A Domain-Hybridized Plasma Model Using Discontinuous Galerkin Finite Elements

Iman Datta

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

Doctor of Philosophy

University of Washington

2021

Reading Committee:

Uri Shumlak, Chair

Justin Little

Eric Meier

Program Authorized to Offer Degree:
Aeronautics & Astronautics

University of Washington

Abstract

A Domain-Hybridized Plasma Model Using Discontinuous Galerkin Finite Elements

Iman Datta

Chair of the Supervisory Committee:
Professor Uri Shumlak
Aeronautics & Astronautics

Plasmas are found all over the Universe and exhibit varying properties depending on temperature, density, field strength, and collisionality. Thus, various physics models can be used to describe plasma dynamics. This work uses the WARPXM discontinuous Galerkin finite element framework developed at the University of Washington to develop a domain-hybridized model incorporating MHD, multi-fluid, and continuum kinetic models to perform simulations in which different models are applied in different subdomains on an unstructured mesh, dictated by the underlying physics. As part of this work, the MHD and continuum kinetic models are developed, the latter in up to 2D2V using a duoprism phase space element constructed from tensor products of triangular elements in an unstructured physical space and lines in velocity space. The hybrid model is formulated using two methods. The first is a direct variable translation method which converts variables at model interfaces to be consistent with the lower fidelity model. The second is a conservative method which ensures the flux at model boundaries are equal and opposite for the adjacent models. The coupling between the continuum kinetic model and the multi-fluid model is specifically studied in this work using a variety of simulations to test these coupling methods.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Number Page

GLOSSARY

m: mass of particle

q: charge of particle

T: temperature

n: number density

$\boldsymbol{v}, \boldsymbol{u}$: velocity

$\boldsymbol{E}$: electric field vector

$\boldsymbol{B}$: magnetic field vector

ρ: mass density

ρ_c: charge density

$\boldsymbol{j}$: current density

k: Boltzmann constant

ϵ_0: permittivity of free space

μ_0: permeability of free space

c: speed of light, $c = \frac{1}{\sqrt{\epsilon_0 \mu_0}}$

Larmor radius, r_L: the radius of gyration of a charged particle moving in a magnetic field, $r_L = \frac{m v_\perp}{|q| B}$.

cyclotron (or gyro) frequency, ω_c: the frequency of gyration of a charged particle moving in a magnetic field, $\omega_c = \frac{v_\perp}{r_L} = \frac{qB}{m}$.

Debye length, λ_D: the scale over which mobile charge carriers screen out electric fields in plasmas and other conductors, $\lambda_D = \sqrt{\frac{\epsilon_0 k T_e}{n_e q_e^2}}$, assuming cold (unmoving) ions relative to electrons.

mean free path, λ_{mfp}: average distance traveled by a moving particle between successive collisions.

plasma frequency, ω_p: the typical electrostatic oscillation frequency of a given species in response to a small charge separation, $\omega_p = \sqrt{\frac{e^2 n}{\epsilon_0 m}}$.

Alfvén speed, V_A: speed at which Alfvén waves travel in which ions oscillate in response to a restoring force provided by an effective tension on magnetic field lines, $V_A = \frac{B}{\sqrt{mn\mu_0}}$.

skin depth δ_p: the depth in a plasma to which electromagnetic radiation can penetrate, $\delta_p = \frac{c}{\omega_p}$.

Coulomb logarithm, $\ln \Lambda$: the factor by which small-angle collisions are more important than large-angle collisions.

Field-reversed configuration, FRC: compact-toroidal magnetic system with small to no toroidal field and high plasma β

plasma β: ratio of plasma pressure to magnetic pressure, $\beta = \frac{nk_BT}{\frac{B^2}{2\mu_0}}$

Z-pinch: a plasma confinement device in which an azimuthal magnetic field generated by an axial plasma current compresses, or “pinches” the plasma so as to lead to favorable conditions required for fusion.

ACKNOWLEDGMENTS

I would like to express gratitude to the Department of Aeronautics & Astronautics for giving me the opportunity to pursue graduate school and the support I needed in making it through the program milestones. I am also greatly indebted to my advisor, Professor Uri Shumlak, who gave me the opportunity and support to pursue this degree, allowing me to grow as a graduate student and researcher as I learned about plasma physics and computational modeling.

I would like to express appreciation for the other graduate students I worked with in the Computational Plasma Dynamics Group, including Jack Coughlin, Daniel Crews, Andrew Ho, Aria Johansen, Sean Miller, Sina Taheri, Yu Takagaki, and Whitney Thomas. My discussions with them as well as the friends I made during this time helped me get through graduate school. Special thanks also go to Eric Meier, whose experience in computational plasma physics I leaned on for interpretation of various simulations, and Genia Vogman, who helped me to setup and understand some continuum kinetic simulations. I would also like to thank Susan Griffith for the enriching discussions we had and for keeping me focused on the progression of my degree.

I would like to also acknowledge the support from the Air Force Office of Scientific Research under grant number FA9550-15-1-0271 for much of this research as well as the computational resources this has provided me from the Department of Defense, including the Centennial, Mustang, and Onyx clusters. I am also grateful for the computational resources provided by the University of Washington Hyak cluster for nodes purchased through the PSI-Center as well as support provided by Zap Energy, Inc.

Finally, I would like to thank my family for their encouragement throughout graduate

school and my partner, Sarah Carter, for believing in me and standing by me while I concentrated on finishing my research and this dissertation.

DEDICATION

To my parents and brother

Chapter 1

INTRODUCTION

Most of the observable matter in the universe is in a plasma state, characterized as a quasineutral gas of charged and neutral particles exhibiting collective behavior. It is readily found in stellar interiors and atmospheres, gaseous nebulae, and as well as in interstellar hydrogen regions. Closer to home, one can find plasmas in the Van Allen radiation belts and in solar wind [22]. Human-produced plasmas appear in space-propulsion and fusion-generation devices and can be found being used in industrial and extractive metallurgy [46], plasma coating, etching of microelectronics [123], metal cutting [82], welding, and fluorescent lighting [42]. Its ubiquity in the universe and myriad of applications on Earth motivates the study of plasma physics, while at the same time, the fact that it is seen in many of these scenarios, spanning many different temperatures, densities, field strength, and particle collisionality, makes plasma physics a challenging field of study in which clever approaches to experimentation and computational modeling must be utilized in order to properly capture their dynamics.

With this in mind, it is understandable why various physics models describing plasmas have been developed. Particle-in-cell, continuum kinetic, multi-fluid, and magneto-hydrodyanamic descriptions all exist as separate trade-offs between accurate realization of phenomena and ease of computation, given a required fidelity in a specific regime. The balance between these two factors is always of importance in a computational simulation as one would always like to get the maximum-fidelity result using the least computational time and resources. However, it is not a given that the same model is the most appropriate throughout a given simulation domain and that this will not change in time. Thus, the goal of this work is to develop a computational framework which can apply multiple plasma models within a single simulation, in which the domain is decomposed into separate subdomains in which different models are used to solve for plasma dynamics and that these subdomains can be dynamically adjusted in time as conditions change.

1.1 Applications

A few relevant applications of this technique are given in the following subsections.

1.1.1 Plasma Photonic Crystals

Photonic crystals are periodic optical structures which affect the motions of photons through an array of repeating regions of high and low dielectric constant. They can be thought of as affecting the motion of photons through a material in much the same way as the motion of electrons are affected in a semiconductor. In these materials, certain bandgaps exist which can reflect specific frequencies of electromagnetic radiation while absorbing others. In nature, this structure is manifest in the wings of certain butterflies [10]. The phenomenon can be utilized anywhere the manipulation of light is desired. A currently-used application is in thin-film optics for coatings of lenses. Recently, researchers have studied the use of plasmas to make up photonic crystals, as opposed to conventional solid or liquid materials [104, 130], since plasmas exhibit the advantage of tunability. Specifically, electrical permittivity can be tuned via plasma and collision frequencies, which can be controlled by the power supply and pressure, respectively. Another advantage is that with plasmas the structure of the crystal is reconfigurable. There is also an inherent nonlinearity associated with using plasmas which represents an area of high scientific interest, including applications such as multi-harmonic-wave generation, memory effects with hysteresis, and high-power microwave switches. Other applications also involve wave scattering guards and blackout removal upon satellite or spacecraft reentry [104].

Thus, the area of plasma photonic crystals is an area of high scientific research interest. The spacing of the band gaps in these plasmas imply microplasmas and potential microinstabilities, representing problems on a vast range of scales in which a multi-model approach may be helpful.

1.1.2 Field-Reversed Configuration

Field-reversed configurations (FRC) are compact-toroidal magnetic systems with small to no toroidal magnetic field with high plasma β. The lack of toroidal field makes it a simpler device to construct as compared with other magnetic confinement fusion devices such as spheromaks, tokamaks, and stellarators. Despite this, these devices are challenging to understand physically, as they are famously very MHD-unstable [4] due to bad curvature, despite experimental results implying some level of stability [44], though rotational, tearing and tilting modes are still possible [111]. It is commonly believed kinetic effects especially in a low "null" magnetic field region account for this stabilization [111, 131]. Thus this device represents an interesting area of research for multi-plasma models to accurately describe the physics in a computationally efficient and physically accurate way. Work has already been done on modeling this device using a multi-fluid model in the past [51], though an argument exists for modeling this problem with a hybrid approach of Hall MHD, multi-fluid, and continuum kinetic models in appropriate regions.

1.1.3 Z-Pinch

A Z-pinch is a plasma confinement device in which an azimuthal magnetic field generated by an axial plasma current compresses, or "pinches" the plasma so as to lead to favorable conditions required for fusion. Its simple one-dimensional equilibrium and efficient use of confining magnetic field make it an attractive device for study, but is known to be MHD unstable from $m = 0$ sausage and $m = 1$ kink instabilities [50]. Despite this, the motivation for using such devices is high which has led to successful attempts to stabilize them, including that of shear flow stabilization [107]. Research into the understanding of this mechanism is however ongoing, and computational simulations of this device, using a plasma-neutral model as described by Meier [76] may help with diagnosis of the effect of the the neutral gas injection associated with the shear flow stabilized Z-pinch. In addition, a better understanding of stability can be gleaned by higher fidelity models as the sausage and kink instabilities are shown using ideal MHD, which neglects effects from resistivity, viscosity, thermal conduction, pressure anistotropy, finite Larmor radius, Hall terms, and kinetic phase mixing of particles [71]. Thus, the overall understanding of the dynamics of a Z-pinch could be greatly enhanced using the computational multimodel approach presented in this research.

Chapter 2

MODELS

2.1 *Defining Features*

As described in Chapter 1, plasmas are characterized as gases of charged and neutral particles exhibiting collective behavior. However, this behavior can vary drastically, depending on a few primary factors. These are the degree of magnetization, charge separation, and collisionality. These can all be measured by taking the ratio between a critical length and and reference length, L. For magnetization, the reference length is the Larmor radius, r_L. For charge separation, the reference length is the Debye length, λ_D, and for collisionality, the reference length is the mean free path, λ_{mfp}. The appropriate model for description of the plasma depends on the regime defined by these ratios.

2.2 *Model Description*

2.2.1 Continuum Kinetic Model

A complete description of a plasma would track the position and velocity of each particle as a function of time. The density in phase space can be written as

$$N_\alpha\left(\boldsymbol{x},\boldsymbol{v},t\right)=\sum_{1\le i\le \bar{N}_\alpha}\delta\left[\boldsymbol{x}-\boldsymbol{x}_i(t)\right]\delta\left[\boldsymbol{v}-\boldsymbol{v}_i(t)\right], \tag{2.2.1}$$

where $\bar{N}_\alpha=\int N_\alpha d\boldsymbol{x}d\boldsymbol{v}$ is the total number of particles of species α in the system. If the equations of motion of each particle, i are given by

$$\frac{d\boldsymbol{x}_i}{dt}=\boldsymbol{v}_i \tag{2.2.2}$$

and

$$\frac{d\boldsymbol{v}_i}{dt}=\frac{q_i}{m_i}\left(\boldsymbol{E}^M+\boldsymbol{v}_i\times\boldsymbol{B}^M\right)', \tag{2.2.3}$$

where the M signifies microscopic fields generated by particles and the $'$ signifies the force on particle i from all other particles excluding i itself, the total derivative of N_α with respect to t can be calculated

$$\frac{dN_\alpha(\boldsymbol{x},\boldsymbol{v},t)}{dt} = \frac{\partial N_\alpha(\boldsymbol{x},\boldsymbol{v},t)}{\partial t} + \boldsymbol{v}\cdot\frac{\partial N_\alpha(\boldsymbol{x},\boldsymbol{v},t)}{\partial \boldsymbol{x}} + \frac{q_\alpha}{m_\alpha}\left(\boldsymbol{E}^M + \boldsymbol{v}_i \times \boldsymbol{B}^M\right)'\cdot\frac{\partial N_\alpha(\boldsymbol{x},\boldsymbol{v},t)}{\partial \boldsymbol{v}} = 0. \tag{2.2.4}$$

This is the Klimontovich equation describing conservation of particles in phase space [63]. This, along with Maxwell's equations constitute an exact description of a plasma. However, this contains far too much information about individual particles to be tractable, containing all information about particle orbits, while mapping a particle to some location in phase space, $(\boldsymbol{x},\boldsymbol{v})$. Given the collective nature of plasma behavior and interest in a more macro scale, the desired information is how many particles are likely to be found in a small volume of phase space $\Delta\boldsymbol{x}\Delta\boldsymbol{v}$. Thus instead of evolving the spikey function $N_\alpha(\boldsymbol{x},\boldsymbol{v},t)$ it is more advantageous to consider the smooth function

$$f_\alpha(\boldsymbol{x},\boldsymbol{v},t) \equiv \langle N_\alpha(\boldsymbol{x},\boldsymbol{v},t)\rangle\,, \tag{2.2.5}$$

where the $\langle\rangle$ is an ensemble average over an infinite number of realizations of the plasma [85]. Taking such an ensemble average over the entire Eq. (2.2.4) results in the Boltzmann equation describing the evolution of the probability distribution function f_α of species α in a plasma

$$\frac{\partial f_\alpha}{\partial t} + v_i\frac{\partial f_\alpha}{\partial x_i} + \frac{q_\alpha}{m_\alpha}\left(E_i + \epsilon_{ijk}v_jB_k\right)\frac{\partial f_\alpha}{\partial v_i} = \left.\frac{\partial f_\alpha}{\partial t}\right|_C. \tag{2.2.6}$$

The Boltzmann equation can also be arrived at by starting from the conservative form of the Klimontovich equation, giving

$$\frac{\partial f_\alpha}{\partial t} + \frac{\partial}{\partial x_i}\left(v_if_\alpha\right) + \frac{\partial}{\partial v_i}\left[\frac{q_\alpha}{m_\alpha}\left(E_i + \epsilon_{ijk}v_jB_k\right)f_\alpha\right] = \left.\frac{\partial f_\alpha}{\partial t}\right|_C. \tag{2.2.7}$$

Macroscopic variables can be calculated by taking velocity-space moments of the distribution function. Some of the most physical ones are found from lower moments. The 0th moment yields number density

$$n_\alpha(\boldsymbol{x},t) = \int f_\alpha(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}, \tag{2.2.8}$$

from which mass density $\rho_\alpha = n_\alpha m_\alpha$ and charge density $\rho_{c_\alpha} = n_\alpha q_\alpha$ are calculated. The first moment yields particle flux

$$\boldsymbol{\Gamma}_\alpha(\boldsymbol{x},t) = n_\alpha(\boldsymbol{x},t)\boldsymbol{v}_\alpha(\boldsymbol{x},t) = \int \boldsymbol{v} f_\alpha(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}. \tag{2.2.9}$$

The combination of these lowest moments yields velocity, $\boldsymbol{v}_\alpha$ as well as current density $\boldsymbol{j}_\alpha(\boldsymbol{x},t) = q_\alpha\boldsymbol{\Gamma}_\alpha(\boldsymbol{x},t)$. The 2nd moment can yield the pressure tensor

$$\overleftrightarrow{P}_\alpha(\boldsymbol{x},t) = m_\alpha \int \left(\boldsymbol{v}-\boldsymbol{v}_\alpha\right)\left(\boldsymbol{v}-\boldsymbol{v}_\alpha\right) f_\alpha(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}. \tag{2.2.10}$$

The scalar pressure can be found with a 2nd reduced moment

$$p_\alpha(\boldsymbol{x},t) = n_\alpha T_\alpha = \frac{m_\alpha}{d}\int \left(\boldsymbol{v}-\boldsymbol{v}_\alpha(\boldsymbol{x},t)\right)^2 f_\alpha(\boldsymbol{x},t)d\boldsymbol{v}, \tag{2.2.11}$$

where d is the number of velocity space dimensions. Heat flux is given by a reduced 3rd moment

$$\boldsymbol{h}_\alpha(\boldsymbol{x},\boldsymbol{v},t) = \frac{1}{2}m_\alpha \int \left(\boldsymbol{v}-\boldsymbol{v}_\alpha\right)\cdot\left(\boldsymbol{v}-\boldsymbol{v}_\alpha\right)\left(\boldsymbol{v}-\boldsymbol{v}_\alpha\right) f_\alpha(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}. \tag{2.2.12}$$

A simple representation for collisions is given by the Bhatnagar-Gross-Krook (BGK) operator [6], written as

$$\left.\frac{\partial f_\alpha}{\partial t}\right|_C = -\,\nu_\alpha\left(f_\alpha - f_{M_\alpha}\right), \tag{2.2.13}$$

for some relaxation frequency ν_α and where f_{M_α} is a Maxwellian distribution function given by

$$f_{M_\alpha}(\boldsymbol{x},\boldsymbol{v},t) = n_\alpha(\boldsymbol{x},t)\left(\frac{m_\alpha}{2\pi T_\alpha}\right)^{\frac{d}{2}} \exp\left(\frac{-m_\alpha(\boldsymbol{v}-\boldsymbol{v}_\alpha(\boldsymbol{x},t))\cdot(\boldsymbol{v}-\boldsymbol{v}_\alpha(\boldsymbol{x},t))}{2T_\alpha}\right), \tag{2.2.14}$$

which can be constructed from Eqs.(2.2.8), (2.2.9), and (2.2.11) applied on f_α. A locally dependent collision frequency is given by [78]

$$\nu_\alpha = \frac{q_\alpha^4 n_\alpha}{3\sqrt{2}\pi^{3/2}\epsilon_0^2 m_\alpha^{\frac{1}{2}} T_\alpha^{\frac{3}{2}}}\ln\Lambda. \tag{2.2.15}$$

2.2.2 Moment Models

Just as velocity-space moments yield macroscopic variables, velocity-space moments can also be taken of the Boltzmann equation, yielding conservation laws for these macroscopic variables. In principle, the amount of moments taken can be infinite, and each moment equation involves the next higher moment. In practice, one must stop at a particular moment and develop a closure scheme relating the highest moment left to known quantities.

5N-moment Model

In this model, the 0th order moment, three 1st order moments, and a reduced 2nd order moment are taken. Details of the derivation are given in Krall & Trivelpiece [63]. Multiplying the 0th moment equation by m_α

$$m_\alpha \int \left(\frac{\partial f_\alpha}{\partial t} + v_i \frac{\partial f_\alpha}{\partial x_i} + \frac{q_\alpha}{m_\alpha} \left(E_i + \epsilon_{ijk} v_j B_k \right) \frac{\partial f_\alpha}{\partial v_i} \right) d\boldsymbol{v} = m_\alpha \int \left. \frac{\partial f_\alpha}{\partial t} \right|_C d\boldsymbol{v}, \tag{2.2.16}$$

yields the continuity equation

$$\frac{\partial \rho_\alpha}{\partial t} + \frac{\partial \left(\rho v_{\alpha_i} \right)}{\partial x_i} = 0. \tag{2.2.17}$$

Three 1st order moments multiplied by m_α

$$m_\alpha \int v_l \left(\frac{\partial f_\alpha}{\partial t} + v_i \frac{\partial f_\alpha}{\partial x_i} + \frac{q_\alpha}{m_\alpha} \left(E_i + \epsilon_{ijk} v_j B_k \right) \frac{\partial f_\alpha}{\partial v_i} \right) d\boldsymbol{v} = m_\alpha \int v_l \left. \frac{\partial f_\alpha}{\partial t} \right|_C d\boldsymbol{v}, \tag{2.2.18}$$

yield the momentum equation

$$\frac{\partial \left(\rho_\alpha v_{\alpha_i} \right)}{\partial t} + \frac{\partial \left(\rho_\alpha v_{\alpha_i} v_{\alpha_j} + P_{\alpha_{ij}} \right)}{\partial x_j} = \frac{q_\alpha \rho_\alpha}{m_\alpha} \left(E_i + \epsilon_{ijk} v_{\alpha_j} B_k \right) + \sum_{\beta \neq \alpha} R_{\alpha \beta_i}, \tag{2.2.19}$$

where $R_{\alpha\beta_i}$ represents an interspecies momentum exchange between all species $\beta \neq \alpha$. The energy equation can be obtained using a reduced 2nd order moment, $\frac{1}{2} m_\alpha \left(\boldsymbol{v} \cdot \boldsymbol{v} \right)$

$$\frac{1}{2n} m_\alpha \int v_l v_l \left(\frac{\partial f_\alpha}{\partial t} + v_i \frac{\partial f_\alpha}{\partial x_i} + \frac{q_\alpha}{m_\alpha} \left(E_i + \epsilon_{ijk} v_j B_k \right) \frac{\partial f_\alpha}{\partial v_i} \right) d\boldsymbol{v} = \frac{1}{2} m_\alpha \int v_l v_l \left. \frac{\partial f_\alpha}{\partial t} \right|_C d\boldsymbol{v}, \tag{2.2.20}$$

leading to

$$\frac{\partial e_\alpha}{\partial t}+\frac{\partial\left(e_\alpha v_{\alpha_i}+P_{\alpha_{ij}}v_{\alpha_j}+h_{\alpha_i}\right)}{\partial x_i}=\frac{q_\alpha\rho_\alpha}{m_\alpha}v_{\alpha_i}E_i+\sum_{\beta\neq\alpha}v_{\alpha_i}R_{\alpha\beta_i}+\sum_{\beta\neq\alpha}Q_{\alpha\beta} \tag{2.2.21}$$

for $e_\alpha=\frac{p_\alpha}{\gamma-1}+\frac{1}{2}\rho_\alpha v_\alpha^2$ and where $Q_{\alpha\beta}$ represents an interspecies heat exchange between all species $\beta\neq\alpha$. A closure requirement remains to define intraspecies collision terms, which are the pressure tensor and heat flux. A Chapman-Enskog expansion of the distribution function about a Maxwellian can be performed. First, the pressure tensor is broken up into isotropic and anisotropic parts

$$P_{\alpha_{ij}}\equiv\frac{1}{3}p_{\alpha_{kk}}\delta_{ij}+\Pi_{\alpha_{ij}}, \tag{2.2.22}$$

where p is the the pressure taken from the trace of the full pressure tensor, also defined in Eq. (2.2.11). For a neutral fluid, the Chapman-Enskog procedure yields the following forms for the anisotropic pressure tensor and heat flux [124]

$$\Pi_{\alpha_{ij}}=\mu_\alpha\left(\frac{\partial v_{\alpha_i}}{\partial x_j}+\frac{\partial v_{\alpha_j}}{\partial x_i}-\frac{2}{3}\frac{\partial v_{\alpha_k}}{\partial x_k}\delta_{ij}\right), \tag{2.2.23}$$

$$h_{\alpha_i}=-K_\alpha\frac{\partial T_\alpha}{\partial x_i}. \tag{2.2.24}$$

For a magnetized plasma the procedure follows the derivation as shown by Braginskii [12]. Interspecies collisionality is derived by Hinton [56]

$$R_{\alpha\beta_i}=m_\alpha n_\alpha\left(v_{\beta_i}-v_{\alpha_i}\right)\nu_{\alpha\beta}, \tag{2.2.25}$$

$$Q_{\alpha\beta}=\frac{3n_\alpha m_\alpha\left(T_\beta-T_\alpha\right)}{m_\alpha+m_\beta}\nu_{\alpha\beta}+\frac{m_\beta}{m_\alpha+m_\beta}R_{\alpha\beta_i}\left(v_{\beta_i}-v_{\alpha_i}\right), \tag{2.2.26}$$

with collision frequency (see Appendix A)

$$\nu_{\alpha\beta}=\frac{1}{3\pi^{\frac{3}{2}}\epsilon_0^2}\frac{n_\beta Z_\beta^2 Z_\alpha^2 e^4\ln\Lambda\left(1+\frac{m_\alpha}{m_\beta}\right)}{m_\alpha^2\left(2\frac{T_\alpha}{m_\alpha}+2\frac{T_\beta}{m_\beta}\right)^{\frac{3}{2}}}. \tag{2.2.27}$$

The second term on the right-hand side of Eq. (2.2.26) ensures particles, momentum, and energy are conserved in collisions ($R_{\beta\alpha_i} = -R_{\alpha\beta_i},\ v_{\alpha_i} R_{\alpha\beta_i} + v_{\beta_i} R_{\beta\alpha_i} + Q_{\alpha\beta} + Q_{\beta\alpha} = 0$) and captures additional heating of light fast-moving particles colliding with heavy slow-moving particles [12, 79]. Electromagnetic sources are obtained by coupling with Maxwell's equations. Ampere's law yields a time evolution of electric field

$$-\frac{1}{c^2}\frac{\partial E_i}{\partial t} + \epsilon_{ijk}\frac{\partial B_k}{\partial x_j} = \mu_0 j_i = \mu_0 \sum_\alpha q_\alpha n_\alpha v_{\alpha_i}, \tag{2.2.28}$$

and Faraday's law yields a time evolution of magnetic field

$$\frac{\partial B_i}{\partial t} + \epsilon_{ijk}\frac{\partial E_k}{\partial x_j} = 0. \tag{2.2.29}$$

