. Springer Berlin Heidelberg, 2010.

[28] R. J. Goldston and P. H. Rutherford. *Introduction to plasma physics*. CRC Press, 1995.

[29] T. Islam and S. Balbus. Dynamics of the magnetoviscous instability. *The Astrophysical Journal*, 633(1):328, 2005.

[30] R J Hosking and G M Marinoff. Magneto-viscous effects on the ideal and resistive gravitational instabilities in cartesian geometry. *Plasma Physics*, 15(5):327, 1973.

[31] R. J. Goldston and P. H. Rutherford. *Introduction to Plasma Physics*. Taylor & Francis, 1995.

[32] P. J. Catto, W. M. Tang, and D. E. Baldwin. Generalized gyrokinetics. *Plasma Physics*, 23(7):639, 1981.

[33] A. J. Brizard and T. S. Hahm. Foundations of nonlinear gyrokinetic theory. *Rev. Mod. Phys.*, 79(2):421–468, 2007.

[34] C. D. Munz, P. Omnes, R. Schneider, E. Sonnendrücker, and U. Voss. Divergence correction techniques for maxwell solvers based on a hyperbolic model. *Journal of Computational Physics*, 161(2):484–511, 2000.

[35] F. Kemm, C. D. Munz, R. Schneider, and E. Sonnendrcker. Divergence corrections in the numerical simulation of electromagnetic wave propagation. In *Hyperbolic Problems: Theory, Numerics, Applications*, volume 141 of *ISNM International Series of Numerical Mathematics*, pages 603–612. Birkhuser Basel, 2001.

[36] R. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

[37] J. S. Hesthaven and T. Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer, 2007.

[38] B. Srinivasan, A. Hakim, and U. Shumlak. Numerical methods for two-fluid dispersive fast mhd phenomena. *Communications in Computational Physics*, 10(1):183, 2011.

[39] George Karniadakis and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

[40] G. A. Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of computational physics*, 27(1):1–31, 1978.

[41] L. Ivan. *Development of high-order CENO finite-volume schemes with block-based adaptive mesh refinement*. PhD thesis, University of Toronto, 2011.

[42] S. McDonald. Development of a high-order finite-volume method for unstructured meshes. Master's thesis, University of Toronto, 2011.

[43] B. van Leer. Towards the ultimate conservative difference scheme. ii. monotonicity and conservation combined in a second-order scheme. *Journal of Computational Physics*, 14 (4):361 – 370, 1974.

[44] K. Michalak and C. Ollivier-Gooch. Limiters for unstructured higher-order accurate solutions of the euler equations. In *Proceedings of the AIAA forty-sixth aerospace sciences meeting*, 2008.

[45] K. Michalak and C. Ollivier-Gooch. Differentiability of slope limiters on unstructured grids. In *Proceedings of fourteenth annual conference of the computational fluid dynamics society of Canada*, 2006.

[46] C. Hu and C. W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150(1):97–127, 1999.

[47] B. Cockburn and C. W. Shu. The local discontinuous galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6): 2440–2463, 1998.

[48] R. M. Kirby and G. E. Karniadakis. Selecting the numerical flux in discontinuous galerkin methods for diffusion problems. *Journal of Scientific Computing*, 22(1-3):385–411, 2005.

[49] D. N. Arnold, F. Brezzi, B. Cockburn, and D. Marini. Discontinuous galerkin methods for elliptic problems. In *Discontinuous Galerkin Methods*, pages 89–101. Springer, 2000.

[50] D. N. Arnold, F. Brezzi, B. Cockburn, and D. Marini. Unified analysis of discontinuous galerkin methods for elliptic problems. *SIAM journal on numerical analysis*, 39(5): 1749–1779, 2002.

[51] X. Zhong and C. W. Shu. A simple weighted essentially nonoscillatory limiter for rungekutta discontinuous galerkin methods. *Journal of Computational Physics*, 232(1): 397 – 415, 2013.

[52] B. Srinivasan. *Numerical methods for 3-dimensional magnetic confinement configurations using two-fluid plasma equations*. PhD thesis, University of Washington, 2010.

[53] R. Franke, W. Rodi, and B. Schnung. Numerical calculation of laminar vortex-shedding flow past cylinders. *Journal of Wind Engineering and Industrial Aerodynamics*, 35: 237–257, 1990.

[54] O. S. Jones, U. Shumlak, and D. S. Eberhardt. An implicit scheme for nonideal magnetohydrbodynamics. *Journal of Computational Physics*, 130(2):231–242, 1997.

[55] P. Ricci, G. Lapenta, and J. U. Brackbill. Gem reconnection challenge: Implicit kinetic simulations with the physical mass ratio. *Geophysical research letters*, 29(23), 2002.

[56] A. Hakim. Extended mhd modelling with the ten-moment equations. *Journal of Fusion Energy*, 27(1-2):36–43, 2008.

[57] N. Reddell. *A kinetic vlasov model for plasma simulation using discontinuous Galerkin method on many-core architectures*. PhD thesis, University of Washington, 2016.

[58] L. Zhang, T. Cui, and H. Liu. A set of symmetric quadrature rules on triangles and tetrahedra. *Journal of Computational Mathematics*, 27(1), 2009.

Appendix A

UNSTRUCTURED MESH GEOMETRY

Unstructured meshes are construction from general-geometry elements. For the finite volume and discontinuous Galerkin methods used in this study, the geometry of the mesh is important when considering face normals for numerical flux calculations (Sec. 3.1.1), Jacobian matrices for coordinate transformations (Sec. 3.3), and quadrature rules for flux and source term integration (Sec. 3.2). For discontinuous discretizations, it is common practice to map the element to an isoparametric form through an isoparametric coordinate transformation (ICT). The ICT defines the configuration coordinate $\vec{x}$ to a normalized coordinate system $\vec{\xi}$. In general this mapping can be accomplished using any basis dependent on $\vec{\xi}$, however, for the geometry considered in this study, a monomial series,

$$\vec{x}(\vec{\xi}) = \sum_i \vec{\Gamma}_i \xi_0^{a_i} \xi_1^{b_i} \xi_2^{c_i}, \tag{A.0.1}$$

is sufficient. For linear ICTs, such as those used in this study, the integer powers a_i, b_i, and c_i are either 0 or 1. The transform coefficients $\vec{\Gamma}_i$ are entirely dependent on the element's nodal coordinates. For this study, the isoparametric forms define three dimensional slab geometries. For example, a slab line element describes a rectangular prism whose length is equal to the length of the line element, and width and height are 1. Slab triangles and slab quadrilaterals would then define triangular and quadrilateral prisms of unit height. The purpose of the ICT is to simplify the evaluation of all geometry terms required by discontinuous numerical methods.