Center of Mass 5N-moment Model

The $5N$-moment model can be rewritten in an identical form in the center of mass frame. While the center of mass model is not used explicitly in this work, it represents a useful step in the reduction of the $5N$-moment model to the MHD model. To arrive at this model, define the center of mass density

$$\rho = \sum_\alpha m_\alpha n_\alpha \tag{2.2.30}$$

and center of mass velocity

$$v_i = \frac{\sum\limits_\alpha m_\alpha n_\alpha v_{\alpha_i}}{m_\alpha n_\alpha}, \tag{2.2.31}$$

as well as charge density

$$\rho_c = \sum_\alpha q_\alpha n_\alpha = \sum_\alpha q_\alpha \frac{\rho_\alpha}{m_\alpha}, \tag{2.2.32}$$

pressure

$$p = \sum_\alpha p_\alpha, \tag{2.2.33}$$

pressure tensor

$$\Pi_{ij} = \sum_{\alpha} \Pi_{\alpha_{ij}}, \tag{2.2.34}$$

and heat flux

$$h_i = \sum_{\alpha} h_{\alpha_i}. \tag{2.2.35}$$

Summing up each species' continuity equation yields a center of mass continuity equation

$$\frac{\partial \rho}{\partial t} + \frac{\partial \left(\rho v_i\right)}{\partial x_i} =0. \tag{2.2.36}$$

A center of mass momentum can be derived first by defining a relative species velocity in relation to the center of mass velocity

$$w_{\alpha_i} =v_{\alpha_i} - v_i \tag{2.2.37}$$

so that an evolution of species number density can be written

$$\frac{\partial n_\alpha}{\partial t} + \frac{\partial \left[n_\alpha \left(v_i + w_{\alpha_i}\right)\right]}{\partial x_i} =0. \tag{2.2.38}$$

Then a center of mass momentum equation can be written by summing each individual species momentum

$$\frac{\partial \left(\rho v_i\right)}{\partial t} + \frac{\partial \left(\rho v_i v_j + P_{ij} + \sum\limits_{\alpha} m_\alpha n_\alpha w_{\alpha_i} w_{\alpha_j}\right)}{\partial x_j} =\rho_c E_i + \epsilon_{ijk} j_j B_k + \sum_{\alpha} \sum_{\beta \neq \alpha} R_{\alpha\beta_i}. \tag{2.2.39}$$

Note $\sum\limits_{\alpha} \sum\limits_{\beta \neq \alpha} R_{\alpha\beta_i} = 0$. The evolution of w_{α_i} is still required to use this equation, which can be arrived at using Eq. (2.2.36) to write Eq. (2.2.39) as an evolution of center of mass velocity, using Eq. (2.2.17) to write Eq. (2.2.19) as an evolution of species velocity, and subtracting one result from the other. This yields

$$\begin{aligned}\frac{\partial w_{\alpha_i}}{\partial t} + v_j \frac{\partial w_{\alpha_i}}{\partial x_j} + w_{\alpha_j} \frac{\partial \left(v_i + w_{\alpha_i}\right)}{\partial x_j} - \frac{1}{\rho} \frac{\partial \left(\sum\limits_{\alpha} m_\alpha n_\alpha w_{\alpha_i} w_{\alpha_j}\right)}{\partial x_j} + \\ \frac{1}{m_\alpha n_\alpha} \frac{\partial P_{\alpha_{ij}}}{\partial x_j} - \frac{1}{\rho} \frac{\partial P_{ij}}{\partial x_j} = \left(\frac{q_\alpha}{m_\alpha} - \frac{\rho_c}{\rho}\right) E_i + \epsilon_{ijk} \left(\frac{q_\alpha v_{\alpha_j}}{m_\alpha} - \frac{j_j}{\rho}\right) B_k + \\ \frac{1}{m_\alpha n_\alpha} \sum_{\beta \neq \alpha} R_{\alpha\beta_i}.\end{aligned} \tag{2.2.40}$$

Finally, a center of mass energy equation is arrived at by adding up the individual species energy equations, yielding

$$\frac{\partial e}{\partial t}+\frac{\partial\left(ev_i+P_{ij}v_j+h_i+\sum\limits_{\alpha}\left(e_\alpha w_{\alpha_i}+P_{\alpha_{ij}}w_{\alpha_j}\right)\right)}{\partial x_i}=j_iE_i+\sum_{\alpha}\sum_{\beta\neq\alpha}v_{\alpha_i}R_{\alpha\beta_i}+\sum_{\alpha}\sum_{\beta\neq\alpha}Q_{\alpha\beta}, \tag{2.2.41}$$

where e_α can be evolved using Eq. (2.2.21), where Eq. (2.2.37) can be used to remove v_{α_i}. Assuming no chemical reactions, ionization, or recombination, $\sum\limits_{\alpha}\sum\limits_{\beta\neq\alpha}v_{\alpha_i}R_{\alpha\beta_i}+\sum\limits_{\alpha}\sum\limits_{\beta\neq\alpha}Q_{\alpha\beta}=0$. The model is completed again with Maxwell's Eqs. (2.2.28) and (2.2.29). The model is exactly equivalent to the non-center of mass $5N$-moment model, but written in a different frame of reference.

MHD Model

The MHD model can be derived from the center of mass $5N$-moment model by formally applying 2 asymptotic approximations:

1. Infinite speed of light, leading to charge neutrality

2. Negligible electron inertia

The first point removes the displacement current term in Amperes Law (equation 2.2.28) leading to

$$\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}=\mu_0 j_i. \tag{2.2.42}$$

Guass' law, written as

$$\epsilon_0\frac{\partial E_i}{\partial x_i}=\rho_c=\sum_{\alpha}q_\alpha n_\alpha \tag{2.2.43}$$

also turns into

$$\rho_c=0, \tag{2.2.44}$$

since $\epsilon_0 = \frac{1}{\mu_0 c^2}$ and thus the $\lim\limits_{c\to\infty} \epsilon_0 = 0$. This assumption leads to an adjustment to Eq. (2.2.39) representing center of mass momentum

$$\frac{\partial\left(\rho v_i\right)}{\partial t} + \frac{\partial\left(\rho v_i v_j + P_{ij} + \sum\limits_{\alpha} m_\alpha n_\alpha w_{\alpha_i} w_{\alpha_j}\right)}{\partial x_j} = \epsilon_{ijk}\left(\frac{1}{\mu_0}\epsilon_{jlm}\frac{\partial B_m}{\partial x_l}\right) B_k \tag{2.2.45}$$

as well as the drift velocity in Eq. (2.2.40)

$$\begin{aligned}\frac{\partial w_{\alpha_i}}{\partial t} + v_j \frac{\partial w_{\alpha_i}}{\partial x_j} + w_{\alpha_j}\frac{\partial\left(v_i + w_{\alpha_i}\right)}{\partial x_j} - \frac{1}{\rho}\frac{\partial\left(\sum\limits_{\alpha} m_\alpha n_\alpha w_{\alpha_i} w_{\alpha_j}\right)}{\partial x_j} + &\\ \frac{1}{m_\alpha n_\alpha}\frac{\partial P_{\alpha_{ij}}}{\partial x_j} - \frac{1}{\rho}\frac{\partial P_{ij}}{\partial x_j} = \left(\frac{q_\alpha}{m_\alpha}\right)\left(E_i + \epsilon_{ijk} v_{\alpha_j} B_k\right) - \frac{1}{\rho}\epsilon_{ijk}\left(\frac{1}{\mu_0}\epsilon_{jlm}\frac{\partial B_m}{\partial x_l}\right) B_k &\\ + \frac{1}{m_\alpha n_\alpha}\sum_{\beta\neq\alpha} R_{\alpha\beta_i} &\end{aligned} \tag{2.2.46}$$

and the energy Eq. (2.2.41)

$$\frac{\partial e}{\partial t} + \frac{\partial\left(e v_i + P_{ij} v_j + h_i + \sum\limits_{\alpha}\left(e_\alpha w_{\alpha_i} + P_{\alpha_{ij}} w_{\alpha_j}\right)\right)}{\partial x_i} = \left(\frac{1}{\mu_0}\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}\right) E_i. \tag{2.2.47}$$

The second approximation ($m_e \to 0$) can be applied separately and yields a momentum equation as

$$\frac{\partial\left(\rho v_i\right)}{\partial t} + \frac{\partial\left(\rho v_i v_j + P_{ij} + \sum\limits_{\substack{\alpha\\ \alpha\neq e}} m_\alpha n_\alpha w_{\alpha_i} w_{\alpha_j}\right)}{\partial x_j} = \rho_c E_i + \epsilon_{ijk} j_j B_k. \tag{2.2.48}$$

In the specific two-fluid case of ions and electrons, the center of mass velocity, as described in Eq. (2.2.31) implies $v_i = v_{\text{ions}_i}$, implying by Eq. (2.2.37) that $w_{\text{ions}_i} = 0$. This turns Eq. (2.2.48) into

$$\frac{\partial\left(\rho v_i\right)}{\partial t} + \frac{\partial\left(\rho v_i v_j + P_{ij}\right)}{\partial x_j} = \rho_c E_i + \epsilon_{ijk} j_j B_k. \tag{2.2.49}$$

The species drift velocity evolution Eq. (2.2.40) for the specific two-fluid case can also be written with these approximations. The ion equation disappears from the 0 condition on

w_{ions_i} while the electron drift equation turns into a state equation

$$-\frac{1}{n_e}\frac{\partial P_{\mathrm{e}_{ij}}}{\partial x_j} =e\left(E_i+\epsilon_{ijk}\left(v_j+w_{\mathrm{e}_j}\right)B_k\right)-\frac{1}{n_{\mathrm{e}}}R_{\mathrm{ei}_i}. \tag{2.2.50}$$

Also notice the interspecies momentum transfer term, R_{ei_i} can be written using Eq. (2.3.22) as

$$R_{\mathrm{ei}_i} =m_{\mathrm{e}}n_{\mathrm{e}}\left(v_{\mathrm{i}_i}-v_{\mathrm{e}_i}\right)\nu_{\mathrm{ei}}=m_e\frac{j_i}{q_{\mathrm{e}}}\nu_{\mathrm{ei}}=en_{\mathrm{e}}\eta j_i, \tag{2.2.51}$$

where

$$\eta \equiv\frac{m_e\nu_{\mathrm{ei}}}{n_{\mathrm{e}}q_{\mathrm{e}}}. \tag{2.2.52}$$

Finally, the center of mass energy Eq. (2.2.41) under these approximations can be written (noticing $e_{\mathrm{e}}=\frac{p_{\mathrm{e}}}{\gamma-1}$)

$$\frac{\partial e}{\partial t}+\frac{\partial\left(ev_i+P_{ij}v_j+h_i+\left(\frac{\gamma p_{\mathrm{e}}}{\gamma-1}w_{\mathrm{e}_i}+\Pi_{\mathrm{e}_{ij}}w_{\mathrm{e}_j}\right)\right)}{\partial x_i} =j_iE_i. \tag{2.2.53}$$

The MHD model then comes from combining these assumptions. It does not affect the continuity Eq. (2.2.36). Combining Eqs. (2.2.45) and (2.2.49) gives the momentum equation

$$\frac{\partial\left(\rho v_i\right)}{\partial t}+\frac{\partial\left(\rho v_iv_j+P_{ij}\right)}{\partial x_j} =\epsilon_{ijk}\left(\frac{1}{\mu_0}\epsilon_{jlm}\frac{\partial B_m}{\partial x_l}\right)B_k. \tag{2.2.54}$$

Combining the assumptions of Eqs. (2.2.46) and (2.2.50) simply leads back to Eq. (2.2.50), but also leads to the expression for electron drift velocity

$$w_{\mathrm{e}_i} =-\frac{j_i}{n_{\mathrm{e}}q_{\mathrm{e}}}=-\frac{\frac{1}{\mu_0}\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}}{n_{\mathrm{e}}q_{\mathrm{e}}}. \tag{2.2.55}$$

Applying this to Eq. (2.2.50) yields the generalized Ohm's law

$$E_i =-\epsilon_{ijk}v_jB_k+\frac{1}{n_{\mathrm{e}}e}\left(\epsilon_{ijk}\left(\frac{1}{\mu_0}\epsilon_{jlm}\frac{\partial B_m}{\partial x_l}\right)B_k-\frac{\partial P_{\mathrm{e}_{ij}}}{\partial x_j}+R_{\mathrm{ei}_i}\right). \tag{2.2.56}$$

For the energy equation, Eqs. (2.2.47) and (2.2.53) are combined

$$\frac{\partial e}{\partial t}+\frac{\partial\left(ev_i+P_{ij}v_j+h_i+\left(\frac{\gamma p_{\mathrm{e}}}{\gamma-1}w_{\mathrm{e}_i}+\Pi_{\mathrm{e}_{ij}}w_{\mathrm{e}_j}\right)\right)}{\partial x_i} =\left(\frac{1}{\mu_0}\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}\right)E_i. \tag{2.2.57}$$

Equations (2.2.36), (2.2.54), (2.2.56), (2.2.57) and (2.2.29) are the equations of Hall MHD. However, MHD energy often represents the sum of plasma energy e and magnetic field energy, $\frac{B^2}{2\mu_0}$. This gives a different form of the energy equation. To do this, the magnetic field energy evolution equation is first obtained by dotting Faraday's Law in Eq. (2.2.29) with B_i,

$$\frac{\partial B_i}{\partial t} B_i = - \epsilon_{ijk} \frac{\partial E_k}{\partial x_j} B_i. \tag{2.2.58}$$

Some manipulation turns this into

$$\frac{\partial}{\partial t}\left(\frac{B^2}{2}\right) = - \epsilon_{ijk} \frac{\partial B_k}{\partial x_j} E_i - \frac{\partial}{\partial x_i}\left(\epsilon_{ijk} E_j B_k\right). \tag{2.2.59}$$

The first term on the right hand side can be substituted into Eq. (2.2.57) leaving the time derivative term and the divergence term, seen to be the Poynting flux. Making this substitution using Ohm's Law in Eq. (2.2.56) and defining total energy as $e_t = e + \frac{B^2}{2\mu_0}$ leads to

$$\begin{aligned}\frac{\partial e_t}{\partial t} + \frac{\partial\left(\left(e_t + p + \frac{B^2}{2\mu_0}\right) v_i + \Pi_{ij} v_j + h_i - \frac{v_j B_j B_i}{\mu_0} + \left(e_{\mathrm{e}_t} + p_{\mathrm{e}} + \frac{B^2}{2\mu_0}\right) w_{\mathrm{e}_i} + \Pi_{\mathrm{e}_{ij}} w_{\mathrm{e}_j} - \frac{w_{\mathrm{e}_j} B_j B_i}{\mu_0}\right)}{\partial x_i} \\ = \frac{1}{\mu_0 n_{\mathrm{e}} e} \frac{\partial\left(\epsilon_{ijk} \frac{\partial P_{\mathrm{e}_{jl}}}{\partial x_l} B_k - \epsilon_{ijk} R_{\mathrm{ei}_j} B_k\right)}{\partial x_i},\end{aligned} \tag{2.2.60}$$

where w_{e_i} is defined in Eq. (2.2.55).

2.3 Normalization Method

Normalization of an equation set can provide a convenient means for evaluating the relative importance of terms or the effect of a given set of physical parameters. From a numerical perspective, non-dimensionalization can also make a system less stiff and provide ease in convergence toward a solution [109]. For a particular equation set, a normalization scheme requires choosing characteristic values, which are not unique. However, in this work it is desired to choose a consistent normalization scheme using non-dimensional parameters valid across all models. This is also critical for the domain-hybridized plasma model work.

With this in mind, a normalization is chosen that is preserved through asymptotic approximations of the center of mass single fluid MHD model. Specifically, this normalization naturally converts a normalized Ampere's Law to its pre-Maxwellian form when these approximations are made to arrive at the MHD model. Take the Maxwellian form of this law in SI units, as shown in Eq. (2.2.28). It is apparent that the approximation $c \to \infty$ removes the displacement current term. The normalization scheme should maintain this property. To do this, first write all dimensional terms as products of reference values denoted with subscripts 0 and normalized values denoted by superscripted $\sim$, i.e. $E_i = E_0 \tilde{E}_i$. Then if one uses the definitions $\omega_p^2 = \frac{q_0^2 n_0}{m_0 \epsilon_0}$, $\omega_c = \frac{q_0 B_0}{m_0}$ and $c^2 = \frac{1}{\mu_0 \epsilon_0}$ and applies the following normalization relationships, $v_0 = \frac{L}{\tau} = \sqrt{\frac{T_0}{m_0}}$, $\beta_0 = \frac{n_0 T_0}{\frac{B_0^2}{2\mu_0}} = 2$, $E_0 = v_0 B_0$, and $j_0 = q_0 n_0 v_0$, a normalized form of Eq. (2.2.28) can be written as

$$-\frac{(\omega_c \tau)^2}{(\omega_p \tau)^2} \frac{\partial \tilde{E}_i}{\partial \tilde{t}} + \epsilon_{ijk} \frac{\partial \tilde{B}_k}{\partial \tilde{x}_j} = (\omega_c \tau)\, \tilde{j}_i. \tag{2.3.1}$$

Since the normalization aims to be generalized for problems that do not necessarily involve a magnetic field, the definition of a reference skin depth, given by $\delta_p = \frac{c}{\omega_p}$ can be used to show that $\omega_c \tau = \frac{L}{\delta_p}$, yielding an alternate form of Eq. (2.3.1)

$$-\frac{1}{(\omega_p \tau)^2} \left(\frac{L}{\delta_p}\right)^2 \frac{\partial \tilde{E}_i}{\partial \tilde{t}} + \epsilon_{ijk} \frac{\partial \tilde{B}_k}{\partial \tilde{x}_j} = \left(\frac{L}{\delta_p}\right) \tilde{j}_i. \tag{2.3.2}$$

It can be seen here that since μ_0 is tied to a finite reference v_0 through β_0, $\epsilon_0 \to 0$ as $c \to \infty$ and consequently $\omega_p \to \infty$, removing the displacement current term, as needed. The same thing happens with Gauss law, as shown in Eq. (2.2.43) which becomes after normalization

$$\frac{1}{(\omega_p \tau)^2} \left(\frac{L}{\delta_p}\right) \frac{\partial \tilde{E}_i}{\partial \tilde{x}_i} = \tilde{\rho}_c. \tag{2.3.3}$$

Again in both cases for the same reasons, the approach of $c \to \infty$ implies a lack of charge density in both the dimensional and non-dimensional forms, as required by MHD. The same procedure can be applied to Faraday's law in Eq. (2.2.29), which upon normalizing with the same reference values yields

$$\frac{\partial \tilde{B}_i}{\partial \tilde{t}} + \epsilon_{ijk} \frac{\partial \tilde{E}_k}{\partial \tilde{x}_j} = 0. \tag{2.3.4}$$

Rounding out Maxwell's equations, magnetic Gauss' law

$$\frac{\partial B_i}{\partial x_i} = 0 \tag{2.3.5}$$

simply becomes

$$\frac{\partial \tilde{B}_i}{\partial \tilde{x}_i} = 0. \tag{2.3.6}$$

2.3.1 Models written in normalized form

Here the equations as described in Sec. 2.2 are given with the normalization described in Sec. 2.3. All terms are considered normalized with tildes dropped for readability. The Boltzmann equation in non-conservative form is

Boltzmann-Maxwell Continuum Kinetic Model

$$\frac{\partial f_\alpha}{\partial t} + v_i \frac{\partial f_\alpha}{\partial x_i} + \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha}{A_\alpha} \left(E_i + \epsilon_{ijk} v_j B_k\right) \frac{\partial f_\alpha}{\partial v_i} = \left.\frac{\partial f_\alpha}{\partial t}\right|_C . \tag{2.3.7}$$

In conservative form, it is written

$$\frac{\partial f_\alpha}{\partial t} + \frac{\partial}{\partial x_i}\left(v_i f_\alpha\right) + \frac{\partial}{\partial v_i}\left[\left(\frac{L}{\delta_p}\right) \frac{Z_\alpha}{A_\alpha} \left(E_i + \epsilon_{ijk} v_j B_k\right) f_\alpha\right] = \left.\frac{\partial f_\alpha}{\partial t}\right|_C . \tag{2.3.8}$$

The current density is given by

$$j_i = \sum_\alpha Z_\alpha \Gamma_{\alpha_i}(\boldsymbol{x}, t). \tag{2.3.9}$$

Ampere's law is

$$-\frac{1}{\left(\omega_p \tau\right)^2} \left(\frac{L}{\delta_p}\right)^2 \frac{\partial E_i}{\partial t} + \epsilon_{ijk} \frac{\partial B_k}{\partial x_j} = \left(\frac{L}{\delta_p}\right) j_i \tag{2.3.10}$$

and Faraday's law is

$$\frac{\partial B_i}{\partial t} + \epsilon_{ijk} \frac{\partial E_k}{\partial x_j} = 0. \tag{2.3.11}$$

Other normalized equations become

$$\overleftrightarrow{P}_{\alpha}(\boldsymbol{x},t) = A_{\alpha}\int\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}\right)\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}\right)f_{\alpha}(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}, \tag{2.3.12}$$

$$p_{\alpha}(\boldsymbol{x},t) = n_{\alpha}T_{\alpha} = \frac{A_{\alpha}}{d}\int\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}(\boldsymbol{x},t)\right)^{2}f_{\alpha}(\boldsymbol{x},t)d\boldsymbol{v}, \tag{2.3.13}$$

$$\boldsymbol{h}_{\alpha}(\boldsymbol{x},\boldsymbol{v},t) = \frac{1}{2}A_{\alpha}\int\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}\right)\cdot\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}\right)\left(\boldsymbol{v}-\boldsymbol{v}_{\alpha}\right)f_{\alpha}(\boldsymbol{x},\boldsymbol{v},t)d\boldsymbol{v}, \tag{2.3.14}$$

$$\left.\frac{\partial f_{\alpha}}{\partial t}\right|_{C} = -\left(\nu_{p}\tau\right)\nu_{\alpha}\left(f_{\alpha}-f_{M_{\alpha}}\right), \tag{2.3.15}$$

$$\nu_{\alpha} = \frac{n_{\alpha}}{A_{\alpha}^{\frac{1}{2}}T_{\alpha}^{\frac{3}{2}}}\ln\Lambda, \tag{2.3.16}$$

$$f_{M_{\alpha}}(\boldsymbol{x},\boldsymbol{v},t) = n_{\alpha}(\boldsymbol{x},t)\left(\frac{A_{\alpha}}{2\pi T_{\alpha}}\right)^{\frac{d}{2}}\exp\left(\frac{-A_{\alpha}(\boldsymbol{v}-\boldsymbol{v}_{\alpha}(\boldsymbol{x},t))\cdot(\boldsymbol{v}-\boldsymbol{v}_{\alpha}(\boldsymbol{x},t))}{2T_{\alpha}}\right). \tag{2.3.17}$$

5N-moment Model

These are the equations of continuity

$$\frac{\partial\rho_{\alpha}}{\partial t}+\frac{\partial\left(\rho v_{\alpha_i}\right)}{\partial x_i} = 0, \tag{2.3.18}$$

momentum

$$\frac{\partial\left(\rho_{\alpha}v_{\alpha_i}\right)}{\partial t}+\frac{\partial\left(\rho_{\alpha}v_{\alpha_i}v_{\alpha_j}+P_{\alpha_{ij}}\right)}{\partial x_j} = \left(\frac{L}{\delta_p}\right)\frac{Z_{\alpha}}{A_{\alpha}}\rho_{\alpha}\left(E_i+\epsilon_{ijk}v_{\alpha_j}B_k\right)+\sum_{\beta\neq\alpha}R_{\alpha\beta_i}, \tag{2.3.19}$$

and energy

$$\frac{\partial e_{\alpha}}{\partial t}+\frac{\partial\left(e_{\alpha}v_{\alpha_i}+P_{\alpha_{ij}}v_{\alpha_j}+h_{\alpha_i}\right)}{\partial x_i} = \left(\frac{L}{\delta_p}\right)\frac{Z_{\alpha}}{A_{\alpha}}\rho_{\alpha}v_{\alpha_i}E_i+\sum_{\beta\neq\alpha}v_{\alpha_i}R_{\alpha\beta_i}+\sum_{\beta\neq\alpha}Q_{\alpha\beta}. \tag{2.3.20}$$

These are coupled with Eqs. (2.3.10) and (2.3.11), where the current density is given by

$$j_i = \sum_{\alpha}\frac{Z_{\alpha}}{A_{\alpha}}\rho_{\alpha}v_{\alpha_i}. \tag{2.3.21}$$

The normalized momentum transfer term from Eq. (2.3.22) becomes

$$R_{\alpha\beta_i} = \rho_\alpha \left(v_{\beta_i} - v_{\alpha_i}\right) \nu_{\alpha\beta}. \tag{2.3.22}$$

The normalized collision frequency becomes derived from Eq. (2.2.27) becomes (see Appendix A)

$$\nu_{\alpha\beta} = \left(\nu_p \tau\right) 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln \Lambda \frac{A_\alpha + A_\beta}{A_\alpha^2 A_\beta^2} \frac{\rho_\beta}{\left(\frac{T_\alpha}{A_\alpha} + \frac{T_\beta}{A_\beta}\right)^{3/2}}. \tag{2.3.23}$$

The normalized heat exchange term from Eq. (2.2.26) is

$$Q_{\alpha\beta} = \frac{3\rho_\alpha \left(T_\beta - T_\alpha\right)}{A_\alpha + A_\beta} \nu_{\alpha\beta} + \frac{A_\beta}{A_\alpha + A_\beta} R_{\alpha\beta_i} \left(v_{\beta_i} - v_{\alpha_i}\right). \tag{2.3.24}$$

Hall MHD

The continuity equation is just Eq. (2.2.36) written in normalized units

$$\frac{\partial \rho}{\partial t} + \frac{\partial \left(\rho v_i\right)}{\partial x_i} = 0. \tag{2.3.25}$$

The momentum Eq. (2.2.54) in normalized form becomes

$$\frac{\partial \left(\rho v_i\right)}{\partial t} + \frac{\partial \left(\rho v_i v_j + P_{ij}\right)}{\partial x_j} = \epsilon_{ijk} \left(\epsilon_{jlm} \frac{\partial B_m}{\partial x_l}\right) B_k.$$

Upon some manipulation the right hand side, source can be brought into the flux term (assuming $\frac{\partial B_i}{\partial x_i} = 0$), yielding

$$\frac{\partial \left(\rho v_i\right)}{\partial t} + \frac{\partial \left[\rho v_i v_j + P_{ij} - \left(B_i B_j - \frac{1}{2} B_k B_k \delta_{ij}\right)\right]}{\partial x_j} = 0. \tag{2.3.26}$$

Ohm's law in Eq. (2.2.56) normalized becomes

$$E_i = -\epsilon_{ijk} v_j B_k + \frac{1}{n_\mathrm{e}} \left(\frac{\delta_p}{L}\right) \left[\epsilon_{ijk} \left(\epsilon_{jlm} \frac{\partial B_m}{\partial x_l}\right) B_k - \frac{\partial P_{\mathrm{e}_{ij}}}{\partial x_j}\right] + \left(\frac{\delta_p}{L}\right) \left(\nu_p \tau\right) \eta j_i,$$

where Eq. (A.2.2) is used for the resistive term. This can now be substituted into the energy and induction equations. First, note the normalized form of Eq. (2.2.55)

$$w_{\mathrm{e}_i} = -\left(\frac{\delta_p}{L}\right) \frac{\epsilon_{ijk} \frac{\partial B_k}{\partial x_j}}{n_\mathrm{e}} = -\frac{j_i}{n_\mathrm{e}}, \tag{2.3.27}$$

where the current density comes from the high frequency version of Eq. (2.3.10) and n_{e} is derived in Appendix B (Eq. (B.1.9)). Also normalized resistivity is given by Eq. (A.2.1). This yields an energy equation

$$\frac{\partial e_t}{\partial t}+\frac{\partial\left(\left(e_t+p+\frac{B^2}{2}\right)v_i-v_jB_jB_i+\Pi_{ij}v_j+h_i+\left(e_{\mathrm{e}_t}+p_{\mathrm{e}}+\frac{B^2}{2}\right)w_{\mathrm{e}_i}+\Pi_{\mathrm{e}_{ij}}w_{\mathrm{e}_j}-w_{\mathrm{e}_j}B_jB_i\right)}{\partial x_i}$$
$$=\frac{\partial\left(\left(\frac{\delta_p}{L}\right)\epsilon_{ijk}\frac{1}{n_{\mathrm{e}}}\frac{\partial P_{\mathrm{e}_{jl}}}{\partial x_l}B_k-\left(\frac{\delta_p}{L}\right)\epsilon_{ijk}\left(\nu_p\tau\right)\eta j_jB_k\right)}{\partial x_i},\tag{2.3.28}$$

where resistive terms are highlighted in red, Hall terms are highlighted in blue, and viscous terms are highlighted in brown. Finally, the normalized Ohm's Law can be applied to the normalized Faraday's Law in Eq. (2.3.11)

$$\frac{\partial B_i}{\partial t}+\frac{\partial}{\partial x_j}\left[B_iv_j-v_iB_j+\epsilon_{ijk}\left\{\frac{1}{n_{\mathrm{e}}}\left(\frac{\delta_p}{L}\right)\left[\epsilon_{klm}\left(\epsilon_{lnq}\frac{\partial B_q}{\partial x_n}\right)B_m-\frac{\partial P_{\mathrm{e}_{kl}}}{\partial x_l}\right]\right\}\right.$$
$$\left.+\epsilon_{ijk}\left(\frac{\delta_p}{L}\right)\left(\nu_p\tau\right)\eta j_k\right]=0,$$

or taking into account high-frequency Eq. (2.3.10) as well as Eq. (2.3.27), this can be written

$$\frac{\partial B_i}{\partial t}+\frac{\partial}{\partial x_j}\left[B_iv_j-v_iB_j+\epsilon_{ijk}\left\{\left[-\epsilon_{klm}w_{\mathrm{e}_l}B_m-\frac{1}{n_{\mathrm{e}}}\left(\frac{\delta_p}{L}\right)\frac{\partial P_{\mathrm{e}_{kl}}}{\partial x_l}\right]\right\}+\epsilon_{ijk}\left(\frac{\delta_p}{L}\right)\left(\nu_p\tau\right)\eta j_k\right]=0.\tag{2.3.29}$$

2.4 Hybrid Modeling

As described in the introduction, the goal of this work is to develop a framework that allows for the combination of each of the models described into a hybrid simulation, where different models are used in different domains, as dictated by the physics and ease of computation. The consistent normalization as described in the previous section is pivotal for allowing direct translation between the equation sets at model interfaces. With this in place, the critical part of the hybrid model is the development of appropriate boundary conditions between models. In this work, these boundary conditions are to be developed for the case of coupling between

1. 5N-moment model and the Hall MHD model
2. Kinetic model and the 5N-moment model

The coupling between models is being explored based on

- "Consistent" direct variable translation
- "Conservative" numerical flux matching

2.4.1 Direct Variable Translation

In this method, variables between models are matched in a consistent manner between the models.

5N-moment-MHD Coupling

The conversion between the 5N-moment center of mass (com) fluid variables can be written as

$$\rho_{\mathrm{com}} = \rho_{\mathrm{i}} + \rho_{\mathrm{e}}, \tag{2.4.1}$$

$$\left(\rho v_i\right)_{\mathrm{com}} = \left(\rho v_i\right)_{\mathrm{i}} + \left(\rho v_i\right)_{\mathrm{e}}, \tag{2.4.2}$$

$$\left(p\right)_{\mathrm{com}} = \left(p\right)_{\mathrm{i}} + \left(p\right)_{\mathrm{e}}, \tag{2.4.3}$$

$$\left(\Pi_{ij}\right)_{\mathrm{com}} = \left(\Pi_{ij}\right)_{\mathrm{i}} + \left(\Pi_{ij}\right)_{\mathrm{e}}, \tag{2.4.4}$$

$$\left(P_{ij}\right)_{\mathrm{com}} = \left(P_{ij}\right)_{\mathrm{i}} + \left(P_{ij}\right)_{\mathrm{e}}, \tag{2.4.5}$$

$$\left(j_i\right)_{\mathrm{com}} = \frac{Z_{\mathrm{i}}}{A_{\mathrm{i}}}\left(\rho v_i\right)_{\mathrm{i}} + \frac{Z_{\mathrm{e}}}{A_{\mathrm{e}}}\left(\rho v_i\right)_{\mathrm{e}} = -\frac{\left(\frac{L}{\delta_p}\right)}{\left(\omega_p \tau\right)^2}\frac{\partial E_i}{\partial t} + \left(\frac{\delta_p}{L}\right)\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}, \tag{2.4.6}$$

$$(e)_{\mathrm{com}} = \frac{p_{\mathrm{i}}}{\gamma - 1} + \frac{1}{2}\rho_{\mathrm{i}} {v_{i}}_{\mathrm{i}}^{2} + \frac{p_{\mathrm{e}}}{\gamma - 1} + \frac{1}{2}\rho_{\mathrm{e}} {v_{i}}_{\mathrm{e}}^{2}. \tag{2.4.7}$$

From here, the asymptotic approximations described in Sec. 2.2.2 can be applied, leading to

$$\rho_{\mathrm{mhd}} = \rho_{\mathrm{i}}, \tag{2.4.8}$$

$$(\rho v_i)_{\mathrm{mhd}} = (\rho v_i)_{\mathrm{i}}\,, \tag{2.4.9}$$

$$v_{i_{\mathrm{mhd}}} = \frac{(\rho v_i)_{\mathrm{mhd}}}{\rho_{\mathrm{mhd}}}, \tag{2.4.10}$$

$$(j_i)_{\mathrm{mhd}} = \frac{Z_{\mathrm{i}}}{A_{\mathrm{i}}}(\rho v_i)_{\mathrm{mhd}} + \frac{Z_{\mathrm{e}}}{A_{\mathrm{e}}}(\rho v_i)_{\mathrm{e}} = \left(\frac{\delta_p}{L}\right)\epsilon_{ijk}\frac{\partial B_k}{\partial x_j}, \tag{2.4.11}$$

$$(p)_{\mathrm{mhd}} = (p)_{\mathrm{com}}\,, \tag{2.4.12}$$

$$(\Pi_{ij})_{\mathrm{mhd}} = (\Pi_{ij})_{\mathrm{com}}\,, \tag{2.4.13}$$

$$(P_{ij})_{\mathrm{mhd}} = (P_{ij})_{\mathrm{com}}\,, \tag{2.4.14}$$

and incorporating magnetic field energy into the MHD energy definition,

$$e_{\mathrm{mhd}} = \frac{(p)_{\mathrm{mhd}}}{\gamma - 1} + \frac{1}{2}\rho_{\mathrm{mhd}} v_{\mathrm{mhd}}^{2} + \frac{1}{2}B^{2}. \tag{2.4.15}$$

Equations (2.4.8), (2.4.9), and (2.4.15) enable the consistency conversion of $5N$-moment variables to MHD. Going the other way, first note that by quasineutrality

$$Z_{\mathrm{i}} n_{\mathrm{i}} + Z_{\mathrm{e}} n_{\mathrm{e}} = 0. \tag{2.4.16}$$

Noting $\rho_\alpha = A_\alpha n_\alpha$,

$$\rho_{\mathrm{e}} = -\frac{Z_{\mathrm{i}}}{Z_{\mathrm{e}}}\frac{A_{\mathrm{e}}}{A_{\mathrm{i}}}\rho_{\mathrm{i}}. \tag{2.4.17}$$

Now incorporating Eq. (2.4.8) leads to

$$\rho_{\mathrm{e}} = -\frac{Z_{\mathrm{i}}}{Z_{\mathrm{e}}}\frac{A_{\mathrm{e}}}{A_{\mathrm{i}}}\rho_{\mathrm{mhd}}. \tag{2.4.18}$$

Next, from Eq. (2.4.11)

$$(\rho v_i)_{\mathrm{e}} = \frac{A_{\mathrm{e}}}{Z_{\mathrm{e}}}(j_i)_{\mathrm{mhd}} - \frac{A_{\mathrm{e}}}{A_{\mathrm{i}}}\frac{Z_{\mathrm{i}}}{Z_{\mathrm{e}}}(\rho v_i)_{\mathrm{mhd}}. \tag{2.4.19}$$

To calculate energy, first note

$$(p)_{\mathrm{mhd}} = (\gamma-1)\left[(e)_{\mathrm{mhd}} - \frac{1}{2}\rho_{\mathrm{mhd}}v_{\mathrm{mhd}}^2 - \frac{1}{2}B^2\right] = (p)_{\mathrm{i}} + (p)_{\mathrm{e}} = (n_{\mathrm{i}} + n_{\mathrm{e}})\,T_{\mathrm{mhd}} \tag{2.4.20}$$

assuming for MHD $T_{\mathrm{i}} = T_{\mathrm{e}} \equiv T_{\mathrm{mhd}}$. So

$$(T)_{\mathrm{mhd}} = \frac{(p)_{\mathrm{mhd}}}{n_{\mathrm{i}} + n_{\mathrm{e}}} = \frac{(p)_{\mathrm{mhd}}}{\frac{\rho_{\mathrm{i}}}{A_{\mathrm{i}}} + \frac{\rho_{\mathrm{e}}}{A_e}} \tag{2.4.21}$$

and therefore

$$(e)_{\mathrm{i}} = \frac{\frac{\rho_{\mathrm{i}}}{A_{\mathrm{i}}}(T)_{\mathrm{mhd}}}{\gamma-1} + \frac{1}{2}\frac{(\rho v_i)_{\mathrm{i}}^2}{\rho_{\mathrm{i}}} \tag{2.4.22}$$

and

$$(e)_{\mathrm{e}} = \frac{\frac{\rho_{\mathrm{e}}}{A_{\mathrm{e}}}(T)_{\mathrm{mhd}}}{\gamma-1} + \frac{1}{2}\frac{(\rho v_i)_{\mathrm{e}}^2}{\rho_{\mathrm{e}}}. \tag{2.4.23}$$

Thus, Eqs. (2.4.8), (2.4.9), and (2.4.22) are used to determine the $5N$-moment ion variables from MHD and Eqs. (2.4.18), (2.4.19), and (2.4.23) determine electron variables.

Kinetic-5N-moment Coupling

Consistency coupling between the kinetic and $5N$-moment models assumes a Maxwellian distribution function at the interface between models on different subdomains. This allows for a direct construction of a Maxwellian distribution function on the kinetic side from fluid variables on the fluid side and for moments to be taken of the distribution function on the kinetic side that become the fluid variables on the fluid side. The moments as given in Eqs. (2.2.8) and (2.2.9), give density and momentum, respectively, while a reduced 2nd moment, Eq. (2.3.13) determines pressure. Going the other way, the fluid variables $(n_\alpha, \boldsymbol{v}_\alpha, p_\alpha)$ are used to construct the Maxwellian using Eq. (2.3.17).

2.4.2 Conservative Flux Matching

The consistent approach, while matching variables, may not necessarily maintain the correct flux between the models. This effect is more pronounced when the assumptions associated with the lower fidelity model (lack of high frequency dynamics, negligible electron mass, single temperature for MHD, or non-Maxwellian-ness for $5N$-moment) are less valid. Thus, an alternative boundary condition is to apply an equal and opposite flux between the models.

5N Moment-MHD Coupling

Going through each equation of the MHD model, the flux as calculated from the $5N$-moment model is applied. For the continuity equation

$$\left(F_{i(\rho)}n_i\right)^*_{\mathrm{mhd}} = \left(F_{i(\rho)}n_i\right)^*_{\mathrm{i}} + \left(F_{i(\rho)}n_i\right)^*_{\mathrm{e}} = \left([\rho v_i]_{\mathrm{i}}\, n_i\right)^* + \left([\rho v_i]_{\mathrm{e}}\, n_i\right)^*, \tag{2.4.24}$$

where the asterisk signifies the numerical (surface) flux between finite elements and n signifies the normal vector between them. This formulation is elaborated upon in Chapter 3. For the momentum equation

$$\left(F_{ij\,(\rho\boldsymbol{v})}n_j\right)^*_{\mathrm{mhd}} = \left(F_{ij\,(\rho\boldsymbol{v})}n_j\right)^*_{\mathrm{i}} + \left(F_{ij\,(\rho\boldsymbol{v})}n_j\right)^*_{\mathrm{e}} = \left([\rho v_i v_j + p\delta_{ij}]_{\mathrm{i}}\, n_j\right)^* + \left([\rho v_i v_j + p\delta_{ij}]_{\mathrm{e}}\, n_j\right)^*. \tag{2.4.25}$$

For the energy equation

$$\begin{aligned}\left(F_{i(e)}n_i\right)^*_{\mathrm{mhd}} &= \left(F_{i(e)}n_i\right)^*_{\mathrm{i}} + \left(F_{i(e)}n_i\right)^*_{\mathrm{e}} + \left(F_{i\left(\frac{B^2}{2}\right)}n_i\right)^* \\ &= \left([(e+p)\, v_i]_{\mathrm{i}}\, n_i\right)^* + \left([(e+p)\, v_i]_{\mathrm{e}}\, n_i\right)^* + \left(\hat{E}_{ij}n_j\right)^* B_i,\end{aligned} \tag{2.4.26}$$

where the flux associated with the magnetic field energy in Eq. (2.2.58), inherent in the MHD energy, is incorporated using the matrix form of the curl operator, $\hat{E}_{ij} = \epsilon_{ijk}E_k$. Finally, for the induction equation,

$$\left(F_{ij\,(\boldsymbol{B})}n_j\right)^*_{\mathrm{mhd}} = \left(F_{ij\,(\boldsymbol{B})}n_j\right)^* = \left(\hat{E}_{ij}n_j\right)^*. \tag{2.4.27}$$

Kinetic-5N-moment Coupling

In this case, a flux is calculated from appropriate moments of the distribution function at the interface between the continuum kinetic and 5N-moment models that are based on the common composite distribution function. These moments correspond to the flux terms when taking the moments of the Boltzmann equation leading to the 5N-moment equations. For continuity

$$\left(F_{i(\rho)} n_i\right)^*_\alpha = \left(A_\alpha n_\alpha v_{\alpha i} n_i\right)^* = n_i \int v_i \left(A_\alpha f_\alpha\right) d\boldsymbol{v}, \tag{2.4.28}$$

for momentum

$$\left(F_{ij\,(\rho\boldsymbol{v})} n_j\right)^*_\alpha = \left(\left[A_\alpha n_\alpha v_{\alpha i} v_{\alpha j} + P_{\alpha_{ij}}\right] n_j\right)^* = n_j \int v_j \left(A_\alpha v_i f_\alpha\right) d\boldsymbol{v}, \tag{2.4.29}$$

and for energy

$$\left(F_{i(e)} n_i\right)^*_\alpha = \left(\left[e_\alpha v_{\alpha i} + P_{\alpha ij} v_{\alpha j} + h_{\alpha i}\right] n_i\right)^* = n_i \int v_i \left(\frac{1}{2} A_\alpha v_j v_j f_\alpha\right) d\boldsymbol{v}. \tag{2.4.30}$$

The calculation of the moments which are integrals in velocity space must be performed so as to include the effect of the near-Maxwellian distribution on the 5N-moment fluid model as well the arbitrary distribution existing on the continuum kinetic side of the interface. This should be performed in a manner that incorporates the effect of both models on each other through the common flux at the interface and is performed by summing the contribution of the moments by each distribution function at the appropriate velocities. The method of implementation is elaborated upon further in Chapter 3.

Chapter 3

NUMERICAL METHOD

The numerical scheme applied in this work is that of discontinuous finite elements, specifically the discontinuous Galerkin method. This is method is well-developed to handle conservation laws of the following form, especially when dominated by wave behavior,

$$\frac{\partial q_i}{\partial t} + \frac{\partial \mathcal{F}_{ij}}{\partial x_j} = S_i + \frac{\partial}{\partial x_j}\left(D_{ijkl}\frac{\partial q_l}{\partial x_k}\right). \tag{3.0.1}$$

The method was first introduced by Reed and Hill [100] in an investigation of the neutron transport equation and was later put on a solid theoretical footing in a series of papers by Cockburn and Shu [30, 29, 28, 26, 32]. It is a scheme that combines elements from the finite volume and classical continuous Galerkin finite element methods. Specifically, it breaks up a solution of a partial differential equation system on a spatial domain, such as given in Eq. (3.0.1), into a number of elements, upon which the solution is projected onto a set of basis functions. The distinguishing feature from the continuous Galerkin method is that C^0 continuity is not enforced between elements, that is, the solution can be discontinuous across element interfaces. The distinguishing feature from the finite volume method is that the projection of the solution can be onto a basis other than 1, or an average value. This combination allows for a compact method amenable to unstructured meshes that can also handle shocks and large discontinuities in the solution. The method also results in computational locality, which allows for high parallelizability in calculation. A derivation of the method is given in the next section.

3.1 The Discontinuous Galerkin Method

3.1.1 Differential Operators

Before delving into the method, the differential operators to be used in describing the method are defined. Assume the convention for divergence of a vector is

$$\nabla \cdot \boldsymbol{a} = \frac{\partial a_i}{\partial x_i}. \tag{3.1.1}$$

For a 2nd order tensor,

$$\left(\nabla \cdot \overline{\overline{A}}\right)_i = \frac{\partial A_{ij}}{\partial x_j}. \tag{3.1.2}$$

The gradient of a vector is

$$(\nabla \boldsymbol{a})_{ij} = \frac{\partial a_i}{\partial x_j}. \tag{3.1.3}$$

With this convention, the Laplacian is

$$\Delta \boldsymbol{a} = \frac{\partial^2 a_i}{\partial x_j \partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\partial a_i}{\partial x_j}\right) = \frac{\partial}{\partial x_j}(\nabla \boldsymbol{a})_{ij} = \nabla \cdot \nabla \boldsymbol{a}. \tag{3.1.4}$$

A curl of a vector is defined as

$$(\nabla \times \boldsymbol{a})_i = \epsilon_{ijk}\frac{\partial a_k}{\partial x_j}. \tag{3.1.5}$$

3.1.2 Derivation of Method

Consider Eq. (3.0.1), without the diffusive term

$$\frac{\partial q_i}{\partial t} + \frac{\partial \mathcal{F}_{ij}}{\partial x_j} = S_i. \tag{3.1.6}$$

This equation determines the evolution of a set of conserved variables, $\boldsymbol{q}$, which for example, can be given by mass, momentum, and energy, $\boldsymbol{q} = [\rho, \rho\boldsymbol{v}, e]^T$ for a species α in the $5N$-moment model. This is solved over some physical domain Ω with boundary $\partial\Omega$. The domain is divided into a set of Λ elements, so that the overall solution on $\boldsymbol{q}$ is made up of the individual solution on each element λ

$$\boldsymbol{q}(\boldsymbol{x},t) \simeq \bigoplus_{\lambda=1}^{\Lambda} \boldsymbol{q}^{\lambda}(\boldsymbol{x},t). \tag{3.1.7}$$

Before moving on, note that a coordinate transformation from some isoparametric space, given by spatial coordinate ξ_i to the real space x_i is made through some Jacobian transformation. This will help in calculations upon each individual element,

$$J_{ij}^{\lambda} = \frac{\partial x_i}{\partial \xi_j}. \tag{3.1.8}$$

Its inverse can also be constructed

$$J_{ij}^{\lambda\,-1} = \frac{\partial \xi_i}{\partial x_j}. \tag{3.1.9}$$

Applying this to Eq. (3.1.6) yields

$$\frac{\partial q_i^{\lambda}}{\partial t} + J_{kj}^{\lambda\,-1} \frac{\partial \mathcal{F}_{ij}^{\lambda}}{\partial \xi_k^{\lambda}} = S_i^{\lambda}. \tag{3.1.10}$$

Equations (3.0.1) and (3.1.6) both represent a strong form of the conservation law. For the discontinuous Galerkin method (as well as for the finite volume and continuous Galerkin methods) the equations are rewritten into the weak form. To arrive at this, multiply these local solutions by a set of element-wise smooth test functions ϕ_m up to N_p and integrate over each particular element, D^{λ}, which leads to

$$\int_{D^{\lambda}} \left(\frac{\partial q_i^{\lambda}}{\partial t} + J_{kj}^{\lambda\,-1} \frac{\partial \mathcal{F}_{ij}^{\lambda}}{\partial \xi_k} - S_i^{\lambda} \right) \phi_m d\boldsymbol{\xi} = 0, \text{ for } m = 1, 2, \cdots N_p. \tag{3.1.11}$$

The middle term on the flux can be broken down into volume and surface terms, first by applying the product rule

$$\int_{D^{\lambda}} \frac{\partial \mathcal{F}_{ij}^{\lambda}}{\partial \xi_k} \phi_m d\boldsymbol{\xi} = \int_{D^{\lambda}} \frac{\partial \left(\mathcal{F}_{ij}^{\lambda} \phi_m \right)}{\partial \xi_k} d\boldsymbol{\xi} - \int_{D^{\lambda}} \mathcal{F}_{ij}^{\lambda} \frac{\partial \phi_m}{\partial \xi_k} d\boldsymbol{\xi}, \tag{3.1.12}$$

and then by applying the divergence rule on the first term on the right hand side

$$\int_{D^{\lambda}} \frac{\partial \left(\mathcal{F}_{ij}^{\lambda} \phi_m \right)}{\partial \xi_k} d\boldsymbol{\xi} = \int_{\partial D^{\lambda}} \left(\mathcal{F}_{ij}^{\lambda} m_k \phi_m \right) d\boldsymbol{\xi}, \tag{3.1.13}$$

with m_k representing the outward-pointing normal along the surface ∂D^λ of element D^λ in isoparametric space. Inserting Eqs. (3.1.12) and (3.1.13) into Eq. (3.1.11) leads to

$$\int_{D^\lambda} \left(\frac{\partial q_i^\lambda}{\partial t} \phi_m - {J_{kj}^\lambda}^{-1} \mathcal{F}_{ij}^\lambda \frac{\partial \phi_m}{\partial \xi_k} - S_i^\lambda \phi_m \right) d\boldsymbol{\xi} = -\int_{\partial D^\lambda} {J_{kj}^\lambda}^{-1} \mathcal{F}_{ij}^* m_k \phi_m d\boldsymbol{\xi}, \tag{3.1.14}$$

where a unique $\mathcal{F}^*$ is denoted for interface ∂D^λ indicating the flux between adjoining elements must be the same. Next, assume the solution q^λ, the flux, $\mathcal{F}^\lambda$, and source terms can be expanded upon summations of another basis ψ_n,

$$\boldsymbol{q}^\lambda = \sum_{n=1}^{N_p} \hat{\boldsymbol{q}}_n^\lambda \psi_n,\ \mathcal{F}^\lambda = \sum_{n=1}^{N_p} \hat{\mathcal{F}}_n^\lambda \psi_n,\ \boldsymbol{S}^\lambda = \sum_{n=1}^{N_p} \hat{\boldsymbol{S}}_n^\lambda \psi_n, \tag{3.1.15}$$

where the hatted terms are the coefficients on each term of the expansion. Substitution of these terms into Eq. (3.1.14) yields for the m^{th} test function ϕ_m, given by

$$\sum_{n=1}^{N_p} \int_{D^\lambda} \left(\frac{\partial \hat{q}_{in}^\lambda}{\partial t} \psi_n \phi_m - {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^\lambda \psi_n \frac{\partial \phi_m}{\partial \xi_k} - \hat{S}_{in}^\lambda \psi_n \phi_m \right) d\boldsymbol{\xi} = -\sum_{n=1}^{N_p} \int_{\partial D^\lambda} {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^* \psi_n m_k \phi_m d\boldsymbol{\xi}. \tag{3.1.16}$$