When a mesh generating program, e.g CUBIT, exports a mesh, it contains a list of vertex

positions and a list of vertex indexes that make up each element, known as the connectivity array (see Sec. 4.2). The order of the vertex indexes in the connectivity array describe the shape of an element. Since each face of a given primitive is in itself a primitive (e.g. the faces of a tetrahedra are triangles), the following sections also define the face connectivity arrays $\mathcal{Z}_f$. The face connectivity array specifies the vertexes in the element connectivity array that make up the face primitive. The face connectivity arrays are designed to be consistent with the element connectivity arrays for the face primitive. For example, the quadrature integration of a polynomial on the surface of a tetrahedra, can be broken into volume quadrature integrals over triangular elements defined by vertex connectivities $\mathcal{Z}_f$.

In the following sections, superscripts represent the primitive ID from Tab. 4.1. To simplify the geometry relations a normalization operator is define as

$$\vec{\mathcal{N}}(\vec{x}) = \frac{\vec{x}}{|\vec{x}|}. \tag{A.0.2}$$

A.1 Preface for quadrature rules

When discussing quadrature rules for primitive geometries, it is important to first discuss some background. Quadrature rules are designed to convert an integral into a summation of point specific evaluations

$$\int f(\vec{x})\, dV = \sum_{i=0}^{i<m} w_i f(\vec{x}_i) \tag{A.1.1}$$

where m is the number of quadrature weights, w_i, and points, $\vec{x}_i$. In terms of Sec. 3.2, quadrature integration on surfaces,

$$\int f\, dA = \sum_{i}^{N_A} \omega_i f(\vec{x}_i), \tag{A.1.2}$$

is written in terms of N_A surface quadrature points with weights ω_i. Volume integration,

$$\int f\, dV = \sum_{i}^{N_V} \Omega_i f\left(\vec{x}_i\right), \tag{A.1.3}$$

is described by N_V quadrature weights Ω_i. In general, these quadrature rules are derived for unit primitives (e.g. a line element of length 1), which can be converted to work in general geometry space using the ICTs described in later sections.

Quadrature rules for polynomials are described both by their number of quadrature points and the quadrature order for which an exact integral of a polynomial can be found. The number of points required to integrate a polynomial of order p, defined in Eq. (3.2.5), over an element's volume is presented in Tab. A.1. The table shows that for the used quadrature rules, the quadrilateral and hexahedral quadrature rules are far less efficient than that of the triangle and tetrahedron.

Table A.1: The number of points m required for exact slab volume quadrature integration of a polynomial of order p for various primitives within WARPXM.

	p				
	3	5	7	9	11
Slab Line	2	3	4	5	6
Slab Triangle	6	7	15	19	28
Slab Quadrilateral	4	9	16	25	36
Tetrahedron	8	14	36	61	109
Hexahedron	8	27	64	125	216

Quadrature rules for slab lines, slab quadrilaterals and hexahedra are related to m point Gauss-Legendre quadrature rules with abscissa ξ_i^0 and weights w_i. Example Gauss-Legendre quadrature rules are given in Tab. A.2.

Table A.2: Gauss-Legendre weights w_i and abscissa ξ_i^0 for $m = 2, 3, 4$. Additional weights and abscissa can be found in Hesthaven and Warburton[37].

m	w_i	ξ_i^0
2	1	$\pm\frac{1}{\sqrt{3}}$
3	$\frac{8}{9}$, $\frac{5}{9}$	0, $\pm\sqrt{\frac{3}{5}}$
4	$\frac{1}{2}+\frac{1}{7}\sqrt{\frac{6}{5}}$, $\frac{1}{2}-\frac{1}{7}\sqrt{\frac{6}{5}}$	$\pm\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}$, $\pm\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$

Slab triangle and tetrahedral quadrature rules are related to the m point symmetric quadrature rules[58] with abscissa $(\xi_i^0, \xi_i^1, \xi_i^2)$ and weights w_i. Example quadrature rules are listed in Tab. A.3 for slab triangles, and Tab. A.4 for tetrahedra. These quadrature definitions are used in the following sections for the primitive element types presented.

Table A.3: Symmetric quadrature weights w_i and abscissa ξ_i^0, ξ_i^1 for $m = 3, 4, 6$ on triangles. Additional quadrature points can be found in Zhang et al.[58].

m	w_i	(ξ_i^0, ξ_i^1)
3	$\frac{1}{3}$, $\frac{1}{3}$, $\frac{1}{3}$	$(\frac{2}{3}, \frac{1}{6})$, $(\frac{1}{6}, \frac{2}{3})$, $(\frac{1}{6}, \frac{1}{6})$
4	$-\frac{9}{16}$, $\frac{25}{48}$, $\frac{25}{48}$, $\frac{25}{48}$	$(\frac{1}{3}, \frac{1}{3})$, $(\frac{3}{5}, \frac{1}{5})$, $(\frac{1}{5}, \frac{3}{5})$, $(\frac{1}{5}, \frac{1}{5})$
6	0.223381589678011	(0.108103018168070, 0.445948490915965)
	0.223381589678011	(0.445948490915965, 0.108103018168070)
	0.223381589678011	(0.445948490915965, 0.445948490915965)
	0.109951743655322	(0.816847572980459, 0.091576213509771)
	0.109951743655322	(0.091576213509771, 0.816847572980459)
	0.109951743655322	(0.091576213509771, 0.091576213509771)