If the space of test functions ϕ and basis expansions ψ are chosen to be the same, that is $\phi_m = \psi_m$, the method is known to be Galerkin. Making this substitution leads to

$$\sum_{n=1}^{N_p} \int_{D^\lambda} \left(\frac{\partial \hat{q}_{in}^\lambda}{\partial t} \psi_n \psi_m - {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^\lambda \psi_n \frac{\partial \psi_m}{\partial \xi_k} - \hat{S}_{in}^\lambda \psi_n \psi_m \right) d\boldsymbol{\xi} = -\sum_{n=1}^{N_p} \int_{\partial D^\lambda} {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^* \psi_n m_k \psi_m d\boldsymbol{\xi}. \tag{3.1.17}$$

Note that simply choosing the single basis ($N_p = 1$) $\psi_n = 1$, yields the finite volume method. If C^0 continuity is imposed, there can be no flux between elements and the right hand side surface term goes away, leading to the continuous Galerkin finite element method. Now, for clarity, assume the Einstein summation convention and drop the sum on index n,

$$\int_{D^\lambda} \left(\frac{\partial \hat{q}_{in}^\lambda}{\partial t} \psi_n \psi_m - {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^\lambda \psi_n \frac{\partial}{\partial \xi_k} \psi_m - \hat{S}_{in}^\lambda \psi_n \psi_m \right) d\boldsymbol{\xi} = -\int_{\partial D^\lambda} {J_{kj}^\lambda}^{-1} \hat{\mathcal{F}}_{ijn}^* m_k \psi_n \psi_m d\boldsymbol{\xi}. \tag{3.1.18}$$

The coefficients can be moved out of the spatially-dependent integrals

$$\begin{aligned}\frac{\partial \hat{q}^{\lambda}_{in}}{\partial t}\int_{D^{\lambda}}\psi_n\psi_m d\boldsymbol{\xi} - {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\int_{D^{\lambda}}\psi_n\frac{\partial}{\partial \xi_k}\psi_m d\boldsymbol{\xi} - \hat{S}^{\lambda}_{in}\int_{D^{\lambda}}\psi_n\psi_m d\boldsymbol{\xi}& \\ = - {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_n\psi_m d\boldsymbol{\xi}.&\end{aligned} \tag{3.1.19}$$

Rearranging terms to show an equation for the evolution of the conserved variable (also swap the ψ_n and ψ_m order) leads to

$$\begin{aligned}\frac{\partial \hat{q}^{\lambda}_{in}}{\partial t}\int_{D^{\lambda}}\psi_m\psi_n d\boldsymbol{\xi} = \Bigg(& {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\int_{D^{\lambda}}\psi_n\frac{\partial}{\partial \xi_k}\psi_m d\boldsymbol{\xi} - {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_m\psi_n d\boldsymbol{\xi} \\ & + \hat{S}^{\lambda}_{in}\int_{D^{\lambda}}\psi_m\psi_n d\boldsymbol{\xi}\Bigg).\end{aligned} \tag{3.1.20}$$

Left-multiplying both sides by $\left(\int_{D^{\lambda}}\psi_m\psi_n d\boldsymbol{\xi}\right)^{-1}_{lm}$ yields

$$\begin{aligned}\frac{\partial \hat{q}^{\lambda}_{il}}{\partial t} = & \left(\int_{D^{\lambda}}\psi_m\psi_n d\boldsymbol{\xi}\right)^{-1}_{lm}\Bigg({J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\int_{D^{\lambda}}\psi_n\frac{\partial}{\partial \xi_k}\psi_m d\boldsymbol{\xi} - {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_m\psi_n d\boldsymbol{\xi} \\ & + \hat{S}^{\lambda}_{in}\int_{D^{\lambda}}\psi_m\psi_n d\boldsymbol{\xi}\Bigg).\end{aligned} \tag{3.1.21}$$

This completes the conversion from the strong form partial differential equation given in Eq. (3.1.6) to a weak form ordinary differential equation, assuming the basis functions and their derivatives are known. In practice, the integrals which are only spatially-dependent can be pre-calculated and thus the equation can be advanced using a time integration scheme. At this point, define the mass matrix

$$\mathcal{M}_{ij} \equiv \int_{D^{\lambda}}\psi_i\psi_j d\boldsymbol{\xi}, \tag{3.1.22}$$

and advection matrix

$$\mathcal{A}_{ijk} \equiv \int_{D^{\lambda}}\psi_i\frac{\partial}{\partial \xi_j}\psi_k d\boldsymbol{\xi}. \tag{3.1.23}$$

Substituting in these definitions, Eq. (3.1.21) becomes

$$\frac{\partial \hat{q}^{\lambda}_{il}}{\partial t} = \mathcal{M}^{-1}_{lm}\mathcal{A}_{nkm}{J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn} - \mathcal{M}^{-1}_{lm}{J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_m\psi_n d\boldsymbol{\xi} + \mathcal{M}^{-1}_{lm}\mathcal{M}_{mn}\hat{S}^{\lambda}_{in} \tag{3.1.24}$$

or

$$\frac{\partial \hat{q}^{\lambda}_{il}}{\partial t} = {J^{\lambda}_{kj}}^{-1}\mathcal{M}^{-1}_{lm}\mathcal{A}_{nkm}\hat{\mathcal{F}}^{\lambda}_{ijn} - {J^{\lambda}_{kj}}^{-1}\mathcal{M}^{-1}_{lm}\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_m\psi_n d\boldsymbol{\xi} + \hat{S}^{\lambda}_{il}. \tag{3.1.25}$$

Also, given straight-sided (simplex) elements the surface integral can be written as

$$\hat{\mathcal{F}}^{*}_{ijn}\int_{\partial D^{\lambda}} m_k\psi_m\psi_n d\boldsymbol{\xi} = \sum_{\gamma\in\bar{\Gamma}_{\lambda}} \hat{\mathcal{F}}^{\lambda\gamma}_{ijn} m^{\lambda\gamma}_{k}\int_{\partial D^{\lambda\gamma}} \psi_m\psi_n d\boldsymbol{\xi} = \sum_{\gamma\in\bar{\Gamma}_{\lambda}} \hat{\mathcal{F}}^{\lambda\gamma}_{ijn} m^{\lambda\gamma}_{k}\Gamma^{\lambda\gamma}_{mn}, \tag{3.1.26}$$

where the surface integral has been broken up into integrals along faces γ in the set of total faces $\bar{\Gamma}$ associated with element λ and a mass matrix defined by an integral along a simplex face is defined as

$$\Gamma^{\lambda\gamma}_{ij} \equiv \int_{\partial D^{\lambda\gamma}} \psi_i\psi_j d\boldsymbol{\xi}. \tag{3.1.27}$$

Next, define basis arrays based on these matrices that incorporate the inverse mass matrix

$$\Upsilon_{ijk} \equiv \mathcal{M}^{-1}_{il}\mathcal{A}_{kjl}, \tag{3.1.28}$$

$$\Xi^{\lambda\gamma}_{ij} \equiv \mathcal{M}^{-1}_{il}\Gamma^{\lambda\gamma}_{lj}. \tag{3.1.29}$$

Substitution of Eqs. (3.1.28), (3.1.26), and (3.1.29) into Eq. (3.1.25) yields

$$\frac{\partial \hat{q}^{\lambda}_{il}}{\partial t} = {J^{\lambda}_{kj}}^{-1}\Upsilon_{lkn}\hat{\mathcal{F}}^{\lambda}_{ijn} - \sum_{\gamma\in\bar{\Gamma}_{\lambda}} {J^{\lambda}_{kj}}^{-1}\hat{\mathcal{F}}^{\lambda\gamma}_{ijn} m^{\lambda\gamma}_{k}\Xi^{\lambda\gamma}_{ln} + \hat{S}^{\lambda}_{il}. \tag{3.1.30}$$

Now performing a change of indexes such that $l \to j$, $j \to m$, $k \to l$, $n \to k$ leads to

$$\frac{\partial \hat{q}^{\lambda}_{ij}}{\partial t} = {J^{\lambda}_{lm}}^{-1}\Upsilon_{jlk}\hat{\mathcal{F}}^{\lambda}_{imk} - \sum_{\gamma\in\bar{\Gamma}_{\lambda}} {J^{\lambda}_{lm}}^{-1} m^{\lambda\gamma}_{l}\Xi^{\lambda\gamma}_{jk}\hat{\mathcal{F}}^{\lambda\gamma}_{imk} + \hat{S}^{\lambda}_{ij}. \tag{3.1.31}$$

This can be further simplified by converting the isoparametric normal $m^{\lambda\gamma}_{l}$ to a real space normal, $n^{\lambda\gamma}_{l}$. Notice that the isoparametric normal can be written

$$m^{\lambda\gamma}_{j} = \frac{\partial m^{\lambda\gamma}_{\xi_j}}{\partial \xi_j}\Delta\xi_j \tag{3.1.32}$$

assuming no Einstein summation convention. Then using the Jacobian

$$n_i^{\lambda\gamma} = \frac{J_{ji}^{\lambda\ -1} m_j^{\lambda\gamma}}{\left|\left|J_{ji}^{\lambda\ -1} m_j^{\lambda\gamma}\right|\right|}, \tag{3.1.33}$$

where the denominator is for unitization of the normal which will be defined

$$G_{\lambda\gamma}^{-1} \equiv \left|\left|J_{ji}^{\lambda\ -1} m_j^{\lambda\gamma}\right|\right| = \left(J_{ji}^{\lambda\ -1} m_j J_{ki}^{\lambda\ -1} m_k\right)^{\frac{1}{2}}. \tag{3.1.34}$$

Substituting this into Eq. (3.1.31) leads to

$$\frac{\partial \hat{q}_{ij}^{\lambda}}{\partial t} = J_{lm}^{\lambda\ -1} \Upsilon_{jlk} \hat{\mathcal{F}}_{imk}^{\lambda} - \sum_{\gamma \in \bar{\Gamma}_\lambda} G_{\lambda\gamma}^{-1} n_m^{\lambda\gamma} \Xi_{jk}^{\lambda\gamma} \hat{\mathcal{F}}_{imk}^{\lambda\gamma} + \hat{S}_{ij}^{\lambda}. \tag{3.1.35}$$

Also define

$$\tilde{\mathcal{F}}_i^{\lambda\gamma} = n_m \hat{\mathcal{F}}_{im}^{\lambda\gamma} \tag{3.1.36}$$

so that the surface flux can be calculated directly in the plane of the surface. Substituting this into Eq. (3.1.35) achieves

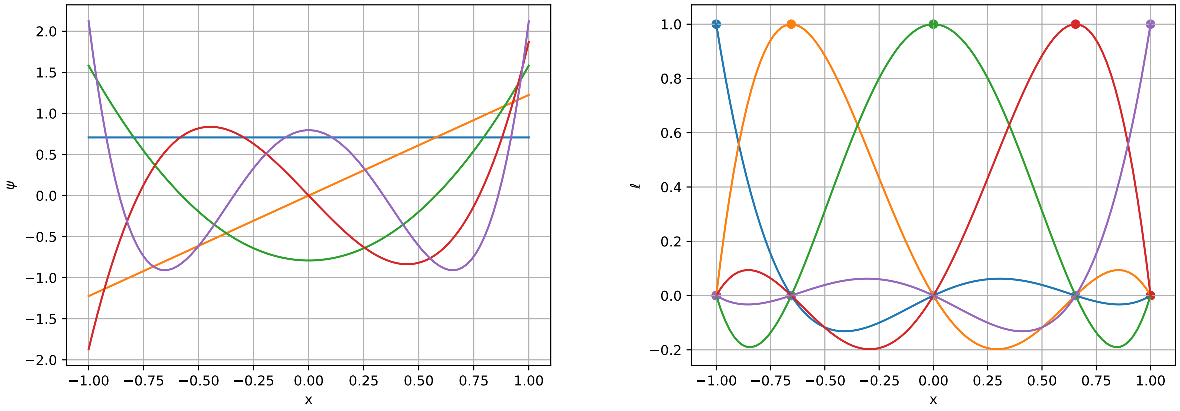
$$\frac{\partial \hat{q}_{ij}^{\lambda}}{\partial t} = J_{lm}^{\lambda\ -1} \Upsilon_{jlk} \hat{\mathcal{F}}_{imk}^{\lambda} - \sum_{\gamma \in \bar{\Gamma}_\lambda} G_{\lambda\gamma}^{-1} \Xi_{jk}^{\lambda\gamma} \tilde{\mathcal{F}}_{ik}^{\lambda\gamma} + \hat{S}_{ij}^{\lambda}. \tag{3.1.37}$$

This is the discretized ordinary differential equation translated from the conservation law given by Eq. (3.0.1), solved for each element λ. It describes the evolution of conserved variable component i of $\hat{q}$ with basis weight component j in element λ through a volume integral contribution (first right term) and surface integral term (second right term) as well as from the source term. The basis arrays Υ and Ξ, are element geometry and basis-dependent and precomputed. The Jacobian terms J and G are also element geometry dependent and can be precomputed. The internal flux terms $\hat{\mathcal{F}}$ are directly the fluxes in the model equations as written in the conservation law in Eq. (3.0.1), where index i represents again the variable component, m represents the spatial dimension, and k represents the expansion basis. The numerical flux $\tilde{F}$ is the flux from the equations at element boundaries, which must be dealt with correctly by the method. Source terms contribute to the volume integral directly through $\hat{S}$.

3.1.3 Nodal vs Modal Forms

In the derivation of the DG method in the previous section, mention of the basis functions has been in the abstract. In practice, some basis set must be chosen to make the calculation of Eqs. (3.1.15) realizable and efficient. One choice is a modal expansion for some set of functions defined on the isoparametric element, for which the coefficients in Eq. (3.1.15) (e.g. $\hat{\boldsymbol{q}}_n^\lambda$) are interpreted as coefficients on each basis function, ψ_n. The choice of functions has to made carefully to ensure a well-conditioned mass matrix. An orthogonal basis set such as Legendre polynomials (shown in Fig. 3.1a) ensures this. Another choice is a nodal representation, defined by an interpolation polynomial, such as Lagrange polynomials (shown in Fig. 3.1b), where the coefficients $\hat{\boldsymbol{q}}_n^\lambda$ are actual solution values at particular node locations where $\psi_n(\boldsymbol{x}_n) = 1$ and all other $\psi_n(\boldsymbol{x}_{m\neq n}) = 0$. The choice of these node locations must be done carefully as to avoid Runge's phenomenon, which causes increasing oscillation of the solution within the element. In practice, when choosing a nodal representation, an appropriate modal representation may first be chosen, from which interpolation points are chosen such that the Vandermonde matrix mapping modal coefficients to nodal coefficients is well-conditioned, leading to interpolation points at Legendre-Gauss-Lobatto quadrature points. Further details on the choice of basis and the DG method in general can be found in the book on nodal discontinuous Galerkin methods by Hesthaven and Warburton [55].

A significant advantage of using the modal form, assuming the use of an orthogonal basis set is that the mass matrix becomes diagonal and thus trivial to invert. Additionally, limiting and filtering of the solution can be done simply by dropping higher order modes. The nodal form is advantageous in that integration becomes simpler due to basis functions being 1 or 0 at node locations. Boundary and element interface values also become trivial to calculate, since their values at those points directly correspond to the basis function coefficient. The fact that the coefficients are nodal values also makes interpretation and post-processing of the solution simpler. In this work, the nodal representation using Lagrange polynomials is used for the bases. Thus, basis coefficients as referred to in the previous section can simply



(a) Modal

(b) Nodal

Figure 3.1: Modal Basis of Legendre Polynomials (3.1a) and Nodal Basis of Lagrange Polynomials (3.1b).

be thought of as solution values at node locations.

3.1.4 Numerical Flux

The choice of numerical flux $\tilde{\mathcal{F}}_{ik}^{\lambda\gamma}$ at element interfaces is a central and non-trivial aspect to the discontinuous Galerkin method. Fortunately, this part of the scheme manifests in the same manner as in finite volume methods, and thus these methods can be carried over to DG. Numerical flux schemes can be found in the finite volume books by Leveque [70] and Toro [119], as well as in Hesthaven and Warburton [55]. An exact Riemann solver that tracks jumps from all waves in the system emanating from the element interface would yield the correct numerical flux, but is expensive in general for systems of nonlinear equations, as is the case for the MHD and $5N$-moment equations. Thus, approximate Riemann solvers can be developed to do this more efficiently, though will not necessarily achieve the exact solution. In the next few subsections, the numerical flux schemes for the equation sets considered in this work are summarized.

Fluid and MHD Equations

The hyperbolic portion of the $5N$-moment model is the Euler equation which is a nonlinear set for 5 equations producing 5 waves. The ideal MHD equations are similar, with additional equations for the magnetic field. The simplest numerical flux to use in these cases is the Rusanov, or Local Lax-Friedrichs flux, where only the fastest wave is used to determine the flux. Along the direction of the face normal, this is defined as

$$\tilde{\mathcal{F}}_{ik}^{\lambda\gamma} = \left[\frac{1}{2} \left(\hat{\mathcal{F}}_{imk}^{\lambda\gamma^+} + \hat{\mathcal{F}}_{imk}^{\lambda\gamma^-} \right) + \frac{1}{2} \left| c_{\max} \right| n_m^- \left(q_{ik}^{\lambda\gamma^-} - q_{ik}^{\lambda\gamma^+} \right) \right] n_m^- \tag{3.1.38}$$

or

$$\tilde{\mathcal{F}}_{ik}^{\lambda\gamma} = \frac{1}{2} \left(\tilde{\mathcal{F}}_{ik}^{\lambda\gamma^+} + \tilde{\mathcal{F}}_{ik}^{\lambda\gamma^-} \right) + \frac{1}{2} \left| c_{\max} \right| \left(q_{ik}^{\lambda\gamma^-} - q_{ik}^{\lambda\gamma^+} \right) \tag{3.1.39}$$

using Eq. (3.1.36). Here the $-$ indicates the values on the inside of the element (λ side with n^- pointing outward from the element λ). The $+$ represents the outside element (γ). The advantage of the nodal method can be seen here, as in this case the k index is the node on face $\lambda\gamma^\pm$ so that the flux and variable value can be readily taken there as it is the only basis component with a nonzero ($= 1$) coefficient. $c_{\max}$ is simply the maximum wavespeed in the system. This is the simplest to code and most stable of numerical fluxes, however is also the most diffusive, potentially over-smearing out discontinuous phenomena. A less diffusive solution would be to use an HLL-type numerical flux, so called for the formulation developed by Harten, Lax, and van Leer [54], in which the system may be assumed to have less than the actual number of waves which are shocks, upon which the Rankine-Hugoniot conditions can be applied to determine the state of solution at the interface, and thus the flux. For ideal MHD, in addition to the Rusanov flux, which uses the fast magnetosonic speed, HLL and HLLD fluxes have been implemented, as described in Ref. [80].

Another approximate Riemann solver used in this work is a Roe flux which linearizes the flux Eq. (3.1.6) where the resulting flux Jacobian is an approximation at the element interface that maintains hyperbolicity (diagonizable with real eigenvalues) and consistency (approaches the true flux Jacobian as q on either side approach each other). The linearization is based on the idea that near a strong shock, a large jump exists in at most one wave family of the system. The derivation of this for the equations used in the $5N$-moment model is based on a first order flux in the local frame of the element interface such that $\boldsymbol{n}^{-} = (+1, 0, 0)$, given by

$$F_{j+1/2} = F_{j+1}^{-} + F_{j}^{+} = A^{-}Q_{j+1} + A^{+}Q_{j} = \frac{1}{2}\left(F_j + F_{j+1}\right) - \frac{1}{2}\left(X\left|\Lambda\right|X^{-1}\right)\left(Q_{j+1} - Q_j\right), \tag{3.1.40}$$

where $A \equiv \frac{\partial F}{\partial Q} = X\Lambda X^{-1}$ is the flux Jacobian matrix with right eigvenvector matrix X, left eigenvector matrix X^{-1}, and Λ the diagonal matrix of eigenvalues λ which are wave speeds. The term $A^{-}Q_{j+1} = X\Lambda^{-}X^{-1}Q_{j+1}$ represents the left-going waves from the element Q_{j+1} at the interface $j + \frac{1}{2}$ while the term $A^{+}Q_j = X\Lambda^{+}X^{-1}Q_j$ represents the right-going waves from the element Q_j at the interface $j + \frac{1}{2}$. Λ^{-} and Λ^{+} are the diagonal matrices of left-going and right-going wave speeds, respectively. Further details of this method are described in Ref. [70]. Note that one can use a rotation matrix to rotate into this frame of reference to then calculate the flux in this frame, which can then be converted back to the global frame of the element interface by then antirotating this local flux vector. Previous students applied a Roe flux [103] within the context of this scheme for the Euler equations making up the $5N$-moment model. Additionally, for this work, a Roe solver for the MHD equations has also been implemented, based on the algorithm described in Refs. [92] and [93]. In these papers, the Λ, X, and X^{-1} matrices are derived, though in primitive variable form. That is given conservative variables $Q = \left[\rho, \boldsymbol{p} = \rho\boldsymbol{v}, \boldsymbol{B}, e\right]^T$ and nonconservative variables $W = \left[\rho, \boldsymbol{v}, \boldsymbol{B}, p\right]^T$, Eq. (3.1.40) can be rewritten by letting $S \equiv \frac{\partial Q}{\partial W}$ and $S^{-1} = \frac{\partial W}{\partial Q}$ so that

$$A_p = S^{-1}AS = S^{-1}X\Lambda X^{-1}S = X_p\Lambda X_p^{-1}, \tag{3.1.41}$$

leading to

$$X = SX_p \tag{3.1.42}$$

and

$$X^{-1} = X_p^{-1}S^{-1}, \tag{3.1.43}$$

which can be substituted to find

$$F_{j+1/2} = \frac{1}{2}\left(F_j + F_{j+1}\right) - \frac{1}{2}\left(S X_p \left|\Lambda\right| X_p^{-1} S^{-1}\right)\left(Q_{j+1} - Q_j\right), \tag{3.1.44}$$

or by index notation

$$F_i^{j+1/2} = \frac{1}{2}\left(F_i^j + F_i^{j+1}\right) - \frac{1}{2}\left(S_{ik} X^p_{kl} \left|\Lambda\right|_{lm} {X^p_{mn}}^{-1} S^{-1}_{np}\right)\left(Q_p^{j+1} - Q_p^j\right). \tag{3.1.45}$$

Thus by calculating S and S^{-1}, and formulating Λ, X_p, and X_p^{-1} as derived in Refs. [92] and [93], the MHD Roe flux is determined.

Vlasov Equation

For systems with a single wave speed at a particular location, the correct flux is given by the Rusanov flux in Eq. (3.1.38). This is the case for the Vlasov equation with single wave speeds along spatial and velocity faces, giving a numerical flux of the form

$$\tilde{\mathcal{F}}_{\boldsymbol{x}}^{\lambda\gamma} = \left[\frac{1}{2}\left(\hat{F}_{\boldsymbol{x}}^{\lambda\gamma^+} + \hat{F}_{\boldsymbol{x}}^{\lambda\gamma^-}\right) + \frac{1}{2}\left|\boldsymbol{v}\cdot\hat{\boldsymbol{n}}_{\boldsymbol{x}}^-\right|\left(f^{\lambda\gamma^-} - f^{\lambda\gamma^+}\right)\hat{\boldsymbol{n}}_{\boldsymbol{x}}^-\right]\cdot\hat{\boldsymbol{n}}_{\boldsymbol{x}}^-, \tag{3.1.46a}$$

$$\tilde{\mathcal{F}}_{\boldsymbol{v}}^{\lambda\gamma} = \left[\frac{1}{2}\left(\hat{F}_{\boldsymbol{v}}^{\lambda\gamma^+} + \hat{F}_{\boldsymbol{v}}^{\lambda\gamma^-}\right) + \frac{1}{2}\left|\boldsymbol{a}\cdot\hat{\boldsymbol{n}}_{\boldsymbol{v}}^-\right|\left(f^{\lambda\gamma^-} - f^{\lambda\gamma^+}\right)\hat{\boldsymbol{n}}_{\boldsymbol{v}}^-\right]\cdot\hat{\boldsymbol{n}}_{\boldsymbol{v}}^-, \tag{3.1.46b}$$

respectively for each face type, where $\mathcal{F}_{\boldsymbol{x}} = \boldsymbol{v} f$, $\mathcal{F}_{\boldsymbol{v}} = \boldsymbol{a} f$, and $\boldsymbol{a} = \left(\frac{L}{\delta_p}\right)\frac{Z_\alpha}{A_\alpha}\left(\boldsymbol{E} + \boldsymbol{v}\times\boldsymbol{B}\right)$, as can be seen from Eq. (2.3.7).

Maxwell Equations

Maxwell's equations are a linear system with the single wave speed being the speed of light. Thus, this is simpler than the nonlinear fluid equations but must be determined for the Ampere and Faraday system in Eqs. (2.3.2) and (2.3.4), respectively. One can follow the diagonalization procedure in Eq. (3.1.40) to find the local frame numerical flux as

$$\tilde{\mathcal{F}}^{\lambda\gamma}_{\boldsymbol{E},\boldsymbol{B}} = \begin{pmatrix} 0 \\ \frac{1}{2}c^2\left(B_z^- + B_z^+\right) + \frac{1}{2}c\left(E_y^- - E_y^+\right) \\ -\frac{1}{2}c^2\left(B_y^- + B_y^+\right) + \frac{1}{2}c\left(E_z^- - E_z^+\right) \\ 0 \\ -\frac{1}{2}\left(E_z^- + E_z^+\right) + \frac{1}{2}c\left(B_y^- - B_y^+\right) \\ \frac{1}{2}\left(E_y^- + E_y^+\right) + \frac{1}{2}c\left(B_z^- - B_z^+\right) \end{pmatrix} \tag{3.1.47}$$

where the first 3 elements of this vector are the numerical fluxes for the electric field evolution as written in Ampere's Law and the last 3 are the numerical fluxes for the magnetic field evolution as written in Faraday's Law. As with the Vlasov equation, these fluxes are exact.

3.2 Higher Order Derivatives

The DG method described in the previous section assumes first order derivatives, however, there are terms in the models from Chapter 2 that have higher order derivatives, such as Hall MHD. Thus, a method for handling these terms is needed. The local Discontinuous Galerkin method (LDG) was introduced by Cockburn and Shu [31] and was expanded in Refs. [25, 21, 27]. A set of variants of this known as interior penalty methods have also been studied. Overviews of these methods can be found in Refs. [2, 102, 55]. In these methods, a second order differential equation is split into a system of first order differential equations, upon which each is turned into the weak form in the familiar Galerkin manner.

The derivation in most of the literature involves consideration of the Poisson problem

$$-\Delta \boldsymbol{q} = \boldsymbol{f}, \tag{3.2.1}$$

or converted to index form

$$-\frac{\partial^2 q_i}{\partial x_j \partial x_j} = f_i, \tag{3.2.2}$$

where $\boldsymbol{f}$ is an arbitrary operator, such as $\frac{\partial u_i}{\partial t}$. It is then a system of 2 first order differential equations,

$$\sigma_{ij} = \frac{\partial q_i}{\partial x_j}, \tag{3.2.3}$$

$$-\frac{\partial \sigma_{ij}}{\partial x_j} = f_i. \tag{3.2.4}$$

In this work, $\boldsymbol{q}$ is time advanced as part of more complex equation sets, so the general problem is written as the conservation law in Eq. (3.0.1) with a diffusion term, given by

$$\frac{\partial q_i}{\partial t} + \frac{\partial \mathcal{F}(q)_{ij}}{\partial x_j} + \frac{\partial \mathcal{D}(q, \nabla q)_{ij}}{\partial x_j} = S_i, \tag{3.2.5}$$

where the $\mathcal{F}$ and $\mathcal{S}$ terms were discretized in Sec. 3.1.2. In this section, focus is on the diffusion term in particular,

$$\frac{\partial q_i}{\partial t} + \frac{\partial \mathcal{D}(q, \nabla q)_{ij}}{\partial x_j} = 0, \tag{3.2.6}$$

This is then broken up into the equation for the gradient, which is still Eq. (3.2.3) and

$$\frac{\partial q_i}{\partial t} + \frac{\partial \mathcal{D}(q, \sigma)_{ij}}{\partial x_j} = 0. \tag{3.2.7}$$

Now the aim is to solve this equation on a physical domain Ω with boundary $\partial\Omega$ broken up into elements in space Λ just as in Sec. 3.1.2, so for each element, invoking the Jacobian definitions in Eqs. (3.1.8) and (3.1.9), these equations can be written as

$$\sigma^{\lambda}_{ij} = {J^{\lambda}_{kj}}^{-1} \frac{\partial q^{\lambda}_i}{\partial \xi^{\lambda}_k}, \tag{3.2.8}$$

$$\frac{\partial q^{\lambda}_i}{\partial t} + {J^{\lambda}_{kj}}^{-1} \frac{\partial \mathcal{D}^{\lambda}_{ij}}{\partial \xi^{\lambda}_k} = 0. \tag{3.2.9}$$

Note that Eqs. (3.2.9) and (3.1.10) are identical except for the first's lack of source and differing operators on which the divergence is acting. Just as before, it is desirable to convert these expressions into the weak form. For Eq. (3.2.8), multiply by test function ψ and integrate over the element D^λ.

$$\int_{D^\lambda} \sigma^\lambda_{ij}\psi_n d\boldsymbol{\xi} = \int_{D^\lambda} {J^\lambda_{kj}}^{-1}\frac{\partial q^\lambda_i}{\partial \xi^\lambda_k}\psi_n d\boldsymbol{\xi}. \tag{3.2.10}$$

Apply the product rule

$$\int_{D^\lambda} \sigma^\lambda_{ij}\psi_n d\boldsymbol{\xi} = {J^\lambda_{kj}}^{-1}\int_{D^\lambda} \frac{\partial\left(q^\lambda_i\psi_n\right)}{\partial \xi^\lambda_k} d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\int_{D^\lambda} q^\lambda_i\frac{\partial \psi_n}{\partial \xi^\lambda_k} d\boldsymbol{\xi}. \tag{3.2.11}$$

Apply the divergence rule

$$\int_{D^\lambda} \sigma^\lambda_{ij}\psi_n d\boldsymbol{\xi} = {J^\lambda_{kj}}^{-1}\int_{\partial D^\lambda} {q^\lambda_i}^*\psi_n m_k d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\int_{D^\lambda} q^\lambda_i\frac{\partial \psi_n}{\partial \xi^\lambda_k} d\boldsymbol{\xi}. \tag{3.2.12}$$

Now expand upon basis functions (we make Galerkin choice of ψ again) as in Eq. (3.1.15)

$$\hat{\sigma}^\lambda_{ijl}\int_{D^\lambda} \psi_l\psi_n d\boldsymbol{\xi} = {J^\lambda_{kj}}^{-1}\hat{q}^{\lambda^*}_{il}\int_{\partial D^\lambda} \psi_l\psi_n m_k d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\int_{D^\lambda} \psi_l\frac{\partial \psi_n}{\partial \xi^\lambda_k} d\boldsymbol{\xi}. \tag{3.2.13}$$

Flipping l and n for the next multiplication is allowed due to symmetry. This gives

$$\begin{aligned}
\hat{\sigma}^\lambda_{ijl}\mathcal{M}_{nl} &= {J^\lambda_{kj}}^{-1}\hat{q}^{\lambda^*}_{il}\int_{\partial D^\lambda} \psi_n\psi_l m_k d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\mathcal{A}_{lkn},\\
\hat{\sigma}^\lambda_{ijl}\mathcal{M}^{-1}_{mn}\mathcal{M}_{nl} &= {J^\lambda_{kj}}^{-1}\hat{q}^{\lambda^*}_{il}\mathcal{M}^{-1}_{mn}\int_{\partial D^\lambda} \psi_n\psi_l m_k d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\mathcal{M}^{-1}_{mn}\mathcal{A}_{lkn},\\
\hat{\sigma}^\lambda_{ijm} &= {J^\lambda_{kj}}^{-1}\hat{q}^{\lambda^*}_{il}\mathcal{M}^{-1}_{mn}\int_{\partial D^\lambda} \psi_n\psi_l m_k d\boldsymbol{\xi} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\mathcal{M}^{-1}_{mn}\mathcal{A}_{lkn}.
\end{aligned} \tag{3.2.14}$$

As before, assuming straight-sided simplex elements leads to

$$\hat{\sigma}^\lambda_{ijm} = \sum_{\gamma\in\bar{\Gamma}_\lambda} {J^\lambda_{kj}}^{-1}\hat{q}^{\lambda\gamma}_{il}m^{\lambda\gamma}_k\mathcal{M}^{-1}_{mn}\Gamma^{\lambda\gamma}_{nl} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\mathcal{M}^{-1}_{mn}\mathcal{A}_{lkn}. \tag{3.2.15}$$

Removal of the isoparametric normal using Eq. (3.1.33) and (3.1.34) yields

$$\hat{\sigma}^\lambda_{ijm} = \sum_{\gamma\in\bar{\Gamma}_\lambda} G^{-1}_{\lambda\gamma}n^{\lambda\gamma}_j\hat{q}^{\lambda\gamma}_{il}\mathcal{M}^{-1}_{mn}\Gamma^{\lambda\gamma}_{nl} - {J^\lambda_{kj}}^{-1}\hat{q}^\lambda_{il}\mathcal{M}^{-1}_{mn}\mathcal{A}_{lkn}. \tag{3.2.16}$$

Let

$$\tilde{q}^{\lambda\gamma}_{ijk} = \hat{q}^{\lambda\gamma}_{ik}n^{\lambda\gamma}_j. \tag{3.2.17}$$

This yields

$$\hat{\sigma}^{\lambda}_{ijm} = \sum_{\gamma\in\bar{\Gamma}_{\lambda}} G^{-1}_{\lambda\gamma}\tilde{q}^{\lambda\gamma}_{ijl}\Xi^{\lambda\gamma}_{ml} - {J^{\lambda}_{kj}}^{-1}\hat{q}^{\lambda}_{il}\Upsilon_{mkl}. \tag{3.2.18}$$

Let $k \to l$, $l \to m$, $m \to k$, then

$$\hat{\sigma}^{\lambda}_{ijk} = \sum_{\gamma\in\bar{\Gamma}_{\lambda}} G^{-1}_{\lambda\gamma}\tilde{q}^{\lambda\gamma}_{ijm}\Xi^{\lambda\gamma}_{km} - {J^{\lambda}_{lj}}^{-1}\hat{q}^{\lambda}_{im}\Upsilon_{klm} \tag{3.2.19}$$

or equivalently

$$\hat{\sigma}^{\lambda}_{ijk} = \sum_{\gamma\in\bar{\Gamma}_{\lambda}} G^{-1}_{\lambda\gamma}\tilde{q}^{\lambda\gamma}_{ijl}\Xi^{\lambda\gamma}_{kl} - {J^{\lambda}_{lj}}^{-1}\hat{q}^{\lambda}_{im}\Upsilon_{klm}. \tag{3.2.20}$$

Going through the same procedure on Eq. (3.2.9) is equivalent to the derivation on Eq. (3.1.10) without the source term, as given in Sec. 3.1.2, leading to an analogous form of Eq. (3.1.37)

$$\frac{\partial \hat{q}^{\lambda}_{ij}}{\partial t} = {J^{\lambda}_{lm}}^{-1}\Upsilon_{jlk}\hat{\mathcal{D}}^{\lambda}_{imk} - \sum_{\gamma\in\bar{\Gamma}_{\lambda}} G^{-1}_{\lambda\gamma}\Xi^{\lambda\gamma}_{jk}\tilde{\mathcal{D}}^{\lambda\gamma}_{ik}, \tag{3.2.21}$$

where there is also an analogous form of Eq. (3.1.36)

$$\tilde{\mathcal{D}}^{\lambda\gamma}_{i} = n_m\hat{\mathcal{D}}^{\lambda\gamma}_{im}. \tag{3.2.22}$$

The rest of the section is about the choice of surface terms $\tilde{q}^{\lambda\gamma}$ and $\tilde{\sigma}^{\lambda\gamma}$ in Eqs. (3.2.20) and (3.2.21), respectively.

Arnold et. al [2] gives an overview and summary of some choices that have been applied. First, define averages

$$\left\{\hat{q}_{ik}\right\}^{\lambda\gamma} = \frac{1}{2}\left(\hat{q}^{\lambda}_{ik} + \hat{q}^{\gamma}_{ik}\right), \tag{3.2.23}$$

$$\left\{\hat{\mathcal{D}}_{ikj}\right\}^{\lambda\gamma} = \frac{1}{2}\left(\hat{\mathcal{D}}^{\lambda}_{ikj} + \hat{\mathcal{D}}^{\gamma}_{ikj}\right), \tag{3.2.24}$$

and jumps

$$\begin{aligned} [\![\hat{q}_{ik}]\!]^{\lambda\gamma}_{j} &= \hat{q}^{\lambda}_{ik}n^{\lambda\gamma}_{j} + \hat{q}^{\gamma}_{ik}n^{\gamma\lambda}_{j} \\ &= \left(\hat{q}^{\lambda}_{ik} - \hat{q}^{\gamma}_{ik}\right)n^{\lambda\gamma}_{j}, \end{aligned} \tag{3.2.25}$$

$$\begin{aligned}[\![\hat{\mathcal{D}}_{ikj}]\!]^{\lambda\gamma} =&\hat{\mathcal{D}}_{ikj}^{\lambda}n_k^{\lambda\gamma} + \hat{\mathcal{D}}_{ikj}^{\gamma}n_k^{\gamma\lambda}\\ =&\left(\hat{\mathcal{D}}_{ikj}^{\lambda} - \hat{\mathcal{D}}_{ikj}^{\gamma}\right) n_k^{\lambda\gamma}.\end{aligned} \tag{3.2.26}$$

The LDG formulation is based on the idea of upwinding the two first order equations in opposite directions, so as to apply the hyperbolic wave-propagation scheme to each individually, while the overall 2nd order equation has no preferential wave direction. Thus as shown in Refs. [55, 102, 16], choose for the fluxes

$$\hat{q}_{ik}^{\lambda\gamma} = \begin{cases} \{\hat{q}_{ik}\}^{\lambda\gamma} + \beta_m^{\lambda\gamma}[\![\hat{q}_{ik}]\!]_m^{\lambda\gamma} & \text{if on } \mathcal{E}_I \\ g_{D_{ik}} \quad \text{if on } \mathcal{E}_D \\ \hat{q}_{ik}^{\lambda} \quad \text{if on } \mathcal{E}_N \end{cases} \tag{3.2.27}$$

$$\hat{\mathcal{D}}_{ikj}^{\lambda\gamma} = \begin{cases} \left\{\hat{\mathcal{D}}(\hat{q}_{ij}, \hat{\sigma}_{ikj})\right\}^{\lambda\gamma} - \beta_k^{\lambda\gamma}[\![\hat{\mathcal{D}}(\hat{q}_{ij}, \hat{\sigma}_{imj})]\!]^{\lambda\gamma} - \alpha[\![\hat{q}_{ij}]\!]_k^{\lambda\gamma} & \text{if on } \mathcal{E}_I \\ \hat{\mathcal{D}}(\hat{q}_{ij}, \hat{\sigma}_{ikj})^{\lambda} - \alpha\left(\hat{q}_{ij}^{\lambda} - g_{D_{ij}}\right) n_k^{\lambda\gamma} \quad \text{if on } \mathcal{E}_D \\ \mathcal{D}(\hat{q}^{\lambda}, g_N)_{ij} n_k^{\lambda\gamma} (\text{equivalent to } \hat{\mathcal{D}}_{ikj}^{\lambda\gamma} n_k^{\lambda\gamma} = \mathcal{D}(\hat{q}^{\lambda}, g_N)_{ij}) \quad \text{if on } \mathcal{E}_N \end{cases} \tag{3.2.28}$$

with the boundary conditions described in Refs. [16, 88, 102]. For $\mathcal{E}_D$, the Dirichlet condition is

$$\hat{q}_{ik}^{D} = g_{D_{ik}}, \tag{3.2.29}$$

$$\hat{\sigma}_{ikj}^{D} n_k^{\lambda D} = \hat{\sigma}_{ikj}^{\lambda} n_k^{\lambda D}. \tag{3.2.30}$$

The Neumann condition is

$$\hat{q}_{ik}^{N} = \hat{q}_{ik}^{\lambda}, \tag{3.2.31}$$

$$\hat{\sigma}_{ikj}^{N} n_k^{\lambda N} = g_{N_{ij}}. \tag{3.2.32}$$

In these equations λ is always considered the inside element and γ is considered the outside element. With this is in mind, the normal $\boldsymbol{n}^{\lambda\gamma}$ is the normal pointing from element λ to element γ and $\boldsymbol{n}^{\gamma\lambda}$ is its negative. The vector $\boldsymbol{\beta}$ is a chosen common vector between elements λ and γ in the direction of either $\boldsymbol{n}^{\lambda\gamma}$ or $\boldsymbol{n}^{\gamma\lambda}$ with magnitude $\in (0,1)$. That is, $\boldsymbol{\beta}^{\lambda\gamma} = \boldsymbol{\beta}^{\gamma\lambda}$. Bounds for appropriate values of the penalty-weighting parameter, α, are discussed in Ref. [55].

Applying Eqs. (3.2.17), (3.2.23), and (3.2.25) to Eq. (3.2.27) on $\mathcal{E}_I$ yields

$$\tilde{q}_{ijk}^{\lambda\gamma} = \hat{q}_{ik}^{\lambda\gamma} n_j^{\lambda\gamma} = \hat{q}_{ik}^{\lambda} \left(\frac{1}{2} + \beta_m^{\lambda\gamma} n_m^{\lambda\gamma} \right) n_j^{\lambda\gamma} + \hat{q}_{ik}^{\gamma} \left(\frac{1}{2} - \beta_m^{\lambda\gamma} n_m^{\lambda\gamma} \right) n_j^{\lambda\gamma}. \tag{3.2.33}$$

Note that for Eq. (3.2.33), setting $\beta_m^{\lambda D} = -\frac{1}{2} n_m^{\lambda D}$ on $\mathcal{E}_D$ ensures the Dirichlet condition and any choice of $\beta_m^{\lambda N}$ on $\mathcal{E}_N$ satisfies the Neumann condition. Applying Eqs. (3.2.22), (3.2.24), (3.2.26), and (3.2.25) to Eq. (3.2.28) on $\mathcal{E}_I$ yields

$$\tilde{\mathcal{D}}_{ij}^{\lambda\gamma} = \hat{\mathcal{D}}_{ikj}^{\lambda\gamma} n_k^{\lambda\gamma} = \hat{\mathcal{D}}_{ikj}^{\lambda} \left(\frac{1}{2} - \beta_m^{\lambda\gamma} n_m^{\lambda\gamma} \right) n_k^{\lambda\gamma} + \hat{\mathcal{D}}_{ikj}^{\gamma} \left(\frac{1}{2} + \beta_m^{\lambda\gamma} n_m^{\lambda\gamma} \right) n_k^{\lambda\gamma} - \alpha \left(\hat{q}_{ij}^{\lambda} - \hat{q}_{ij}^{\gamma} \right). \tag{3.2.34}$$

Note that for Eq. (3.2.34), setting $\beta_m^{\lambda N} = +\frac{1}{2} n_m^{\lambda N}$ on $\mathcal{E}_N$ ensures the Neumann condition and any choice of $\beta_m^{\lambda D}$ on $\mathcal{E}_D$ satisfies the Dirichlet condition. These conditions on $\boldsymbol{\beta}$ on boundaries are corroborated in Ref. [21].

The LDG method has been implemented in the WARPXM [73] code. However, as alluded to already, there are alternatives to the LDG formulation as described in Eqs. (3.2.27) and (3.2.28). It has been shown that for elliptic problems the LDG method loses accuracy and can require increased number of iterations due to increasing condition number of the resulting matrix appearing in the equations [55]. A fix for this is the internal penalty flux, which chooses

$$\hat{q}_{ik}^{\lambda\gamma} = \begin{cases} \{\hat{q}_{ik}\}^{\lambda\gamma} & \text{if on } \mathcal{E}_I \\ g_{D_{ik}} & \text{if on } \mathcal{E}_D \\ \hat{q}_{ik}^{\lambda} & \text{if on } \mathcal{E}_N \end{cases} \tag{3.2.35}$$

$$\hat{\mathcal{D}}_{ikj}^{\lambda\gamma} = \begin{cases} \left\{ \hat{\mathcal{D}} \left(\hat{q}_{ij}, \frac{\partial q_i}{\partial x_k} \right) \right\} \Big|_j^{\lambda\gamma} - \alpha [\![\hat{q}_{ij}]\!]_k^{\lambda\gamma} & \text{if on } \mathcal{E}_I \\ \hat{\mathcal{D}} \left(\hat{q}_{ij}, \frac{\partial q_i}{\partial x_k} \right) \Big|_j^{\lambda} - \alpha \left(\hat{q}_{ij}^{\lambda} - g_{D_{ij}} \right) n_k^{\lambda\gamma} & \text{if on } \mathcal{E}_D \\ \mathcal{D}(\hat{q}^{\lambda}, g_N)_{ij} n_k^{\lambda\gamma} (\text{equivalent to } \hat{\mathcal{D}}_{ikj}^{\lambda\gamma} n_k^{\lambda\gamma} = \mathcal{D}(\hat{q}^{\lambda}, g_N)_{ij}) & \text{if on } \mathcal{E}_N. \end{cases} \tag{3.2.36}$$

Note the calculation

$$\left.\frac{\partial q_i}{\partial \xi_k}\right|_j = \left.\frac{\partial}{\partial \xi_k}\right|_j \hat{q}_{il}\psi_l = \hat{q}_{il}\left.\frac{\partial \psi_l}{\partial \xi_k}\right|_j . \tag{3.2.37}$$

Here the j refers to the coordinate location. Let

$$\mathcal{D}_{r_{jkl}} = \left.\frac{\partial \psi_l}{\partial \xi_k}\right|_j . \tag{3.2.38}$$

Note then that

$$\begin{aligned}\mathcal{M}_{ij}\mathcal{D}_{r_{jkl}} =& \sum_{j=1}^{N_p} \int\limits_{D^\lambda} \psi_i\psi_j \left.\frac{\partial \psi_l}{\partial \xi_k}\right|_j d\boldsymbol{\xi} \\ =& \int\limits_{D^\lambda} \psi_i \sum_{j=1}^{N_p} \left.\frac{\partial \psi_l}{\partial \xi_k}\right|_j \psi_j d\boldsymbol{\xi} \\ =& \int\limits_{D^\lambda} \psi_i \frac{\partial \psi_l}{\partial \xi_k} d\boldsymbol{\xi} \\ =& \mathcal{A}_{ikl}\end{aligned} \tag{3.2.39}$$

assuming ψ is a nodal basis function $(q(\xi) = \sum\limits_{i=1}^{N_p} q(\xi_i)\psi_i(\xi))$. Therefore

$$\mathcal{D}_{r_{mkl}} = \mathcal{M}_{mi}^{-1}\mathcal{A}_{ikl} \tag{3.2.40}$$

and application of Eqs. (3.2.37) and (3.2.38) yields

$$\left.\frac{\partial q_i}{\partial \xi_k}\right|_j = \mathcal{D}_{r_{jkl}}\hat{q}_{il}. \tag{3.2.41}$$

This method has also been implemented into the code as an alternative to the LDG fluxes. Note, comparing Eqs. (3.1.28) and (3.2.40)

$$\mathcal{D}_{r_{ijk}} \neq \Upsilon_{ijk}. \tag{3.2.42}$$

With this, one can implement an auxiliary variable for use in the numerical flux as

$$Y_{ijk}^\lambda \equiv \left.\frac{\partial q_i}{\partial x_j}\right|_k^\lambda = \left.\frac{\partial q_i}{\partial \xi_l}\frac{\partial \xi_l}{\partial x_j}\right|_k^\lambda = {J_{lj}^\lambda}^{-1}\mathcal{D}_{r_{klm}}\hat{q}_{im}^\lambda. \tag{3.2.43}$$

The variable, Y_{ijk}^{λ}, could also be used directly as a low-order approximation of the gradient if the complexity of implementing the LDG or IP methods becomes impractical, such as if increasingly higher order derivatives are needed. An example of this is the case of adding hyperviscosity to the Hall MHD model, which is illustrated in Appendix C.

3.3 Limiting

Limiting in the DG scheme has yet to see much development for plasma models, though work has been done for neutral fluid equations (Euler and Navier-Stokes). Thus, a goal of this work will focus on extension of these methods to plasma models, including the MHD and $5N$-moment models. In general, limiting for this method can be classified into 3 types:

1. Slope-Moment
2. Weighted Essentially Non-Oscillatory (WENO)
3. Artificial Viscosity

Brief overviews of these methods can be found in Refs. [41, 81]. In the slope-moment scheme, jumps are controlled by reducing the order of the solution in a given element to some lower order by constraining or nullifying higher-order components, based on some sort of comparison with neighboring elements. These techniques extend from methods used in the finite-volume community in which some form of monotonicity-preserving limiter combined with some smoothness or oscillation detection measures are employed. Examples include a minmod-based TVB limiter [29], moment limiters [64, 11, 15], as well as monotonicity preserving limiters [114, 101]. While these models are widely used and are relatively simple to implement, they tend to degrade the order of accuracy of the solution around the limited discontinuities and can still be difficult to generalize for unstructured grids. Despite this, in the WARPXM code, versions of the slope-moment limiter described by Moe et al. [81] and Tu et al. [122] have been attempted thus far with the aim of extension to plasma models.

The WENO scheme is similar to the moment-slope scheme in that it attempts to redefine the solution on an element in a way that reduces oscillation from jumps. However, in this scheme, a high-order solution in the element is maintained by making reconstructions of the polynomial using different stencils of neighboring elements, each of the same order. A smoothness parameter associated with each stencil is determined and a weighted average of all these stencils determines the reconstructed solution. This method has also been employed in finite-difference and finite-volume methods for shock-capturing. These schemes however, can be computationally expensive to implement, requiring calculations over a stencil for each element for each timestep, and can be difficult to generalize to unstructured, multidimensional meshes, though there have been recent attempts to avoid the non-compactness due to the stencil, with these methods called "Hermite" WENO (HWENO) schemes. A few examples of implementation can be found in Refs. [132, 135, 96, 97, 98, 137, 138, 139, 140, 74, 3, 136].

The artificial viscosity method is to simply add a dissipation term to the otherwise mainly hyperbolic equation set, in order to stabilize shocks. The idea was first tried by von Neumann and Richtmyer during the Manhattan Project [129] and is a common concept in numerical methods for stabilizing systems with shocks. The modern application of this method, as applied to the DG scheme starts with Persson and Peraire [90] with further developments in Refs. [1, 134]. The main drawback with this method is the timestep restriction that occurs when adding viscosity terms ($\Delta t \leq C\,(\Delta x)^2$) at least for explicit timestepping. It also requires a working module for higher order derivatives, as discussed in Sec. 3.2.

3.3.1 Slope-Moment Limiters

In WARPXM, 2 slope-moment limiters have been implemented and will be discussed briefly below.