Table A.4: Symmetric quadrature weights w_i and abscissa $\xi_i^0, \xi_i^1, \xi_i^2$ for $m = 4, 5, 10$ on tetrahedra. Additional quadrature points can be found in Zhang et al.[58].

m	w_i	$(\xi_i^0, \xi_i^1, \xi_i^2)$
4	0.25	$(0.5854101966249685, 0.1381966011250105, 0.1381966011250105)$
	0.25	$(0.1381966011250105, 0.5854101966249685, 0.1381966011250105)$
	0.25	$(0.1381966011250105, 0.1381966011250105, 0.5854101966249685)$
	0.25	$(0.1381966011250105, 0.1381966011250105, 0.1381966011250105)$
5	$-.8$	$(0.25, 0.25, 0.25)$
	0.45	$(\frac{1}{4}, \frac{1}{4}, \frac{1}{4})$
	0.45	$(\frac{1}{2}, \frac{1}{6}, \frac{1}{6})$
	0.45	$(\frac{1}{6}, \frac{1}{2}, \frac{1}{6})$
	0.45	$(\frac{1}{6}, \frac{1}{6}, \frac{1}{2})$
		$(\frac{1}{6}, \frac{1}{6}, \frac{1}{6})$
10	0.2177650698804054	$(0.5684305841968444, 0.1438564719343852, 0.1438564719343852)$
	0.2177650698804054	$(0.1438564719343852, 0.1438564719343852, 0.1438564719343852)$
	0.2177650698804054	$(0.1438564719343852, 0.1438564719343852, 0.5684305841968444)$
	0.2177650698804054	$(0.1438564719343852, 0.5684305841968444, 0.1438564719343852)$
	0.0214899534130631	$(0, 0.5, 0.5)$
	0.0214899534130631	$(0.5, 0, 0.5)$
	0.0214899534130631	$(0.5, 0.5, 0)$
	0.0214899534130631	$(0.5, 0, 0)$
	0.0214899534130631	$(0, 0.5, 0)$
	0.0214899534130631	$(0, 0, 0.5)$

A.2 Slab line

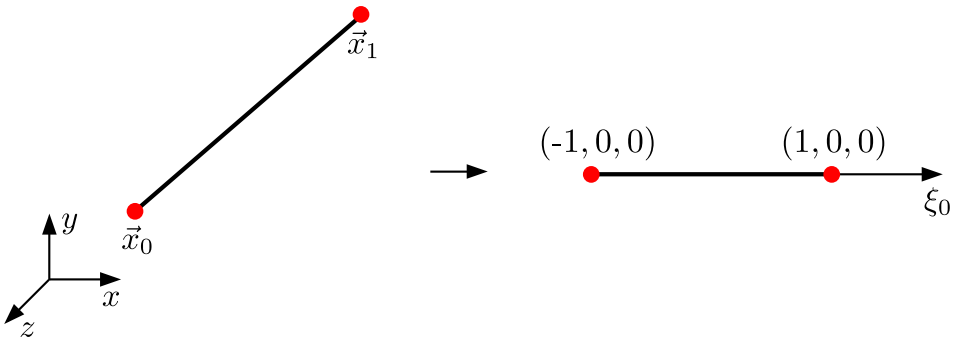


Figure A.2.1: Coordinate transform of line primitive from real space $\vec{x}$ to isoparametric space $\vec{\xi}$.

The line primitive mapping from real space $\vec{x}$ to isoparametric space $\xi_0 \in (-1, 1)$, shown in Fig. A.2.1, is defined by the ICT

$$\vec{x} = \vec{\Gamma}_0^0 + \vec{\Gamma}_1^0 \xi_0 \tag{A.2.1}$$

with coordinate transform coefficients

$$\vec{\Gamma}_0^0 = \frac{1}{2}(\vec{x}_0 + \vec{x}_1), \tag{A.2.2}$$

and

$$\vec{\Gamma}_1^0 = \vec{\Gamma}_0^0 - \vec{x}_0. \tag{A.2.3}$$

Given this coordinate transform, the volume of the slab line is given by

$$V^0 = 2\left|\vec{\Gamma}_1^0\right|. \tag{A.2.4}$$

The quadrature components for volume integration are given by: $N_V^0 = m$, $\Omega_i^0 = V^0 w_i$, $\tilde{x}_i = \vec{\Gamma}_0^0 + \xi_i^0 \vec{\Gamma}_1^0$. An example $m = 3$ quadrature rule (5^{th} order exact polynomial integration) for slab line elements is shown in Fig. A.2.2.

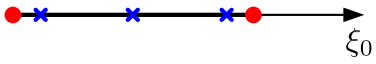


Figure A.2.2: Gauss-Legendre quadrature points (blue) with $m = 3$ on slab line.

Since this is a slab geometry, the integrand in a surface integral is assumed constant on each face. This implies that face integrals are single point evaluations with $N_A^0 = 1$ and $\omega_0 = 1$. For each face the connectivity, face normals, and quadrature points are given by:

1. Face 0:

$$\mathcal{Z}_0^0 = \left[\ 0\ \right]$$
$$\hat{n}_0^0 = -\vec{\mathcal{N}}\left(\vec{\Gamma}_1^0\right)$$
$$\tilde{x}_0^0 = \vec{\Gamma}_0^0 - \vec{\Gamma}_1^0$$

2. Face 1:

$$\mathcal{Z}_1^0 = \left[\ 1\ \right]$$
$$\hat{n}_1^0 = \vec{\mathcal{N}}\left(\vec{\Gamma}_1^0\right)$$
$$\tilde{x}_0^0 = \vec{\Gamma}_0^0 + \vec{\Gamma}_1^0$$

A.3 Slab triangle

The isoparametric coordinate transformation for a triangle from general geometry space $\vec{x}$ to isoparametric space $\vec{\xi}$, shown in Fig. A.3.1, is based on a barycentric coordinate trans-