Moe

This is a slope-moment limiter proposed by Moe et al. [81]. The steps involved in this limiter are

Step 0. Select primitive variables w for bounds checking. Select points j used to approximate element minimums, maximums, and averages. In this work the Gauss-Lobatto points used in the nodal formulation are used.

Step 1. For each element λ and primitive variable component i, compute the minimum and maximum's of each element

$$w^{\lambda}_{\mathrm{M}_i} = \max_{\boldsymbol{x} \in \boldsymbol{x}_j} \{w_{ij}\}, \tag{3.3.1}$$

$$w^{\lambda}_{\mathrm{m}_i} = \min_{\boldsymbol{x} \in \boldsymbol{x}_j} \{w_{ij}\}. \tag{3.3.2}$$

Step 2. For each element λ and each component i, compute the approximate upper and lower bound of the set of element neighbors $\bar{\Gamma}$, to element λ.

$$\mathrm{M}^{\lambda}_{i} = \max \left\{ \bar{w}^{\lambda}_{i} + \alpha(\Delta \boldsymbol{x}), \max_{\gamma \in \bar{\Gamma}} \left\{ w^{\gamma}_{\mathrm{M}_i} \right\} \right\}, \tag{3.3.3}$$

$$\mathrm{m}^{\lambda}_{i} = \max \left\{ \bar{w}^{\lambda}_{i} - \alpha(\Delta \boldsymbol{x}), \max_{\gamma \in \bar{\Gamma}} \left\{ w^{\gamma}_{\mathrm{m}_i} \right\} \right\}, \tag{3.3.4}$$

where $\bar{w}^{\lambda}_{i}$ is the element λ average of w_i. $\alpha(\Delta \boldsymbol{x})$ is a tolerance function defining the aggressiveness of the limiter, and is chosen to be $\mathcal{O}\left(|\Delta x|^{1.5}\right)$.

Step 3. For each element λ, compute

$$\theta^{\lambda}_{M} = \min_{i} \left\{ \phi \left(\frac{M^{\lambda}_{i} - \bar{w}^{\lambda}_{i}}{w^{\lambda}_{\mathrm{M}_i} - \bar{w}^{\lambda}_{i}} \right) \right\}, \tag{3.3.5}$$

$$\theta^{\lambda}_{M} = \min_{i} \left\{ \phi \left(\frac{m^{\lambda}_{i} - \bar{w}^{\lambda}_{i}}{w^{\lambda}_{\mathrm{m}_i} - \bar{w}^{\lambda}_{i}} \right) \right\}. \tag{3.3.6}$$

Step 4. For each element λ compute

$$\theta^{\lambda} = \min \left\{ 1, \theta^{\lambda}_{m}, \theta^{\lambda}_{M} \right\}. \tag{3.3.7}$$

Step 5. Finally, for each element λ, component i, and Gauss-Lobatto point j, conservative variables q^{λ}_{ij} are limited by

$$q^{\lambda}_{ij} = \bar{q}^{\lambda}_{i} + \theta^{\lambda} \left(q^{\lambda}_{ij} - \bar{q}^{\lambda}_{i} \right). \tag{3.3.8}$$

Tu

This is another slope limiter detailed by Tu [122] and described in detail for 2-dimensional triangular elements by Hesthaven [55]. This limiter has been implemented but not extensively tested in this work.

3.3.2 Artificial Viscosity

Compressibility Limiter

In this work, an artificial viscosity limiter based on compressibility has been implemented. For the conservation law, as written in Eq. (3.0.1), a flux term for this limiter has the form

$$\frac{\partial \mathcal{F}_{ij}}{\partial x_j} = -\frac{\partial}{\partial x_j}\left(\epsilon \frac{\partial q_i}{\partial x_j}\right), \tag{3.3.9}$$

where

$$\epsilon =\nu \left|\frac{\partial v_k}{\partial x_k}\right| \tag{3.3.10}$$

for some constant, ν. The velocity divergence is an indication of compressibility, leading to shocks. In this formulation, this will also increase the diffusivity of the limiter to reduce oscillations present due to the shock.

3.4 Time Integration

The time integration of the plasma models is accomplished using implicit and explicit Runge-Kutta methods, including strong stability-preserving Runge Kutta methods [47, 48]. These methods are implemented in a form in which each stage is updated successively over a timestep. The methods solve ordinary differential equations of the form

$$\frac{dq}{dt} =\mathcal{L}(q,t), \tag{3.4.1}$$

where $\mathcal{L}$ is the right-hand-side spatial discretization of the equations as described in Chapter 2. Multiple methods have been employed, including second and third order explicit strong stability preserving Runge Kutta methods [105]. The second-order total variation bounded Runge Kutta method (TVDRK2) [57] is written as

$$q^* = q^n + \Delta t \cdot \mathcal{L}(q^n, t_n), \tag{3.4.2}$$

$$q^{n+1} = \frac{1}{2}q^* + \frac{1}{2}q^n + \frac{1}{2}\Delta t \cdot \mathcal{L}(q^*, t_n + \Delta t). \tag{3.4.3}$$

The third-order total variation diminishing (also known as strong-stability preserving) Runge Kutta method (SSPRK3) is written as

$$q^* = q^n + \Delta t \cdot \mathcal{L}(q^n, t_n), \tag{3.4.4}$$

$$q^{**} = \frac{3}{4}q^n + \frac{1}{4}q^* + \frac{1}{4}\Delta t \cdot \mathcal{L}(q^*, t_n + \Delta t), \tag{3.4.5}$$

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

A fourth order method (RK4) is also used, written as

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

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

$$q^{***} = q^n + \Delta t \cdot \mathcal{L}(q^{**}, t_n + \frac{1}{2}\Delta t), \tag{3.4.9}$$

$$q^{n+1} = \frac{1}{3}\left(-q^n + q^* + 2q^{**} + q^{***}\right) + \frac{1}{6}\Delta t \cdot \mathcal{L}(q^{***}, t_n + \Delta t). \tag{3.4.10}$$

3.5 Boundary Conditions

Boundary conditions can be set for the discontinuous Galerkin method in WARPXM in two ways. The first being to set ghost nodes around boundaries for which simulation variables can be set to values dictated by the boundary condition. Then at these boundaries, the numerical flux is calculated using the internal and ghost values, solving the Riemann problem as described in Sec. 3.1.4. A second way is to impose a flux directly at these boundaries, instead of setting ghost nodes and solving the Riemann problem. Previous students have developed the implementation of the boundary condition using the first method. The second method was implemented for this project, specifically to implement the flux boundary condition for the hybrid model discussed in Sec. 3.7. However, this can also be used to set external boundary conditions, but must be done with care, as shown below.

3.5.1 Comparison of Boundary Conditions of a Free-Slip Wall for Euler Equations

For the implementation of free-slip boundary conditions for the Euler equations, the method is to impose a normal velocity equal to zero at the boundary. This can be done by setting the ghost velocity to the negative of the inside boundary velocity, that is, if $\boldsymbol{Q}_L = [q_0, q_1, q_2, q_3, q_4]^T \equiv [\rho, \rho u, \rho v, \rho w, e]^T$ are the Euler variables at the internal node of a boundary, assuming that u is the normal velocity to the boundary and v and w are tangential velocities, the ghost values can be set by negating u, so that $\boldsymbol{Q}_R = [q_0, -q_1, q_2, q_3, q_4]^T \equiv [\rho, -\rho u, \rho v, \rho w, e]^T$ where a continuous pressure is kept such that $p_L = p_R \equiv p$. Then the Riemann problem can be solved as part of the numerical flux calculation. The alternative way is to set the flux directly as a flux boundary condition. Noticing that the x-direction flux for the Euler equations can be written as

$$\boldsymbol{F}(\boldsymbol{Q})_{\text{Euler}} = \begin{pmatrix} q_1 \\ \frac{q_1^2}{q_0} + p \\ \frac{q_1 q_2}{q_0} \\ \frac{q_1 q_3}{q_0} \\ (q_4 + p)\,\frac{q_1}{q_0} \end{pmatrix}, \tag{3.5.1}$$

it may be tempting to apply, since $q_1 = 0$ at the wall,

$$\boldsymbol{F}^{*}_{\text{unstable direct}} = \begin{pmatrix} 0 \\ p \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{3.5.2}$$

However, comparison with a Riemann solution shows that this would be an incorrect unstable flux. For this case, comparison can be made directly with the Riemann solvers as follows.

Considering the flux of the form in Eq. (3.1.40), the exact solution to the Riemann problem involves understanding the waves propagating from the jump. These could be:

- Rarefaction - Contact Discontinuity - Rarefaction
- Shock - Contact Discontinuity - Rarefaction
- Rarefaction - Contact Discontinuity - Shock
- Shock - Contact Discontinuity - Shock

In the following, just the case of Shock - Contact Discontinuity - Shock case will be considered, where the speed is given by $u > c > 0$ so that the jump decreases the velocity from u to $-u$. The exact solution can be solved following the procedure in Ref. [70], which yields in the general case of 2 intermediate states $\boldsymbol{Q}_L^*$ and $\boldsymbol{Q}_R^*$, where $p_L^* = p_R^* \equiv p^*$ and $u_L^* = u_R^* = u^*$ can be calculated, due to p and u being the Riemann invariants along the contact discontinuity. In this particular case it can also be found $\rho_L^* = \rho_R \equiv \rho^* > \rho_L = \rho_R$, and $e_L^* \equiv e^*$ so $\boldsymbol{Q}_L^* = \boldsymbol{Q}_R^* \equiv \boldsymbol{Q}^*$. Additionally $u^* = 0$. Analysis of the Riemann problem under these conditions yields a left-going shock and a right-going shock (with a stationary contact discontinuity). The left-going shock wave has the Hugoniot shock speed, $\lambda_1 = \frac{\rho_L u_L - \rho^* u^*}{\rho_L - \rho^*} = \frac{\rho u}{\rho - \rho^*} < 0$ with jump $\boldsymbol{\mathcal{W}}_1 = \boldsymbol{Q}^* - \boldsymbol{Q}_L$. The right going wave is the opposite going shock wave with $\lambda_2 = \frac{\rho_R u_R - \rho^* u^*}{\rho_R - \rho^*} = \frac{-\rho u}{\rho - \rho^*} = -\lambda_1 > 0$ with jump $\boldsymbol{\mathcal{W}}_2 = \boldsymbol{Q}_L - \boldsymbol{Q}^*$. Thus application of Eq. (3.1.40) with $A^-\Delta\boldsymbol{Q} = \lambda_1\boldsymbol{\mathcal{W}}_1$ and $A^+\Delta\boldsymbol{Q} = \lambda_2\boldsymbol{\mathcal{W}}_2$ yields

$$\boldsymbol{F}^*_{\text{specific exact SCS Solution}} = \begin{pmatrix} 0 \\ \frac{q_1^2}{q_0} + p + \lambda_1\left(-q_1\right) \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{3.5.3}$$

This is equivalent to seeing the flux as the left-sided flux plus the left-going fluctuations: $\boldsymbol{F}^* = \boldsymbol{F}_L + A^-\Delta\boldsymbol{Q}$. Thus, the true Riemann solution involves not only p but the ram pressure $\frac{q_1^2}{q_0}$ with the addition of what could be thought of a stabilizing term, $-\lambda_1 q_1 > 0$. Thus the flux as given in Eq. (3.5.2) would be incorrect and unstable. An approximate Riemann solver can also be used so as to not have to worry about the exact nature of the Riemann solution (depending on the combination of shocks and rarefactions). For example, following the Roe solver procedure [103], the flux for the free-slip wall is

$$\boldsymbol{F}^*_{\text{Roe Solution}} = \begin{pmatrix} 0 \\ \frac{q_1^2}{q_0} + p + \hat{c} q_1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \tag{3.5.4}$$

where the stabilizing term uses a Roe-averaged sound speed given by $\hat{c}$. A Rusanov flux replaces the stabilizing speed with the maximum speed available in the system, being $|c_{\text{max}} + u_{\text{max}}|$. Thus, unless the true solution of the Riemann problem is known at the boundary, it is generally best to apply boundary conditions by setting appropriate values at ghost nodes. However, the flux boundary condition is useful in the hybrid model for setting fluxes between models at interface boundaries, as is discussed in Sec. 3.7.

3.6 Kinetics Implementation

The continuum kinetic model, as written in Eq. (2.2.7), can also be solved via the DG method as discussed in Sec. 3.1 just as the other equation sets, but has the added complexity of being represented on a separate phase space in up to 6D while other models are represented in up to 3D physical space. Additionally, the kinetic equation must couple with the fields that live in the physical space to calculate the acceleration flux term. One simplifying factor, however, is the velocity space can be developed on a mesh composed of rectilinear hypercubes (lines, rectangles, and rectangular prisms), since velocity is not subject to complex geometries which physical space could be. Thus, the kinetics implementation in this work uses the unstructured mesh for position space variables (fields), a rectilinear velocity space mesh on which to take moments, and a mixed phase space in which the velocity space is stacked onto the physical space, leveraging the unstructured machinery of the code for physical space and the simpler rectilinear mesh of the velocity space to evolve the kinetic equation. The resulting phase space element is a tensor product of these physical and velocity space elements. Only distribution functions f_α are discretized over the entire phase space mesh; electric and magnetic fields are expressed over only the co-located physical space mesh.

Numerical quadrature using Gauss-Lobatto rules in the interval $[0, 1]$ for each dimension is used to compute moments of the distribution function to couple the plasma species with Maxwell's equations, as expressed in Eqs. (2.2.8) and (2.2.9) to obtain charge and current densities, respectively. The integration is performed over all of velocity space at a physical space location,

$$\iiint F(v_x, v_y, v_z) dv_x dv_y dv_z \approx \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} w_i w_j w_k F\left(v_x, v_y, v_z\right) \left|\mathbf{J}(\xi_i, \xi_j, \xi_k)\right|, \tag{3.6.1}$$

where F is the moment function to be integrated, and N is the number of quadrature locations in each dimension, each with weight w. $\boldsymbol{J}$ in this context represents the Jacobian mapping between the isoparametric elements and velocity space. The hypercubes in are implemented as subparametric elements with linearly varying position coordinates and M vertices. Thus, the velocities at quadrature nodes are determined using the basis

$$v_x = P(\xi_i, \xi_j, \xi_k) = \sum_{m=1}^{M} v_{x_m} \psi_m(\xi_1, \xi_2, \xi_3), \tag{3.6.2}$$

$$v_y = Q(\xi_i, \xi_j, \xi_k) = \sum_{m=1}^{M} v_{y_m} \psi_m(\xi_1, \xi_2, \xi_3), \tag{3.6.3}$$

$$v_z = R(\xi_i, \xi_j, \xi_k) = \sum_{m=1}^{M} v_{z_m} \psi_m(\xi_1, \xi_2, \xi_3). \tag{3.6.4}$$

$$\tag{3.6.5}$$

The Jacobian defines the mapping

$$J_{ij} = \frac{\partial v_i}{\partial \xi_j}. \tag{3.6.6}$$

Note that since velocity space is rectilinear, $\boldsymbol{J}$ will be diagonal.

3.6.1 Phase Space Element Construction

The discontinuous Galerkin method for the phase space Vlasov equation is identical to that as described in Sec. 3.1.2, as applied to the fluid equations. The Vlasov equation in some ways is simpler than the fluid equation since it is a single advection equation for the distribution function, where the position-space flux is the velocity coordinate and the velocity-space flux is the acceleration, calculated from the electromagnetic fields. The computational challenge comes from the high dimensionality of this equation (up to 3D3V). In the following sections some details of the implementation of the method for various dimensions are described. To calculate the element basis arrays, tensor products of the physical element basis with lines of the same polynomial order are used (although this does not strictly have to be the case).

Basis Construction

The DG basis arrays for hyperbolic problems are the internal flux array given by Eq. (3.1.28) and the numerical flux array given by Eq. (3.1.29) with the mass matrix and advection matrix defined in Eqs. (3.1.22) and (3.1.23), respectively.

1D1V

At this dimensionality, the line basis in physical space is extended with another line basis for v_x, leading to square elements. Consider a second order line extending to a second order square (corresponding to first order polynomial basis function elements). Here $x_0 = 0$, $x_1 = 1$. Assuming ψ are nodal basis Lagrange polynomials, $\ell_0 = 1 - x$, $\ell_1 = x$. This is now extended in the u direction (x-directed velocity). Table 3.1 shows the points and the basis functions of the square (L_i) as products of the line basis functions. The mass matrix is given by

$$\mathcal{M} = \mathcal{M}_{ij} = \int_{D^\lambda} L_i L_j d\xi = \int_{D^\lambda} \ell_{u_i} \ell_{u_j} du \int_{D^\lambda} \ell_{x_i} \ell_{x_j} dx = \mathcal{M}_{u_i u_j} \mathcal{M}_{x_i x_j} = \mathcal{M}_u \otimes \mathcal{M}_x \tag{3.6.7}$$

Node	Coordinates	Basis L
0	$(0,0)$	$L_0 = \ell_{u_0}\ell_{x_0} = (1-u)(1-x)$
1	$(1,0)$	$L_1 = \ell_{u_0}\ell_{x_1} = (1-u)x$
2	$(0,1)$	$L_2 = \ell_{u_1}\ell_{x_0} = u(1-x)$
3	$(1,1)$	$L_3 = \ell_{u_1}\ell_{x_1} = ux$

Table 3.1: Second order square (first order polynomial) nodes and bases

λ	Constraint	Type	Nodes	$\hat{\boldsymbol{n}}$	γ	Opposite Face Nodes
0	x=0	Line	$\{0,2\}$	$(-1,0)$	1	$\{1,3\}$
1	x=1	Line	$\{1,3\}$	$(+1,0)$	0	$\{0,2\}$
2	u=0	Line	$\{0,1\}$	$(0,-1)$	3	$\{2,3\}$
3	u=1	Line	$\{2,3\}$	$(0,+1)$	2	$\{0,1\}$

Table 3.2: Table of face cells of second order square

since the x and u directions are independent. Notice that indices u_i are distinct from x_i for example. By the property of tensor products

$$\mathcal{M}^{-1} = \mathcal{M}_u^{-1} \otimes \mathcal{M}_x^{-1}. \tag{3.6.8}$$

Now the advection matrix, as defined in Eq. (3.1.23) is

$$\begin{aligned}
\mathcal{A} = \mathcal{A}_{ijk} = \int_{D^\lambda} L_i \partial_j L_k d\xi = \begin{bmatrix} \mathcal{A}_{i0k} \\ \mathcal{A}_{i1k} \end{bmatrix} = \int_{D^\lambda} \begin{bmatrix} L_i \partial_x L_k \\ L_i \partial_u L_k \end{bmatrix} d\xi &= \int_{D^\lambda} \begin{bmatrix} \ell_{u_i} \ell_{x_i} \partial_x \left(\ell_{u_k} \ell_{x_k} \right) \\ \ell_{u_i} \ell_{x_i} \partial_u \left(\ell_{u_k} \ell_{x_k} \right) \end{bmatrix} dxdu \\
&= \int_{D^\lambda} \begin{bmatrix} \ell_{u_i} \ell_{u_k} \ell_{x_i} \partial_x \left(\ell_{x_k} \right) \\ \ell_{u_i} \partial_u \left(\ell_{u_k} \right) \ell_{x_i} \ell_{x_k} \end{bmatrix} dxdu \\
&= \begin{bmatrix} \mathcal{M}_{u_i u_k} \mathcal{A}_{x_{x_i x_k}} \\ \mathcal{A}_{u_{u_i u_k}} \mathcal{M}_{x_i x_k} \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{M}_u \otimes \mathcal{A}_x \\ \mathcal{A}_u \otimes \mathcal{M}_x \end{bmatrix},
\end{aligned} \tag{3.6.9}$$

where the index j denoting dimension is written here as first index of the matrix. The Υ array can then be written

$$\begin{aligned}
\Upsilon = \Upsilon_{ijk} = \mathcal{M}_{il}^{-1} \mathcal{A}_{kjl} = \mathcal{M}_{il}^{-1} \begin{bmatrix} \mathcal{A}_{k0l} \\ \mathcal{A}_{k1l} \end{bmatrix} = \mathcal{M}_{u_i u_l}^{-1} \mathcal{M}_{x_i x_l}^{-1} \begin{bmatrix} \mathcal{M}_{u_k u_l} \mathcal{A}_{x_{x_k x_l}} \\ \mathcal{A}_{u_{u_k u_l}} \mathcal{M}_{x_k x_l} \end{bmatrix} &= \begin{bmatrix} \mathcal{M}_{u_i u_l}^{-1} \mathcal{M}_{x_i x_l}^{-1} \mathcal{M}_{u_k u_l} \mathcal{A}_{x_{x_k x_l}} \\ \mathcal{M}_{u_i u_l}^{-1} \mathcal{M}_{x_i x_l}^{-1} \mathcal{A}_{u_{u_k u_l}} \mathcal{M}_{x_k x_l} \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{M}_{u_i u_l}^{-1} \mathcal{M}_{u_k u_l} \mathcal{M}_{x_i x_l}^{-1} \mathcal{A}_{x_{x_k x_l}} \\ \mathcal{M}_{u_i u_l}^{-1} \mathcal{A}_{u_{u_k u_l}} \mathcal{M}_{x_i x_l}^{-1} \mathcal{M}_{x_k x_l} \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{M}_{u_i u_l}^{-1} \mathcal{M}_{u_l u_k} \Upsilon_{x_{x_i x_k}} \\ \Upsilon_{u_{u_i u_k}} \mathcal{M}_{x_i x_l}^{-1} \mathcal{M}_{x_l x_k} \end{bmatrix} \\
&= \begin{bmatrix} \delta_{u_i u_k} \Upsilon_{x_{x_i x_k}} \\ \Upsilon_{u_{u_i u_k}} \delta_{x_i x_k} \end{bmatrix} \\
&= \begin{bmatrix} I_u \otimes \Upsilon_x \\ \Upsilon_u \otimes I_x \end{bmatrix},
\end{aligned} \tag{3.6.10}$$

where Eq. (3.1.28) and the symmetry of the mass matrix are used. Again, the dimension index is written first here. A similar procedure can be performed for the numerical flux basis, leading to

$$\Gamma^{\lambda\gamma} = \Gamma^{\lambda\gamma}_{ij} = \begin{bmatrix} \Gamma^{\gamma=x^0}_{ij} \\ \Gamma^{\gamma=x^1}_{ij} \\ \Gamma^{\gamma=u^0}_{ij} \\ \Gamma^{\gamma=u^1}_{ij} \end{bmatrix} = \begin{bmatrix} \int_0^1 \int_0^1 \ell_{u_i}\ell_{x_i}\ell_{u_j}\ell_{x_j}\delta(x-0)dxdu \\ \int_0^1 \int_0^1 \ell_{u_i}\ell_{x_i}\ell_{u_j}\ell_{x_j}\delta(x-1)dxdu \\ \int_0^1 \int_0^1 \ell_{u_i}\ell_{x_i}\ell_{u_j}\ell_{x_j}\delta(u-0)dxdu \\ \int_0^1 \int_0^1 \ell_{u_i}\ell_{x_i}\ell_{u_j}\ell_{x_j}\delta(u-1)dxdu \end{bmatrix} = \begin{bmatrix} \mathcal{M}_{u_iu_j}\Gamma^{\gamma=x^0}_{x_ix_j} \\ \mathcal{M}_{u_iu_j}\Gamma^{\gamma=x^1}_{x_ix_j} \\ \Gamma^{\gamma=u^0}_{u_iu_j}\mathcal{M}_{x_ix_j} \\ \Gamma^{\gamma=u^1}_{u_iu_j}\mathcal{M}_{x_ix_j} \end{bmatrix} = \begin{bmatrix} \mathcal{M}_u \otimes \Gamma^{\gamma=x^0}_x \\ \mathcal{M}_u \otimes \Gamma^{\gamma=x^1}_x \\ \Gamma^{\gamma=u^0}_u \otimes \mathcal{M}_x \\ \Gamma^{\gamma=u^1}_u \otimes \mathcal{M}_x \end{bmatrix}, \tag{3.6.11}$$

where it is now written in terms of the face as the first dimension. Then the accompanying basis array can be written

$$\Xi^{\lambda\gamma} = \Xi^{\lambda\gamma}_{ij} = \mathcal{M}^{-1}_{il}\Gamma^{\lambda\gamma}_{lj} = \mathcal{M}^{-1}_{il} \begin{bmatrix} \Gamma^{\gamma=x^0}_{lj} \\ \Gamma^{\gamma=x^1}_{lj} \\ \Gamma^{\gamma=u^0}_{lj} \\ \Gamma^{\gamma=u^1}_{lj} \end{bmatrix} = \begin{bmatrix} \mathcal{M}^{-1}_{u_iu_l}\mathcal{M}^{-1}_{x_ix_l}\mathcal{M}_{u_lu_j}\Gamma^{\gamma=x^0}_{x_lx_j} \\ \mathcal{M}^{-1}_{u_iu_l}\mathcal{M}^{-1}_{x_ix_l}\mathcal{M}_{u_lu_j}\Gamma^{\gamma=x^1}_{x_lx_j} \\ \mathcal{M}^{-1}_{u_iu_l}\mathcal{M}^{-1}_{x_ix_l}\Gamma^{\gamma=u^0}_{u_lu_j}\mathcal{M}_{x_lx_j} \\ \mathcal{M}^{-1}_{u_iu_l}\mathcal{M}^{-1}_{x_ix_l}\Gamma^{\gamma=u^1}_{u_lu_j}\mathcal{M}_{x_lx_j} \end{bmatrix} = \begin{bmatrix} \delta_{u_iu_j}\Xi^{\gamma=x^0}_{x_ix_j} \\ \delta_{u_iu_j}\Xi^{\gamma=x^1}_{x_ix_j} \\ \Xi^{\gamma=u^0}_{u_iu_j}\delta_{x_ix_j} \\ \Xi^{\gamma=u^1}_{u_iu_j}\delta_{x_ix_j} \end{bmatrix}$$
$$= \begin{bmatrix} I_u \otimes \Xi^{\gamma=x^0}_x \\ I_u \otimes \Xi^{\gamma=x^1}_x \\ \Xi^{\gamma=u^0}_u \otimes I_x \\ \Xi^{\gamma=u^1}_u \otimes I_x \end{bmatrix}. \tag{3.6.12}$$

With these tensor product formulas, the basis of squares can be formulated from any basis order of lines. WARPXM currently supports phase space square elements up through third order.

1D2V

At this dimensionality the line basis in physical space is extended with 2 line bases, for velocities u and v leading to a cube element. Once the relevant arrays of the line basis

Node	Coordinates	Basis L
0	$(0,0,0)$	$L_0 = \ell_{v_0}\ell_{u_0}\ell_{x_0} = (1-v)(1-u)(1-x)$
1	$(1,0,0)$	$L_1 = \ell_{v_0}\ell_{u_0}\ell_{x_1} = (1-v)(1-u)x$
2	$(0,1,0)$	$L_2 = \ell_{v_0}\ell_{u_1}\ell_{x_0} = (1-v)u(1-x)$
3	$(1,1,0)$	$L_3 = \ell_{v_0}\ell_{u_1}\ell_{x_1} = (1-v)ux$
4	$(0,0,1)$	$L_4 = \ell_{v_1}\ell_{u_0}\ell_{x_0} = v(1-u)(1-x)$
5	$(1,0,1)$	$L_5 = \ell_{v_1}\ell_{u_0}\ell_{x_1} = v(1-u)x$
6	$(0,1,1)$	$L_6 = \ell_{v_1}\ell_{u_1}\ell_{x_0} = vu(1-x)$
7	$(1,1,1)$	$L_7 = \ell_{v_1}\ell_{u_1}\ell_{x_1} = vux$

Table 3.3: Second order cube (first order polynomial) nodes and bases

λ	Constraint	Type	Nodes	$\hat{\boldsymbol{n}}$	γ	Opposite Face Nodes
0	x=0	Square	$\{0,2,4,6\}$	$(-1,0,0)$	1	$\{1,3,5,7\}$
1	x=1	Square	$\{1,3,5,7\}$	$(+1,0,0)$	0	$\{0,2,4,6\}$
2	u=0	Square	$\{0,1,4,5\}$	$(0,-1,0)$	3	$\{2,3,6,7\}$
3	u=1	Square	$\{2,3,6,7\}$	$(0,+1,0)$	2	$\{0,1,4,5\}$
4	v=0	Square	$\{0,1,2,3\}$	$(0,0,-1)$	5	$\{4,5,6,7\}$
5	v=1	Square	$\{4,5,6,7\}$	$(0,0,+1)$	4	$\{0,1,2,3\}$

Table 3.4: Table of face cells of second order cube

have been calculated, tensor product calculations can be used to generate the cube basis for various orders. These can be constructed by the same approach as shown for the 1D1V element, extending to the second velocity dimension. Thus these become

$$\mathcal{M} = \mathcal{M}_v \otimes \mathcal{M}_u \otimes \mathcal{M}_x, \tag{3.6.13}$$

$$\mathcal{M}^{-1} = \mathcal{M}_v^{-1} \otimes \mathcal{M}_u^{-1} \otimes \mathcal{M}_x^{-1}, \tag{3.6.14}$$

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_{i0k} \\ \mathcal{A}_{i1k} \\ \mathcal{A}_{i2k} \end{bmatrix} = \begin{bmatrix} \mathcal{M}_v \otimes \mathcal{M}_u \otimes \mathcal{A}_x \\ \mathcal{M}_v \otimes \mathcal{A}_u \otimes \mathcal{M}_x \\ \mathcal{A}_v \otimes \mathcal{M}_u \otimes \mathcal{M}_x \end{bmatrix}, \tag{3.6.15}$$

$$\Upsilon = \begin{bmatrix} \Upsilon_{i0k} \\ \Upsilon_{i1k} \\ \Upsilon_{i2k} \end{bmatrix} = \begin{bmatrix} I_v \otimes I_u \otimes \Upsilon_x \\ I_v \otimes \Upsilon_u \otimes I_x \\ \Upsilon_v \otimes I_u \otimes I_x \end{bmatrix}, \tag{3.6.16}$$

$$\Gamma^{\lambda\gamma} = \begin{bmatrix} \Gamma_{ij}^{\gamma=x^0} \\ \Gamma_{ij}^{\gamma=x^1} \\ \Gamma_{ij}^{\gamma=u^0} \\ \Gamma_{ij}^{\gamma=u^1} \\ \Gamma_{ij}^{\gamma=v^0} \\ \Gamma_{ij}^{\gamma=v^1} \end{bmatrix} = \begin{bmatrix} \mathcal{M}_v \otimes \mathcal{M}_u \otimes \Gamma_x^{\gamma=x^0} \\ \mathcal{M}_v \otimes \mathcal{M}_u \otimes \Gamma_x^{\gamma=x^1} \\ \mathcal{M}_v \otimes \Gamma_u^{\gamma=u^0} \otimes \mathcal{M}_x \\ \mathcal{M}_v \otimes \Gamma_u^{\gamma=u^1} \otimes \mathcal{M}_x \\ \Gamma_v^{\gamma=v^0} \otimes \mathcal{M}_u \otimes \mathcal{M}_x \\ \Gamma_v^{\gamma=v^1} \otimes \mathcal{M}_u \otimes \mathcal{M}_x \end{bmatrix}, \tag{3.6.17}$$

$$\Xi^{\lambda\gamma} = \begin{bmatrix} \Xi_{ij}^{\gamma=x^0} \\ \Xi_{ij}^{\gamma=x^1} \\ \Xi_{ij}^{\gamma=u^0} \\ \Xi_{ij}^{\gamma=u^1} \\ \Xi_{ij}^{\gamma=v^0} \\ \Xi_{ij}^{\gamma=v^1} \end{bmatrix} = \begin{bmatrix} I_v \otimes I_u \otimes \Xi_x^{\gamma=x^0} \\ I_v \otimes I_u \otimes \Xi_x^{\gamma=x^1} \\ I_v \otimes \Xi_u^{\gamma=u^0} \otimes I_x \\ I_v \otimes \Xi_u^{\gamma=u^1} \otimes I_x \\ \Xi_v^{\gamma=v^0} \otimes I_u \otimes I_x \\ \Xi_v^{\gamma=v^1} \otimes I_u \otimes I_x \end{bmatrix}. \tag{3.6.18}$$

WARPXM currently supports phase space cubes elements up through fifth order.

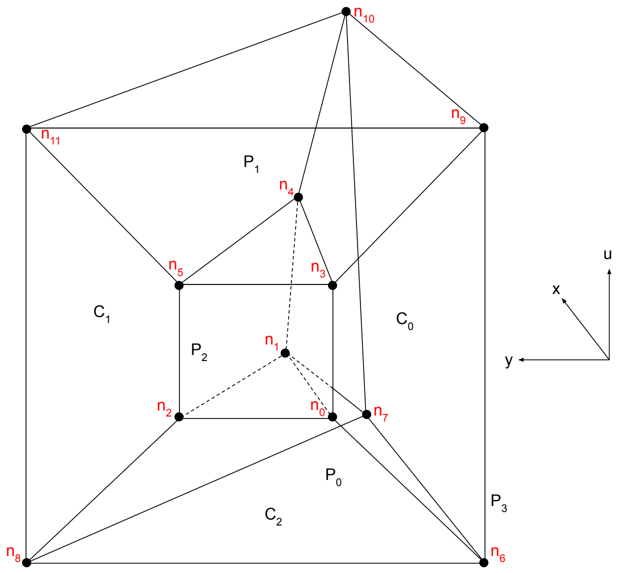


Figure 3.2: Second order (first order polynomial) 2D2V 3-4 duoprism element.

2D2V

In 2D2V, a structured code might extend two line bases on squares, leading to a four-dimensional hypercube. However, in this work the 2D elements are triangles used on unstructured physical meshes. Thus, the extension of line bases in u and v velocity directions leads to the 3-4 duoprism four-dimensional phase space element [35]. Figure 3.2 is a Schlegel diagram of a 3-4 duoprism with nodes shown for a second order (first order polynomial) element. Table 3.5 gives the locations of the nodes and the basis functions while Table 3.6 gives its face information.

With the unstructured mesh, opposite face and face nodes required in the numerical flux calculations for faces with physical space normals will not be as obvious as with elements emanating from lines. For any particular triangular face, the opposite face could be of any triangular face, and the nodes could be oriented in the same or opposite direction. This information must be taken from the physical space DG calculation to determine the opposite face and orientation, which can be used to stack up the nodes on the opposing duoprism

Node	Coordinates	Basis L
0	$(0,0,0,0)$	$L_0 = \ell_{v_0}\ell_{u_0}\ell_{\Delta_0} = (1-v)(1-u)(1-x-y)$
1	$(1,0,0,0)$	$L_1 = \ell_{v_0}\ell_{u_0}\ell_{\Delta_1} = (1-v)(1-u)x$
2	$(0,1,0,0)$	$L_2 = \ell_{v_0}\ell_{u_0}\ell_{\Delta_2} = (1-v)(1-u)y$
3	$(0,0,1,0)$	$L_3 = \ell_{v_0}\ell_{u_1}\ell_{\Delta_0} = (1-v)u(1-x-y)$
4	$(1,0,1,0)$	$L_4 = \ell_{v_0}\ell_{u_1}\ell_{\Delta_1} = (1-v)ux$
5	$(0,1,1,0)$	$L_5 = \ell_{v_0}\ell_{u_1}\ell_{\Delta_2} = (1-v)uy$
6	$(0,0,0,1)$	$L_6 = \ell_{v_1}\ell_{u_0}\ell_{\Delta_0} = v(1-u)(1-x-y)$
7	$(1,0,0,1)$	$L_7 = \ell_{v_1}\ell_{u_0}\ell_{\Delta_1} = v(1-u)x$
8	$(0,1,0,1)$	$L_8 = \ell_{v_1}\ell_{u_0}\ell_{\Delta_2} = v(1-u)y$
9	$(0,0,1,1)$	$L_9 = \ell_{v_1}\ell_{u_1}\ell_{\Delta_0} = vu(1-x-y)$
10	$(1,0,1,1)$	$L_{10} = \ell_{v_1}\ell_{u_1}\ell_{\Delta_1} = vux$
11	$(0,1,1,1)$	$L_{11} = \ell_{v_1}\ell_{u_1}\ell_{\Delta_2} = vuy$

Table 3.5: Second order duoprism nodes and bases

λ	Constraint	Type	Nodes	$\hat{\boldsymbol{n}}$	γ	Opposite Face Nodes
0	y=0	Cube	$\{0,1,3,4,6,7,9,10\}$	$(0,-1,0,0)$	Depends	Depends
1	x+y=1	Cube	$\{1,2,4,5,7,8,10,11\}$	$(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}},0,0)$	Depends	Depends
2	x=0	Cube	$\{2,0,5,3,8,6,11,9\}$	$(-1,0,0,0)$	Depends	Depends
3	u=0	Prism	$\{0,1,2,6,7,8\}$	$(0,0,-1,0)$	4	$\{3,4,5,9,10,11\}$
4	u=1	Prism	$\{3,4,5,9,10,11\}$	$(0,0,+1,0)$	3	$\{0,1,2,6,7,8\}$
5	v=0	Prism	$\{0,1,2,3,4,5\}$	$(0,0,0,-1)$	6	$\{6,7,8,9,10,11\}$
6	v=1	Prism	$\{6,7,8,9,10,11\}$	$(0,0,0,+1)$	5	$\{0,1,2,3,4,5\}$

Table 3.6: Table of face cells of second order duoprism

λ	γ	Orientation	λ nodes	γ nodes
Δ^0	Δ^0	A	$\{0,1,3,4,6,7,9,10\}$	$\{0,1,3,4,6,7,9,10\}$
Δ^0	Δ^0	B	$\{0,1,3,4,6,7,9,10\}$	$\{1,0,4,3,7,6,10,9\}$
Δ^0	Δ^1	A	$\{0,1,3,4,6,7,9,10\}$	$\{1,2,4,5,7,8,10,11\}$
Δ^0	Δ^1	B	$\{0,1,3,4,6,7,9,10\}$	$\{2,1,5,4,8,7,11,10\}$
Δ^0	Δ^2	A	$\{0,1,3,4,6,7,9,10\}$	$\{2,0,5,3,8,6,11,9\}$
Δ^0	Δ^2	B	$\{0,1,3,4,6,7,9,10\}$	$\{0,2,3,5,6,8,9,11\}$
Δ^1	Δ^0	A	$\{1,2,4,5,7,8,10,11\}$	$\{0,1,3,4,6,7,9,10\}$
Δ^1	Δ^0	B	$\{1,2,4,5,7,8,10,11\}$	$\{1,0,4,3,7,6,10,9\}$
Δ^1	Δ^1	A	$\{1,2,4,5,7,8,10,11\}$	$\{1,2,4,5,7,8,10,11\}$
Δ^1	Δ^1	B	$\{1,2,4,5,7,8,10,11\}$	$\{2,1,5,4,8,7,11,10\}$
Δ^1	Δ^2	A	$\{1,2,4,5,7,8,10,11\}$	$\{2,0,5,3,8,6,11,9\}$
Δ^1	Δ^2	B	$\{1,2,4,5,7,8,10,11\}$	$\{0,2,3,5,6,8,9,11\}$
Δ^2	Δ^0	A	$\{2,0,5,3,8,6,11,9\}$	$\{0,1,3,4,6,7,9,10\}$
Δ^2	Δ^0	B	$\{2,0,5,3,8,6,11,9\}$	$\{1,0,4,3,7,6,10,9\}$
Δ^2	Δ^1	A	$\{2,0,5,3,8,6,11,9\}$	$\{1,2,4,5,7,8,10,11\}$
Δ^2	Δ^1	B	$\{2,0,5,3,8,6,11,9\}$	$\{2,1,5,4,8,7,11,10\}$
Δ^2	Δ^2	A	$\{2,0,5,3,8,6,11,9\}$	$\{2,0,5,3,8,6,11,9\}$
Δ^2	Δ^2	B	$\{2,0,5,3,8,6,11,9\}$	$\{0,2,3,5,6,8,9,11\}$

Table 3.7: Opposing face nodes for the cube faces of a second order duoprism emanating from triangles (spatial-flux faces). λ corresponds to the face in the element (0, 1, 2), and γ corresponds to the opposing face in the neighbor element (0, 1, 2). There are two orientations of the adjoining face.

face. For second order, these cases are given in Table 3.7.

The bases are calculated using tensor products of the lines in velocity space with the physical space triangular element in the same manner as the 1D1V and 1D2V calculations on lines, since each dimension is still independent. This leads to

$$\mathcal{M} = \mathcal{M}_v \otimes \mathcal{M}_u \otimes \mathcal{M}_\Delta, \tag{3.6.19}$$

$$\mathcal{M}^{-1} = \mathcal{M}_v^{-1} \otimes \mathcal{M}_u^{-1} \otimes \mathcal{M}_\Delta^{-1}, \tag{3.6.20}$$

$$\mathcal{A} = \begin{bmatrix} \mathcal{A}_{i0k} \\ \mathcal{A}_{i1k} \\ \mathcal{A}_{i2k} \\ \mathcal{A}_{i3k} \end{bmatrix} = \begin{bmatrix} \mathcal{M}_v \otimes \mathcal{M}_u \otimes \mathcal{A}_{\Delta,x} \\ \mathcal{M}_v \otimes \mathcal{M}_u \otimes \mathcal{A}_{\Delta,y} \\ \mathcal{M}_v \otimes \mathcal{A}_u \otimes \mathcal{M}_\Delta \\ \mathcal{A}_v \otimes \mathcal{M}_u \otimes \mathcal{M}_\Delta \end{bmatrix}, \tag{3.6.21}$$

$$\Upsilon = \begin{bmatrix} \Upsilon_{i0k} \\ \Upsilon_{i1k} \\ \Upsilon_{i2k} \\ \Upsilon_{i3k} \end{bmatrix} = \begin{bmatrix} I_v \otimes I_u \otimes \Upsilon_{\Delta,x} \\ I_v \otimes I_u \otimes \Upsilon_{\Delta,y} \\ I_v \otimes \Upsilon_u \otimes I_\Delta \\ \Upsilon_v \otimes I_u \otimes I_\Delta \end{bmatrix}, \tag{3.6.22}$$

$$\Gamma^{\lambda\gamma} = \begin{bmatrix} \Gamma_{ij}^{\gamma=\Delta^0} \\ \Gamma_{ij}^{\gamma=\Delta^1} \\ \Gamma_{ij}^{\gamma=\Delta^2} \\ \Gamma_{ij}^{\gamma=u^0} \\ \Gamma_{ij}^{\gamma=u^1} \\ \Gamma_{ij}^{\gamma=v^0} \\ \Gamma_{ij}^{\gamma=v^1} \end{bmatrix} = \begin{bmatrix} \mathcal{M}_v \otimes \mathcal{M}_u \otimes \Gamma_\Delta^{\gamma=\Delta^0} \\ \mathcal{M}_v \otimes \mathcal{M}_u \otimes \Gamma_\Delta^{\gamma=\Delta^1} \\ \mathcal{M}_v \otimes \mathcal{M}_u \otimes \Gamma_\Delta^{\gamma=\Delta^2} \\ \mathcal{M}_v \otimes \Gamma_u^{\gamma=u^0} \otimes \mathcal{M}_\Delta \\ \mathcal{M}_v \otimes \Gamma_u^{\gamma=u^1} \otimes \mathcal{M}_\Delta \\ \Gamma_v^{\gamma=v^0} \otimes \mathcal{M}_u \otimes \mathcal{M}_\Delta \\ \Gamma_v^{\gamma=v^1} \otimes \mathcal{M}_u \otimes \mathcal{M}_\Delta \end{bmatrix}, \tag{3.6.23}$$

$$\Xi^{\lambda\gamma} = \begin{bmatrix} \Xi_{ij}^{\gamma=\Delta^0} \\ \Xi_{ij}^{\gamma=\Delta^1} \\ \Xi_{ij}^{\gamma=\Delta^2} \\ \Xi_{ij}^{\gamma=u^0} \\ \Xi_{ij}^{\gamma=u^1} \\ \Xi_{ij}^{\gamma=v^0} \\ \Xi_{ij}^{\gamma=v^1} \end{bmatrix} = \begin{bmatrix} I_v \otimes I_u \otimes \Xi_{\Delta}^{\gamma=\Delta^0} \\ I_v \otimes I_u \otimes \Xi_{\Delta}^{\gamma=\Delta^1} \\ I_v \otimes I_u \otimes \Xi_{\Delta}^{\gamma=\Delta^2} \\ I_v \otimes \Xi_{u}^{\gamma=u^0} \otimes I_\Delta \\ I_v \otimes \Xi_{u}^{\gamma=u^1} \otimes I_\Delta \\ \Xi_{v}^{\gamma=v^0} \otimes I_u \otimes I_\Delta \\ \Xi_{v}^{\gamma=v^1} \otimes I_u \otimes I_\Delta \end{bmatrix}. \tag{3.6.24}$$

WARPXM currently supports phase space 3-4 duoprism elements up through fifth order. Note that that for higher dimensional elements such as these, one finds that even at 2nd order, the basis arrays can become very sparse. Consider this second order duoprism constructed second order triangles and lines. In this case, Υ for the triangle in position space is

$$\Upsilon_\Delta = \begin{bmatrix} \Upsilon_{\Delta,x} \\ \Upsilon_{\Delta,y} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -4 & -4 & -4 \\ 4 & 4 & 4 \\ 0 & 0 & 0 \end{bmatrix} \\ \begin{bmatrix} -4 & -4 & -4 \\ 0 & 0 & 0 \\ 4 & 4 & 4 \end{bmatrix} \end{bmatrix} \tag{3.6.25}$$

and for the lines in velocity space is

$$\Upsilon_{u,v} = \begin{bmatrix} \Upsilon_{|,u,v} \end{bmatrix} = \begin{bmatrix} -3 & -3 \\ 3 & 3 \end{bmatrix}. \tag{3.6.26}$$

Application of Eq. (3.6.22) to evaluate Υ for the second order duoprism leads to

$$\Upsilon_{i0k} = \begin{bmatrix} -4 & -4 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 & -4 & -4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -4 & -4 & -4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & -4 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.6.27}$$

$$\Upsilon_{i1k} = \begin{bmatrix} -4 & -4 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -4 & -4 & -4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -4 & -4 & -4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & -4 & -4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 4 \end{bmatrix}, \tag{3.6.28}$$

$$\Upsilon_{i2k} = \begin{bmatrix} -3 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -3 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & -3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 3 \end{bmatrix}, \tag{3.6.29}$$

$$\Upsilon_{i3k} = \begin{bmatrix} -3 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 \\ 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 \\ 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & -3 \\ 3 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}. \tag{3.6.30}$$

Notice the sparsity of these arrays. Similarly sparse arrays appear for the numerical flux basis Ξ. In applying the calculation of the right hand side in the DG update as written in Eq. (3.1.37) for the Vlasov equation, one can see this will result in many unnecessary calculations. To avoid this, for this type of element, these bases can be held in a sparse format. For this work, a compressed sparse row (CSR) format is chosen [24], representing the each 2D matrix by three arrays:

1. Val is the value array which holds the non-zero values, written row by row, left column to right column.

2. Col is the array containing the column index of the non-zero values in the Val array.

3. RowPtr is the array containing the row-index range of the non-zero elements. An easy way to calculate this is to consider each index of this array to correspond to a row, for which one fills in the number of non-zero elements in the matrix through the row before it. Sometimes a last index will be added to account for the last row.

Then an algorithm to do the matrix-vector multiplication for a matrix stored in CSR format with N rows with vector d, storing into a previously-zero'd array `result` is

```
for (i = 0; i < N; i++)
{
  for (k = RowPtr[i]; k < RowPtr[i+1]; k++)
  {
    result[i] += Val[k]*d[Col[k]];
  }
}
```

:

For example, Υ_{i0k} in Eq. (3.6.27) in CSR format is given by

$$\mathrm{Val}_{\Upsilon_{i0k}} = [-4,-4,-4,4,4,4,-4,-4,-4,4,4,4,-4,-4,-4,4,4,4,-4,-4,-4,4,4,4]\,, \tag{3.6.31}$$

$$\mathrm{Col}_{\Upsilon_{i0k}} = [0,1,2,0,1,2,3,4,5,3,4,5,6,7,8,6,7,8,9,10,11,9,10,11]\,, \tag{3.6.32}$$

$$\text{RowPtr}_{\Upsilon_{i0k}} = [0,3,6,6,9,12,12,15,18,18,21,24,24]\,. \tag{3.6.33}$$

Then each l component of Υ_{jlk} in Eq. (3.1.37) held in this format can be multiplied by the appropriate column of the matrix $\Theta_{ilk} \equiv {J^{\lambda}_{lm}}^{-1} \hat{\mathcal{F}}^{\lambda}_{imk}$ for each variable component i. A similar procedure can be performed for the numerical flux term.

3.6.2 Timestepping

Timestepping for the kinetic model uses the same explicit RK scheme used for all other models. The timestep is defined by the Courant-Friedrichs-Lewy (CFL) condition [34] for hyperbolic problems in one dimension

$$\Delta t \leq \frac{\mathrm{C}h}{a}, \tag{3.6.34}$$

where C is the dimensionless Courant number, h is a cell-spacing length, and a is the maximum wavespeed of the system. For the multidimensional phase space problems in this work, the following is adopted for the allowed timestep

$$\Delta t \leq \mathrm{C} \left[\sum_{d=1}^{D} \frac{a_d}{h_d} \right]^{-1} \tag{3.6.35}$$

for D phase space dimensions where a_d is the maximum advection speed in the d-th direction and h_d is a measure of the cell spacing in the d-th direction [128]. Courant numbers are set based on those defined for various combinations of RK scheme and basis order in [33].

3.7 Hybridization

The domain-decomposed hybrid method as described in Chapter 2 is performed by subdivision of the simulation domain into subdomains, where different plasma models are solved. Interface conditions are applied at locations where the subdomains meet. In this work, implementation of the consistent direct variable translation method as described in Sec. 2.4.1 and the conservative flux matching method as described in Sec. 2.4.2 have been performed,

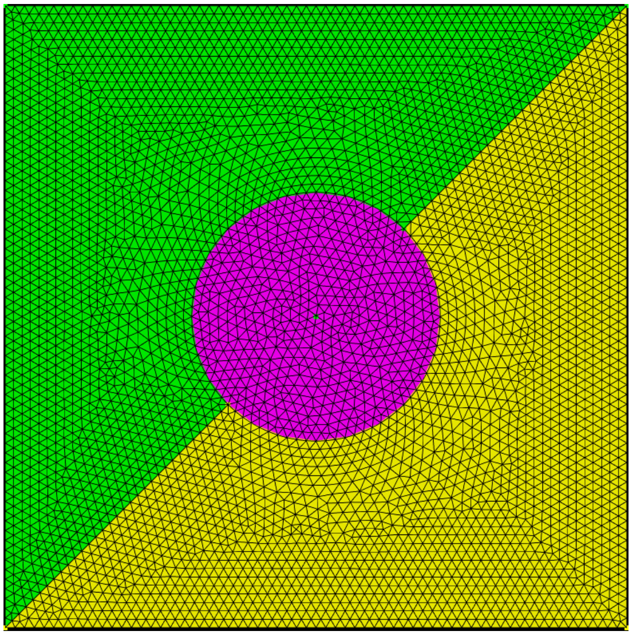


Figure 3.3: Mesh with 3 separate subdomains.

specifically handling the interface conditions between the $5N$-moment and continuum kinetic models. For the case of the conservative flux matching method, the application for the coupling between the fluid and kinetic models is named the composite distribution function method, which is explained below, where details of the implementation for each method are described.