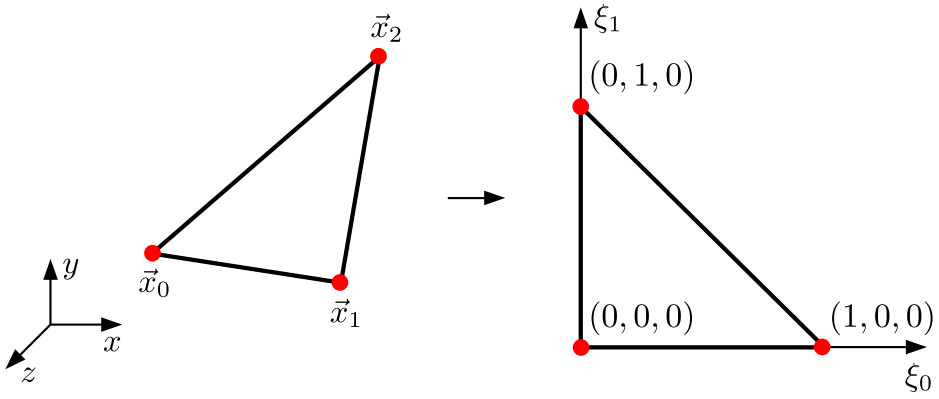


Figure A.3.1: Coordinate transform of triangle from real space $\vec{x}$ to isoparametric space $\vec{\xi}$.

form

$$\vec{x} = \vec{\Gamma}^1_0 + \vec{\Gamma}^1_1 \xi_0 + \vec{\Gamma}^1_2 \xi_1 \tag{A.3.1}$$

with coordinate transform coefficients

$$\vec{\Gamma}^1_0 = \vec{x}_0, \tag{A.3.2}$$

$$\vec{\Gamma}^1_1 = \vec{x}_1 - \vec{x}_0, \tag{A.3.3}$$

and

$$\vec{\Gamma}^1_2 = \vec{x}_2 - \vec{x}_0. \tag{A.3.4}$$

The isoparametric vertexes are located at $\vec{\xi} = (0,0,0), (1,0,0), (0,1,0)$. This coordinate transform can be used to define the slab triangle volume

$$V^1 = \frac{1}{2} \left| \vec{\Gamma}^1_1 \times \vec{\Gamma}^1_2 \right| \tag{A.3.5}$$

The volume quadrature terms for slab triangles are given by $N_V^1 = m$, $\Omega_i^1 = V^1 w_i$ and $\tilde{x}_i = \vec{\Gamma}_0^1 + \vec{\Gamma}_1^1 \xi_i^0 + \vec{\Gamma}_2^1 \xi_i^1$, where w_i and $\vec{\xi}_i$ are described in Tab. A.3. An example $m = 7$ quadrature rule (5^{th} order exact polynomial integration) for the slab triangle primitive is seen in Fig. A.3.2.

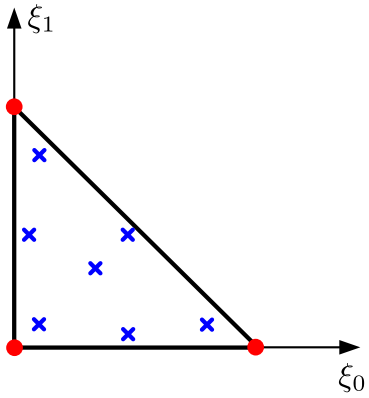


Figure A.3.2: Symmetric quadrature points (blue) with $m = 7$ on slab triangle.

The surfaces of a slab triangle are represented by line primitives meaning that surface quadrature integration is implemented as a slab line volume quadrature. Each face is represented by a coordinate transformation to the slab line element. The surface quadrature terms are given by $N_A^1 = N_V^0$, $\omega_i^1 = \Omega_i^0$ and $\tilde{x}_i = \vec{\Gamma}_0^0 + \xi_i^0 \vec{\Gamma}_1^0$. The surface normals are defined with respect to the triangle surface vector,

$$\vec{s}^1 = \vec{\Gamma}_1^1 \times \vec{\Gamma}_2^1, \tag{A.3.6}$$

and the face-specific transform coefficient $\vec{\Gamma}_1^0$ such that for each face

$$\hat{n}^1 = \vec{\mathcal{N}} \left(\vec{\Gamma}_1^0 \times \vec{s}^1 \right). \tag{A.3.7}$$

The face connectivities and face transform coefficients (i.e. line element coordinate transform

coefficients) are defined as follows:

1. Face 0:

$$\begin{aligned}\mathcal{Z}_0^1 &= \begin{bmatrix} 0 & 1 \end{bmatrix} \\ \vec{\Gamma}_0^0 &= \vec{\Gamma}_0^1 + \frac{1}{2}\left(\vec{\Gamma}_1^1 + \vec{\Gamma}_2^1\right) \\ \vec{\Gamma}_1^0 &= \frac{1}{2}\left(\vec{\Gamma}_2^1 - \vec{\Gamma}_1^1\right)\end{aligned}$$

2. Face 1:

$$\begin{aligned}\mathcal{Z}_1^1 &= \begin{bmatrix} 1 & 2 \end{bmatrix} \\ \vec{\Gamma}_0^0 &= \vec{\Gamma}_0^1 + \frac{1}{2}\vec{\Gamma}_2^1 \\ \vec{\Gamma}_1^0 &= -\frac{1}{2}\vec{\Gamma}_2^1\end{aligned}$$

3. Face 2:

$$\begin{aligned}\mathcal{Z}_2^1 &= \begin{bmatrix} 2 & 0 \end{bmatrix} \\ \vec{\Gamma}_0^0 &= \vec{\Gamma}_0^1 + \frac{1}{2}\vec{\Gamma}_1^1 \\ \vec{\Gamma}_1^0 &= -\frac{1}{2}\vec{\Gamma}_1^1\end{aligned}$$

A.4 Slab quadrilateral

For quadrilateral primitives, the isoparametric coordinate transform, pictured in Fig. A.4.1, is bi-linear and is given by

$$\vec{x} = \vec{\Gamma}_0^2 + \vec{\Gamma}_1^2\xi_0 + \vec{\Gamma}_2^2\xi_1 + \vec{\Gamma}_3^2\xi_0\xi_1 \tag{A.4.1}$$

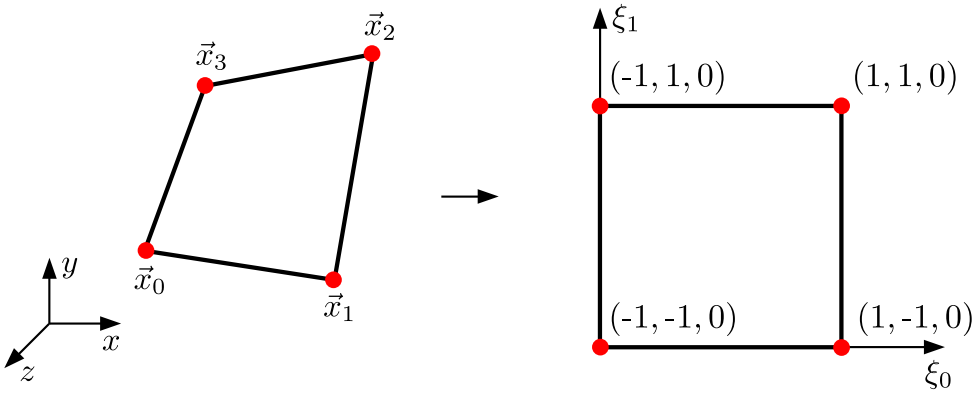


Figure A.4.1: Coordinate transform of quadrilateral from real space $\vec{x}$ to isoparametric space $\vec{\xi}$.

with transform coefficients

$$\vec{\Gamma}_0^2 = \frac{1}{4}\left(\vec{x}_0 + \vec{x}_1 + \vec{x}_2 + \vec{x}_3\right), \tag{A.4.2}$$

$$\vec{\Gamma}_1^2 = \vec{\Gamma}_0^2 - \frac{1}{2}\left(\vec{x}_0 + \vec{x}_3\right), \tag{A.4.3}$$

$$\vec{\Gamma}_2^2 = \vec{\Gamma}_0^2 - \frac{1}{2}\left(\vec{x}_0 + \vec{x}_1\right), \tag{A.4.4}$$

and

$$\vec{\Gamma}_3^2 = \vec{\Gamma}_0^2 - \frac{1}{2}\left(\vec{x}_1 + \vec{x}_3\right). \tag{A.4.5}$$

Vertex locations are found at $\vec{\xi} = (\pm 1, \pm 1, 0)$. The volume of the slab quadrilateral is

$$V^2 = 4\left|\vec{\Gamma}_1^2 \times \vec{\Gamma}_2^2\right|. \tag{A.4.6}$$

The volume quadrature for slab quadrilaterals is based on a tensor product of the m point 1D Gauss-Legendre quadrature such that $N_V^2 = m^2$ and the quadrature points are given

by

$$\tilde{x}_i = \vec{\Gamma}_0^2 + \vec{\Gamma}_1^2 \xi_{n_i}^0 + \vec{\Gamma}_2^2 \xi_{m_i}^1 + \vec{\Gamma}_3^2 \xi_{n_i}^0 \xi_{m_i}^1, \tag{A.4.7}$$

where ξ_i^0 and ξ_i^1 are given by the abscissa in Tab. A.2. Since the coordinate transform is bi-linear, the quadrature weights, $\Omega_i = w_{n_i} w_{m_i} J_{n_i m_i}$, where n_i and m_i represent the mapping of 2D coordinates to a 1D array, must take into account the coordinate transform Jacobian,

$$J_{ij} = \left(\vec{\Gamma}_2^2 + \xi_i^0 \vec{\Gamma}_3^2\right) \times \left(\vec{\Gamma}_1^2 + \xi_j^1 \vec{\Gamma}_3^2\right). \tag{A.4.8}$$

An example $m = 3$ quadrature rule (5^{th} order exact polynomial integration) for slab quadrilateral primitives is shown in Fig. A.4.2.

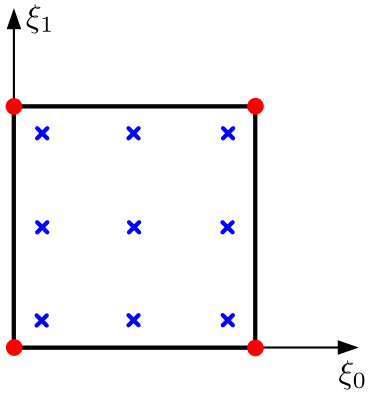


Figure A.4.2: Quadrature points (blue) with $m = 3$ on slab quadrilateral.

As was seen with the triangle primitive, the surface of a slab quadrilateral is represented by a set of line primitives. This implies that slab quadrilateral surface integration can be implemented using the slab line volume quadrature. The quadrilateral's surface normal

vector,

$$\vec{s}^2 = \vec{\Gamma}_1^2 \times \vec{\Gamma}_2^2, \tag{A.4.9}$$

is used to describe the face normal vectors,

$$\hat{n}^2 = \vec{\mathcal{N}}\left(\vec{\Gamma}_1^0 \times \vec{s}^2\right) \tag{A.4.10}$$

in terms of the face transform coefficient $\vec{\Gamma}_1^0$. The face connectivities, and associated face transformation coefficients, are as follows:

1. Face 0:

$$\begin{aligned}
\mathcal{Z}_0^2 &= \begin{bmatrix} 0 & 1 \end{bmatrix} \\
\vec{\Gamma}_0^0 &= \vec{\Gamma}_0^2 - \vec{\Gamma}_2^2 \\
\vec{\Gamma}_1^0 &= \vec{\Gamma}_1^2 - \vec{\Gamma}_3^2
\end{aligned}$$

2. Face 1:

$$\begin{aligned}
\mathcal{Z}_1^2 &= \begin{bmatrix} 1 & 2 \end{bmatrix} \\
\vec{\Gamma}_0^0 &= \vec{\Gamma}_0^2 + \vec{\Gamma}_1^2 \\
\vec{\Gamma}_1^0 &= \vec{\Gamma}_2^2 + \vec{\Gamma}_3^2
\end{aligned}$$

3. Face 2:

$$\begin{aligned}
\mathcal{Z}_2^2 &= \begin{bmatrix} 2 & 3 \end{bmatrix} \\
\vec{\Gamma}_0^0 &= \vec{\Gamma}_0^2 + \vec{\Gamma}_2^2 \\
\vec{\Gamma}_1^0 &= -\vec{\Gamma}_1^2 - \vec{\Gamma}_3^2
\end{aligned}$$

4. Face 3:

$$\mathcal{Z}_3^2 = \begin{bmatrix} 3 & 0 \end{bmatrix}$$
$$\vec{\Gamma}_0^0 = \vec{\Gamma}_0^2 - \vec{\Gamma}_1^2$$
$$\vec{\Gamma}_1^0 = -\vec{\Gamma}_2^2 + \vec{\Gamma}_3^2$$

A.5 Tetrahedron

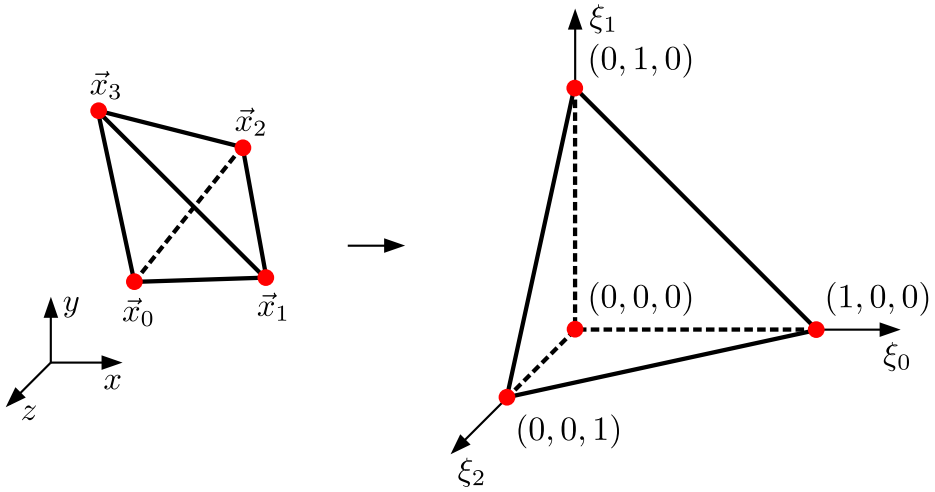


Figure A.5.1: Coordinate transform of tetrahedron from real space $\vec{x}$ to isoparametric space $\vec{\xi}$.

For the tetrahedron primitive, the coordinate transform from real space $\vec{x}$ to isoparametric space $\vec{\xi}$, shown in Fig. A.5.1, is given by

$$\vec{x} = \vec{\Gamma}_0^3 + \vec{\Gamma}_1^3 \xi_0 + \vec{\Gamma}_2^3 \xi_1 + \vec{\Gamma}_3^3 \xi_2 \tag{A.5.1}$$

which contains the coordinate transform coefficients

$$\vec{\Gamma}^3_0 = \vec{x}_0, \tag{A.5.2}$$

$$\vec{\Gamma}^3_1 = \vec{x}_1 - \vec{x}_0, \tag{A.5.3}$$

$$\vec{\Gamma}^3_2 = \vec{x}_2 - \vec{x}_0, \tag{A.5.4}$$

and

$$\vec{\Gamma}^3_3 = \vec{x}_3 - \vec{x}_0 \tag{A.5.5}$$

Given this definition, the volume of a tetrahedron is given by

$$V^3 = \frac{1}{6}\left(\vec{\Gamma}^3_1 \times \vec{\Gamma}^3_2\right) \cdot \vec{\Gamma}^3_3 \tag{A.5.6}$$

Note that this definition is only be positive for right-handed tetrahedron. The volume quadrature for tetrahedra is defined by $N^3_V = m$, $\Omega^3_i = w_i$, and

$$\tilde{x}_i = \vec{\Gamma}^3_0 + \vec{\Gamma}^3_1 \xi^0_i + \vec{\Gamma}^3_2 \xi^1_i + \vec{\Gamma}^3_2 \xi^2_i \tag{A.5.7}$$

using the values found in Tab. A.4. Since tetrahedra are made up of triangular faces, the tetrahedral surface integration uses slab triangle volume integration from Sec. A.3. Similarly, the face normals for a tetrahedron are given by each face's triangle surface normal (see Eq. (A.3.7)) given the following face connectivities and coordinate transform coefficients:

1. Face 0:

$$\mathcal{Z}_0^3 = \begin{bmatrix} 0 & 1 & 2 \end{bmatrix}$$
$$\vec{\Gamma}_0^1 = \vec{\Gamma}_0^3$$
$$\vec{\Gamma}_1^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_1^3$$
$$\vec{\Gamma}_2^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_2^3$$

2. Face 1:

$$\mathcal{Z}_1^3 = \begin{bmatrix} 0 & 1 & 3 \end{bmatrix}$$
$$\vec{\Gamma}_0^1 = \vec{\Gamma}_0^3$$
$$\vec{\Gamma}_1^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_3^3$$
$$\vec{\Gamma}_2^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_1^3$$

3. Face 2:

$$\mathcal{Z}_2^3 = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$
$$\vec{\Gamma}_0^1 = \vec{\Gamma}_0^3$$
$$\vec{\Gamma}_1^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_2^3$$
$$\vec{\Gamma}_2^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_3^3$$

4. Face 3:

$$\mathcal{Z}_3^3 = \begin{bmatrix} 0 & 2 & 3 \end{bmatrix}$$
$$\vec{\Gamma}_0^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_1^3$$
$$\vec{\Gamma}_1^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_3^3$$
$$\vec{\Gamma}_2^1 = \vec{\Gamma}_0^3 + \vec{\Gamma}_2^3$$

A.6 Hexahedron

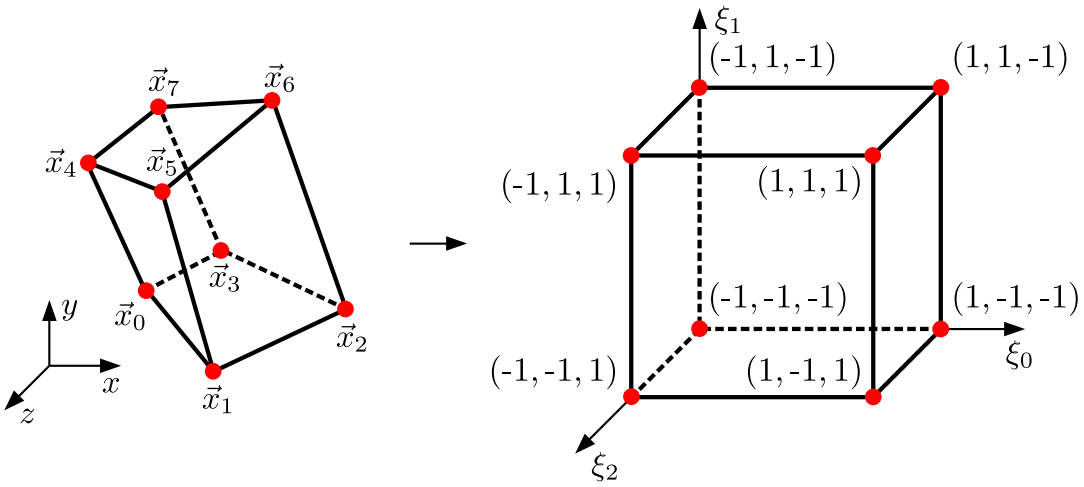


Figure A.6.1: Coordinate transform of hexahedron from real space $\vec{x}$ to isoparametric space $\vec{\xi}$.

Hexahedra are rarely used in unstructured, general geometry meshes due to the computational expense associated with their spatially dependent coordinate transform Jacobian. It is generally more efficient to use tetrahedra in three dimensional geometries, unless the mesh is Cartesian. The isoparametric coordinate transformation, from real space $\vec{x}$ to space $\vec{\xi}$ as shown in Fig. A.6.1, is tri-linear for hexahedrons such that

$$\vec{x} = \vec{\Gamma}_0^4 + \vec{\Gamma}_1^4\xi_0 + \vec{\Gamma}_2^4\xi_1 + \vec{\Gamma}_3^4\xi_2 + \vec{\Gamma}_4^4\xi_0\xi_1 + \vec{\Gamma}_5^4\xi_0\xi_2 + \vec{\Gamma}_6^4\xi_1\xi_2 + \vec{\Gamma}_7^4\xi_0\xi_1\xi_2 \tag{A.6.1}$$

with coordinate transform coefficients

$$\vec{\Gamma}_0^4 = \frac{1}{8}\left(\vec{x}_0 + \vec{x}_1 + \vec{x}_2 + \vec{x}_3 + \vec{x}_4 + \vec{x}_5 + \vec{x}_6 + \vec{x}_7\right), \tag{A.6.2}$$

$$\vec{\Gamma}_1^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_0 + \vec{x}_1 + \vec{x}_4 + \vec{x}_5\right), \tag{A.6.3}$$

$$\vec{\Gamma}_2^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_0 + \vec{x}_1 + \vec{x}_2 + \vec{x}_3\right), \tag{A.6.4}$$

$$\vec{\Gamma}_3^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_0 + \vec{x}_3 + \vec{x}_4 + \vec{x}_7\right), \tag{A.6.5}$$

$$\vec{\Gamma}_4^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_2 + \vec{x}_3 + \vec{x}_4 + \vec{x}_5\right), \tag{A.6.6}$$

$$\vec{\Gamma}_5^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_1 + \vec{x}_3 + \vec{x}_5 + \vec{x}_7\right), \tag{A.6.7}$$

$$\vec{\Gamma}_6^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_1 + \vec{x}_2 + \vec{x}_4 + \vec{x}_7\right), \tag{A.6.8}$$

and

$$\vec{\Gamma}_7^4 = \vec{\Gamma}_0^4 - \frac{1}{4}\left(\vec{x}_0 + \vec{x}_2 + \vec{x}_5 + \vec{x}_7\right). \tag{A.6.9}$$

The vertexes of the isoparametric primitive are found at $\vec{\xi} = (\pm 1, \pm 1, \pm 1)$. A hexahedron's volume is given by

$$V^4 = \left(\vec{\Gamma}_1^4 \times \vec{\Gamma}_2^4\right) \cdot \vec{\Gamma}_3^4 + \frac{2}{3}\left(\left(\vec{\Gamma}_4^4 \times \vec{\Gamma}_5^4\right) \cdot \vec{\Gamma}_1^4 + \left(\vec{\Gamma}_6^4 \times \vec{\Gamma}_4^4\right) \cdot \vec{\Gamma}_2^4 + \left(\vec{\Gamma}_5^4 \times \vec{\Gamma}_6^4\right) \cdot \vec{\Gamma}_3^4\right) \tag{A.6.10}$$

The volume quadrature for hexahedra is based on a tensor product of the m point 1D Gauss-Legendre quadrature. The quadrature rule is a $N_V^4 = m^3$ point scheme, with points

$$\tilde{x}_i = \vec{x}(\xi_{n_i}^0, \xi_{m_i}^1, \xi_{l_i}^2) \tag{A.6.11}$$

where ξ_i^0, ξ_i^1, and ξ_i^2 are the abcissa in Tab. A.2.

The tri-linear coordinate transform implies that the quadrature weights, $\Omega_i = w_{n_i} w_{m_i} w_{l_i} J_{n_i m_i l_i}$,

where n_i, m_i, and l_i represent the mapping of 3D coordinates to a 1D array, must take into account the coordinate transform Jacobian,

$$J_{ijk} = \left(\vec{A}_{jk} \times \vec{B}_{ik}\right) \cdot \vec{C}_{ij} \tag{A.6.12}$$

where

$$\vec{A}_{jk} = \partial_{\xi_0}\vec{x}|(0, \xi_j^1, \xi_k^2) = \vec{\Gamma}_1^4 + \vec{\Gamma}_4^4\xi_j^1 + \vec{\Gamma}_5^4\xi_k^2 + \vec{\Gamma}_7^4\xi_j^1\xi_k^2, \tag{A.6.13}$$

$$\vec{B}_{ik} = \partial_{\xi_1}\vec{x}|(\xi_i^0, 0, \xi_k^2) = \vec{\Gamma}_2^4 + \vec{\Gamma}_4^4\xi_i^0 + \vec{\Gamma}_6^4\xi_k^2 + \vec{\Gamma}_7^4\xi_i^0\xi_k^2, \tag{A.6.14}$$

and

$$\vec{C}_{ij} = \partial_{\xi_2}\vec{x}|(\xi_i^0, \xi_j^1, 0) = \vec{\Gamma}_3^4 + \vec{\Gamma}_5^4\xi_i^0 + \vec{\Gamma}_6^4\xi_j^1 + \vec{\Gamma}_7^4\xi_i^0\xi_j^1. \tag{A.6.15}$$

Hexahedra are made up of quadrilateral faces meaning the slab quadrilateral volume integration methods can be used. Face normals are set as the quadrilateral surface normal, given in Eq. (A.4.10), using the following definitions for face connectivities and coordinate transforms:

1. Face 0:

$$\begin{aligned}
\mathcal{Z}_0^4 &= \begin{bmatrix} 0 & 1 & 5 & 4 \end{bmatrix} \\
\vec{\Gamma}_0^2 &= \vec{\Gamma}_0^4 + \vec{\Gamma}_3^4 \\
\vec{\Gamma}_1^2 &= \vec{\Gamma}_1^4 + \vec{\Gamma}_5^4 \\
\vec{\Gamma}_2^2 &= \vec{\Gamma}_2^4 + \vec{\Gamma}_6^4 \\
\vec{\Gamma}_3^2 &= \vec{\Gamma}_4^4 + \vec{\Gamma}_7^4
\end{aligned}$$

2. Face 1:

$$\mathcal{Z}_1^4 = \begin{bmatrix} 2 & 3 & 7 & 6 \end{bmatrix}$$
$$\vec{\Gamma}_0^2 = \vec{\Gamma}_0^4 - \vec{\Gamma}_3^4$$
$$\vec{\Gamma}_1^2 = -\vec{\Gamma}_1^4 + \vec{\Gamma}_5^4$$
$$\vec{\Gamma}_2^2 = \vec{\Gamma}_2^4 - \vec{\Gamma}_6^4$$
$$\vec{\Gamma}_3^2 = -\vec{\Gamma}_4^4 + \vec{\Gamma}_7^4$$

3. Face 2:

$$\mathcal{Z}_2^4 = \begin{bmatrix} 0 & 4 & 7 & 3 \end{bmatrix}$$
$$\vec{\Gamma}_0^2 = \vec{\Gamma}_0^4 - \vec{\Gamma}_1^4$$
$$\vec{\Gamma}_1^2 = \vec{\Gamma}_2^4 - \vec{\Gamma}_4^4$$
$$\vec{\Gamma}_2^2 = -\vec{\Gamma}_3^4 + \vec{\Gamma}_5^4$$
$$\vec{\Gamma}_3^2 = -\vec{\Gamma}_6^4 + \vec{\Gamma}_7^4$$

4. Face 3:

$$\mathcal{Z}_3^4 = \begin{bmatrix} 1 & 2 & 6 & 5 \end{bmatrix}$$
$$\vec{\Gamma}_0^2 = \vec{\Gamma}_0^4 + \vec{\Gamma}_1^4$$
$$\vec{\Gamma}_1^2 = -\vec{\Gamma}_3^4 - \vec{\Gamma}_5^4$$
$$\vec{\Gamma}_2^2 = \vec{\Gamma}_2^4 + \vec{\Gamma}_4^4$$
$$\vec{\Gamma}_3^2 = -\vec{\Gamma}_6^4 - \vec{\Gamma}_7^4$$

5. Face 4:

$$\mathcal{Z}_4^4 = \begin{bmatrix} 0 & 3 & 2 & 1 \end{bmatrix}$$
$$\vec{\Gamma}_0^2 = \vec{\Gamma}_0^4 - \vec{\Gamma}_2^4$$
$$\vec{\Gamma}_1^2 = -\vec{\Gamma}_3^4 + \vec{\Gamma}_6^4$$
$$\vec{\Gamma}_2^2 = \vec{\Gamma}_1^4 - \vec{\Gamma}_4^4$$
$$\vec{\Gamma}_3^2 = -\vec{\Gamma}_5^4 + \vec{\Gamma}_7^4$$

6. Face 5:

$$\mathcal{Z}_5^4 = \begin{bmatrix} 4 & 5 & 6 & 7 \end{bmatrix}$$
$$\vec{\Gamma}_0^2 = \vec{\Gamma}_0^4 + \vec{\Gamma}_2^4$$
$$\vec{\Gamma}_1^2 = \vec{\Gamma}_1^4 + \vec{\Gamma}_4^4$$
$$\vec{\Gamma}_2^2 = -\vec{\Gamma}_3^4 - \vec{\Gamma}_6^4$$
$$\vec{\Gamma}_3^2 = -\vec{\Gamma}_5^4 - \vec{\Gamma}_7^4$$

The complexity of the surface and volume integration, both in its analytical and quadrature forms, is why tetrahedra primitives are generally used in unstructured systems.