3.7.1 Overview of the domain-decomposed hybrid method for the $5N$-moment and continuum kinetic model

The domain-decomposed hybrid method applies the $5N$-moment model (also referred to as the multi-fluid plasma model) model and multi-species continuum kinetic model on adjacent subdomains of a simulation. At subdomain interfaces the distribution function solutions for each model can be different, as illustrated in Fig. 3.4, where f_F refers to the Maxwellian distribution function constructed from fluid variable solutions to the multi-fluid plasma model

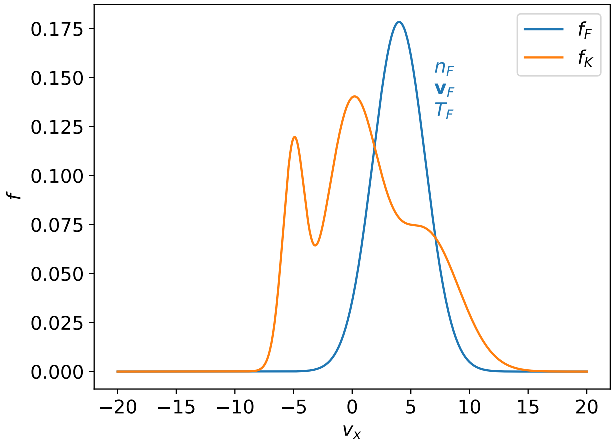


Figure 3.4: Illustrative distribution functions at a subdomain interface between the multi-fluid plasma model, f_F, and the multi-species kinetic model, f_K. The fluid distribution function is a Maxwellian calculated using Eq. (2.2.14) from the fluid variables n_F, $\boldsymbol{v}_F$, and T_F, which are the solutions to the multi-fluid plasma model at the subdomain interface. Solutions for only a single species are shown for illustration, but solutions for multiple species can also exist. The differences between f_F and f_K are also exaggerated for illustrative purposes. In practice, f_F and f_K will have only small differences.

using Eq. (2.2.14) and f_K refers to the distribution function solution to the multi-species kinetic model. Interface conditions determine the interaction between the multi-fluid plasma model and multi-species kinetic model at the subdomain interfaces, which can be specified through the surface numerical flux as described in Sec. 3.1.4. Two approaches are considered to define the numerical fluxes at the subdomain interfaces.

The first approach defines the numerical flux in a manner that is consistent with the assumptions associated with each model, e.g. the distributions used to calculate the numerical flux for the multi-fluid plasma model are Maxwellian. The method achieves the consistency for the multi-fluid plasma model by extracting variables from f_K that define a Maxwellian distribution function with the same first three velocity moments as f_K to enable the flux splitting of the fluid model using a Riemann solver as described for the fluid model in Sec. 3.1.4. The numerical flux for the multi-species kinetic model is calculated from f_F and f_K using Eq. (3.1.46a), which does not assume any particular distribution function profile. Details of the consistent method, referred to as the direct variable translation method, are described in Sec. 3.7.2.

The second approach defines the numerical flux for the two plasma models based on an underlying composite distribution function, which ensures the conservation of mass, momentum, and energy. Details of the conservative method, referred to as the composite distribution function method, are given in Sec. 3.7.3. In the limit where f_K approaches a Maxwellian distribution function identical to f_F, both approaches provide consistency and conservation.

3.7.2 Direct variable translation method

The direct variable translation method calculates the numerical flux for the multi-fluid plasma model and multi-species kinetic model in a manner that is consistent with the assumptions of the multi-fluid plasma model, which is that the distributions on either side of the subdomain interface are Maxwellians. Figure 3.5 illustrates the method for a particular species.

The procedure to calculate the numerical flux for the multi-fluid plasma model is to translate f_K at the subdomain interface into fluid variables n_K, $\boldsymbol{v}_K$, and T_K using Eqs. (2.2.8), (2.2.9), and (2.3.13). These fluid variables are then combined with fluid variable solutions from the multi-fluid plasma model at the subdomain interface, n_F, $\boldsymbol{v}_F$, and T_F, using a fluid Riemann solver as described in Sec. 3.1.4 to calculate a numerical flux. The effect is that the numerical flux for the multi-fluid plasma model is calculated from Maxwellian distribution functions f_F and f_{K_M} as shown in Fig. 3.5, where f_{K_M} is a Maxwellian distribution function with the same first three velocity moments as f_K. In this manner, the numerical flux for the multi-fluid plasma model is consistent with the assumption of Maxwellian distribution functions required for the fluid approximation.

The calculation of the numerical flux for the multi-species kinetic model is performed after the translation of n_F, $\boldsymbol{v}_F$, and T_F to f_F, allowing for application of Eq. (3.1.46a) using f_F and f_K. No assumptions need to be made in the numerical flux calculations for the multi-species kinetic model, which does not impose any restriction on distribution functions shape.

While the direct variable translation method provides consistency in that the numerical flux for the multi-fluid plasma model is calculated assuming Maxwellian distribution functions, deviation of f_K from a Maxwellian causes loss in accuracy, due to inexact representation of the distribution function using the only the velocity moments that yield n_K, $\boldsymbol{v}_K$, and T_K. This loss in accuracy manifests in a loss of conservation of the distribution function as well as the fluid model conserved variables of mass, momentum, and energy due to numerical fluxes being calculated for the multi-fluid plasma and multi-species kinetic models based on different distribution functions (f_F and f_{K_M} for the fluid model as opposed to f_F and f_K for the kinetic model). Conservation is provided, however, as f_K approaches a Maxwellian identical to f_F at the subdomain interface.

3.7.3 Composite distribution function method

While the direct variable translation method provides consistency in the specification of the numerical flux for the multi-fluid plasma model and becomes conservative as f_K approaches f_F, the composite distribution function method provides conservation for arbitrary f_K and becomes consistent as f_K approaches f_F. This conservative approach is performed by constructing a composite distribution function, f_C, from f_F and f_K, from which the numerical fluxes for the multi-fluid plasma model and multi-species kinetic model are calculated directly.

The method for constructing f_C is illustrated in Fig. 3.6 for the case of a subdomain interface in a local frame of reference in which the multi-fluid plasma model is solved on the left and the multi-species kinetic model is solved on the right. The numerical flux for the kinetic model as given in Eq. (3.1.46a) shows that for $v_x \geq 0$, $\mathcal{F} = v_x f_F$ and for $v_x < 0$, $\mathcal{F} = v_x f_K$, where the tildes are dropped for clarity of notation. This flux can be obtained

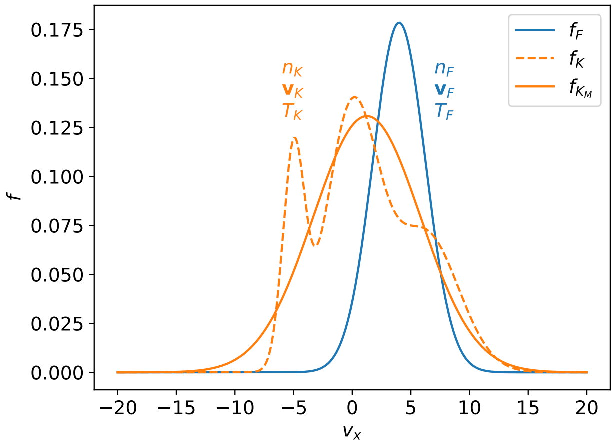


Figure 3.5: Illustrative distribution functions at a subdomain interface between the multi-fluid plasma model, f_F, and the multi-species kinetic model, f_K as in Fig. 3.4. Velocity moments of f_K are calculated using Eqs. (2.2.8), (2.2.9), and (2.3.13) yielding fluid variables n_K, $\boldsymbol{v}_K$, and T_K that describe an equivalent Maxwellian distribution function according to Eq. (2.2.14), denoted as f_{K_M}. The direct variable translation method calculates the numerical flux for the multi-fluid plasma model by approximating f_K as f_{K_M} and using a fluid Riemann solver as described in Sec. 3.1.4 on the fluid variables (n_F, $\boldsymbol{v}_F$, T_F, n_K, $\boldsymbol{v}_K$, T_K). In this way the method consistently calculates the fluid numerical flux. The numerical flux for the multi-species kinetic model is calculated by using f_F and f_K in Eq. (3.1.46a). As with Fig. 3.4, a single species is shown, however, the same procedure can be performed for multiple species. The differences between f_F and f_K are also exaggerated for illustrative purposes. In practice, f_F and f_K will have only small differences.

from a composite distribution function defined as

$$f_C(f_K, f_F) = \begin{cases} f_K \text{ if } v_x < 0 \\ f_F \text{ if } v_x \geq 0, \end{cases} \tag{3.7.1}$$

as illustrated in Fig. 3.6. If the multi-species fluid model is solved on the right of the subdomain interface and the kinetic model is solved on the left, the definition of f_C would be reversed, given by

$$f_C(f_F, f_K) = \begin{cases} f_F \text{ if } v_x < 0 \\ f_K \text{ if } v_x \geq 0. \end{cases} \tag{3.7.2}$$

The numerical flux for each model can then be calculated from this composite distribution function. For the multi-species kinetic model, application Eq. (3.1.46a) on f_F and f_K yields the numerical flux, identical to the calculation for the direct variable translation method, and by inspection of Eqs. (3.7.1) and (3.7.2), is equivalent to $\mathcal{F} = v_x f_C$.

The numerical fluxes for the multi-fluid plasma model are calculated from the velocity moments of f_C that yield the flux tensor terms in Eqs. (2.3.18), (2.3.19), and (2.3.20). The flux tensor terms for a particular species, written for an element face γ at a subdomain interface between models with normal $\boldsymbol{n}$, are given by

$$\left(\mathcal{F}_{i(\rho_\alpha)} n_i\right)^*_\gamma = \left(\rho_\alpha v_{\alpha i} n_i\right)^*_\gamma = n_{i\gamma} \int v_i \left(A_\alpha f_{C,\alpha}\right) d\boldsymbol{v}, \tag{3.7.3}$$

$$\left(\mathcal{F}_{ij\,(\rho_\alpha \boldsymbol{v}_\alpha)} n_j\right)^*_\gamma = \left(\left[\rho_\alpha v_{\alpha i} v_{\alpha j} + P_{\alpha_{ij}}\right] n_j\right)^*_\gamma = n_{j_\gamma} \int v_j \left(A_\alpha v_i f_{C,\alpha}\right) d\boldsymbol{v}, \tag{3.7.4}$$

$$\left(\mathcal{F}_{i(e_\alpha)} n_i\right)^*_\gamma = \left(\left[e_\alpha v_{\alpha i} + P_{\alpha ij} v_{\alpha j} + h_{\alpha i}\right] n_i\right)^*_\gamma = n_{i\gamma} \int v_i \left(\frac{1}{2} A_\alpha v_j v_j f_{C,\alpha}\right) d\boldsymbol{v}, \tag{3.7.5}$$

for Eqs. (2.3.18), (2.3.19), and (2.3.20), respectively.

Deriving a flux from the same composite distribution function for each plasma model ensures conservation of the distribution function and the conserved variables in the fluid

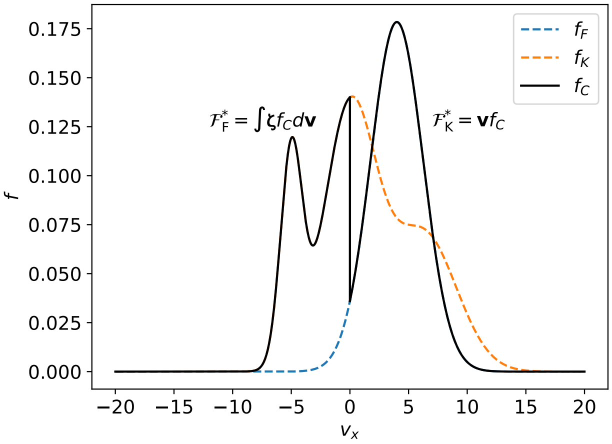


Figure 3.6: Illustration of a composite distribution function, f_C constructed from f_F and f_K at a subdomain interface between the multi-fluid plasma model and multi-species kinetic model, as shown in Fig. 3.4, for the case of the multi-fluid plasma model solved to the left of the boundary and the multi-species kinetic model solved to the right. The composite distribution function is constructed by considering the portions of f_K and f_F that advect across the subdomain interface, which by inspection of Eq. (3.1.46a) yields f_C defined by Eq. (3.7.1). For the composite distribution function method, the numerical flux for the multi-species kinetic model is calculated from f_K and f_F using Eq. (3.1.46a) or equivalently as $\boldsymbol{v} f_C$, which is identical to the direct variable translation method. The numerical flux using the composite distribution function method for the multi-fluid plasma model is then constructed from velocity moments of f_C, given by $\mathcal{F}_{\boldsymbol{\zeta}}{}^* = \int \boldsymbol{\zeta} f_C d\boldsymbol{v}$ for $\boldsymbol{\zeta} = [A_\alpha \boldsymbol{v}, A_\alpha \boldsymbol{v}\boldsymbol{v}, \frac{1}{2} A_\alpha \boldsymbol{v} v^2]$. This leads to a conservative method in which the numerical fluxes for the multi-fluid plasma model and multi-species kinetic model are constructed from the same underlying composite distribution function. The composite distribution function is shown for a single species, but as with the direct variable translation method, the composite distribution function method can be performed for multiple species. The differences between f_F and f_K are also exaggerated for illustrative purposes. In practice, f_F and f_K will have only small differences.

model, mass, momentum, and energy. However, for arbitrary f_K far from f_F, the fluid fluxes calculated in Eqs. (3.7.3), (3.7.4), and (3.7.5) are not solutions based on consistent assumptions of Maxwellian distribution functions, as would be the case if using a Riemann solver for fluid variables. Therefore, use of the composite distribution function method can lead to inaccurate solutions when f_K is far from f_F. As with the direct variable translation method though, conservation and consistency are recovered as f_K approaches f_F. In such a limit, the fluxes in Eqs. (3.7.3), (3.7.4), and (3.7.5) could be calculated analytically, as shown in Appendix D, which could be used to diagnose simulation results.

Extension to 1D2V

The extension to 1D2V is straightforward, in that the normal direction pointing from one subdomain to another in physical space is unchanged and is only in the x-direction, but now a v_y direction is added to the velocity space moments. Consider Fig. 3.7, showing how the negative velocity side of the kinetic distribution is combined with the positive side of the fluid Maxwellian distribution (shaded in grey), just as in 1D1V. However now, negative-sided integrals can be calculated

$$F^*_{\zeta_{v_x<0}} = \int\limits_{v_y=-\infty}^{v_y=\infty} \int\limits_{v_x=-\infty}^{v_x=0} \zeta \, dv_x dv_y, \tag{3.7.6}$$

where the ζ is the particular moment being calculated. The positive-sided integrals are

$$F^*_{\zeta_{v_x>0}} = \int\limits_{v_y=-\infty}^{v_y=\infty} \int\limits_{v_x=0}^{v_x=\infty} \zeta \, dv_x dv_y. \tag{3.7.7}$$

Extension to 2D2V

In 2D2V, the physical space normal between subdomains now has 2 components and the integrals in the v_x and v_y directions cannot be decoupled for an arbitrary direction. Consider an angle θ which the normal in 2D physical space makes at a subdomain interface. Then

$$\boldsymbol{n} = \cos\theta \hat{\boldsymbol{x}} + \sin\theta \hat{\boldsymbol{y}} \equiv n_x \hat{\boldsymbol{x}} + n_y \hat{\boldsymbol{y}}. \tag{3.7.8}$$

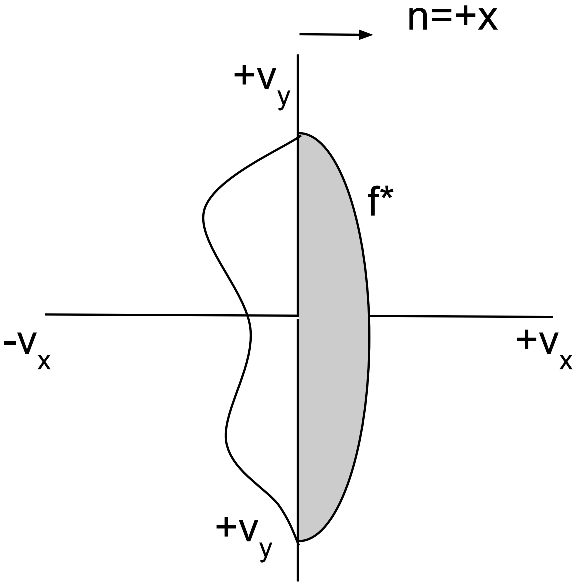


Figure 3.7: 1D2V composite distribution with the Maxwellian part shaded. This is a direct extension of the 1D1V composite distribution in Fig. 3.4. Again, the flux moments are calculated using a split of $v_x < 0$ and $v_x > 0$ as with the 1D1V case, with now the additional integration over all of v_y.

The slope of this line is

$$m = \frac{n_y}{n_x} = \tan\theta. \tag{3.7.9}$$

The composite distribution function along with possible directions it can be integrated for each side are shown in Fig. 3.8. The line of demarcation between sections of this composite distribution function is $\pm\frac{\pi}{2}$ from the normal line, which means the slope of this line is

$$\frac{v_y}{v_x} = \frac{-1}{m} = \frac{-1}{\tan\theta}, \tag{3.7.10}$$

which can be parameterized for v_y

$$v_y = \frac{-v_x}{\tan\theta}, \tag{3.7.11}$$

or v_x

$$v_x = -v_y \tan\theta. \tag{3.7.12}$$

Thus if the integral is performed as shown in Fig. 3.8b

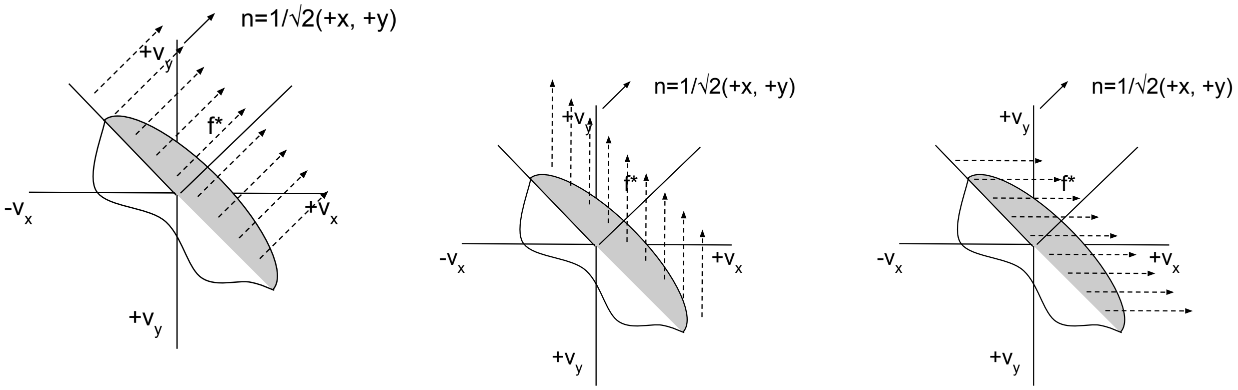
$$F^*_{\zeta_{\text{upper}}} = \int\limits_{v_x=v_{x_{\min}}}^{v_x=v_{x_{\max}}} \int\limits_{v_y=-\frac{v_x}{\tan\theta}}^{v_y=v_{y_{\max}}} \zeta dv_y dv_x, \tag{3.7.13}$$

$$F^*_{\zeta_{\text{lower}}} = \int\limits_{v_x=v_{x_{\min}}}^{v_x=v_{x_{\max}}} \int\limits_{v_y=v_{y_{\min}}}^{v_y=-\frac{v_x}{\tan\theta}} \zeta dv_y dv_x, \tag{3.7.14}$$

or for the case in Fig. 3.8c,

$$F^*_{\zeta_{\text{upper}}} = \int\limits_{v_y=v_{y_{\min}}}^{v_y=v_{y_{\max}}} \int\limits_{v_x=-v_y\tan\theta}^{v_x=v_{x_{\max}}} \zeta dv_x dv_y, \tag{3.7.15}$$

$$F^*_{\zeta_{\text{lower}}} = \int\limits_{v_y=v_{y_{\min}}}^{v_y=v_{y_{\max}}} \int\limits_{v_x=v_{x_{\min}}}^{v_x=-v_y\tan\theta} \zeta dv_x dv_y. \tag{3.7.16}$$



(a) Direction 1

(b) Direction 2

(c) Direction 3

Figure 3.8: 2D2V composite distribution function with integration direction. Figure 3.8a shows the direction of integration of the part of the positive portion of the Maxwellian distribution function constructed from fluid variables on the lower-left side of the interface that is moving into the upper-right kinetic side of the interface. Figures 3.8b and 3.8c show how the integration could be performed, in the way a parameterized double integration can be calculated, according to Eqs. (3.7.13) and (3.7.15), respectively.

3.7.4 Comparison of Direct Variable Translation and the Composite Distribution Function Methods

In Ch. 7, solutions using the direct variable translation and the composite distribution function methods are compared for some illustrative cases coupling the multi-fluid plasma model in 1D to the multi-species kinetic model in 1D1V. Performance is compared through observation of the solutions as well as measurements of conservation properties. A metric measuring the deviation from a Maxwellian distribution function in the kinetic subdomains, given by

$$\chi_\alpha = \frac{\int |f_\alpha - f_{M_\alpha}| \, d\boldsymbol{v}}{n_\alpha}, \tag{3.7.17}$$

where f_{M_α} is a Maxwellian distribution function related to f_α through the fluid variables in Eqs. (2.2.8), (2.2.9), and (2.3.13) [99, 127], is also used to compare the direct variable translation and composite distribution function methods.

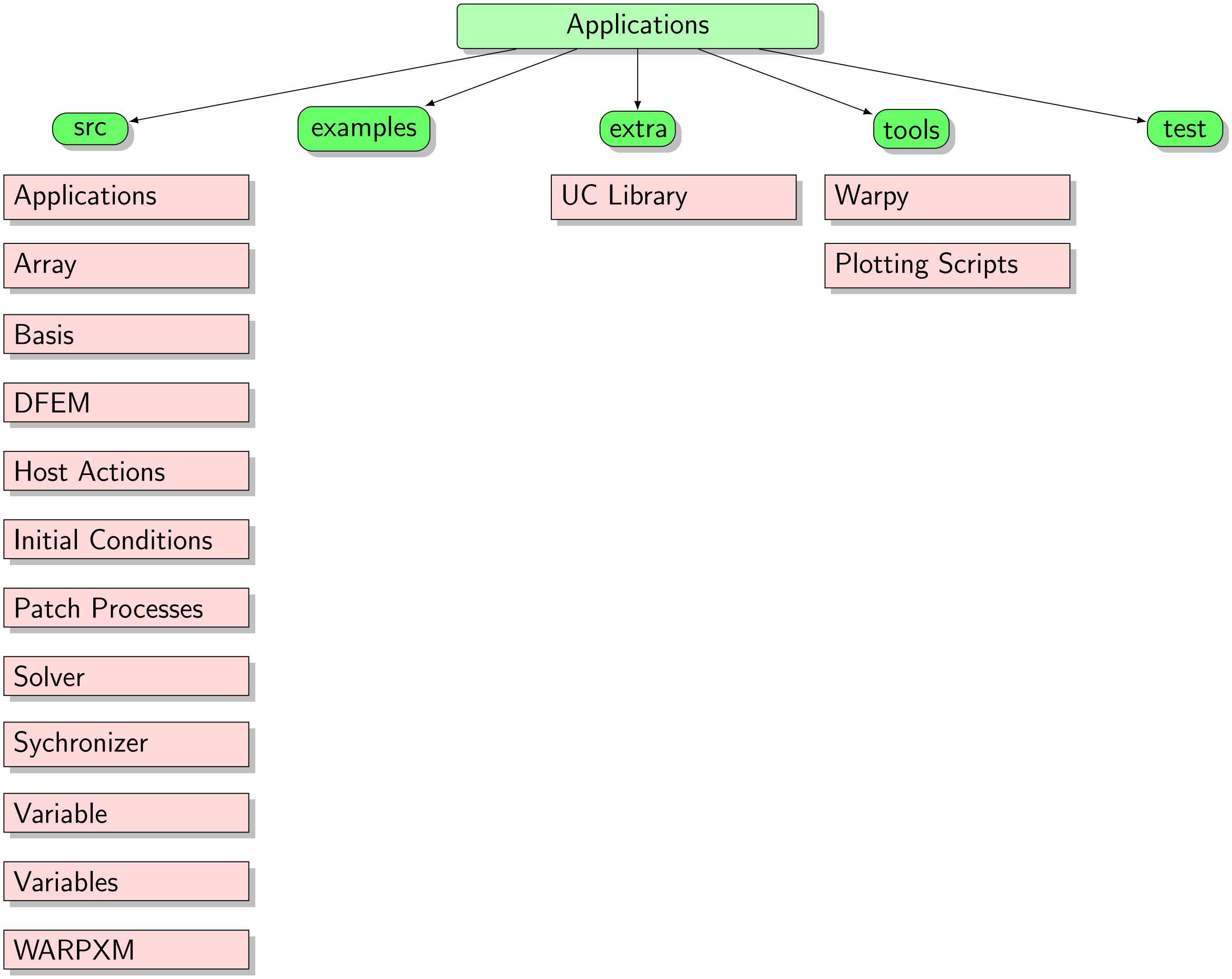
Chapter 4

WARPXM STRUCTURE

The models and numerical methods described in Chapters 2 and 3, respectively are implemented in the WARPXM (Washington Approximate Riemann Plasma eXtended modeling platform - Many-core version) code [73]. This is an unstructured framework designed to solve the plasma models using the discontinuous Galerkin method. The code was initially built by Sean Miller [79] which extended previous work on the structured WARPX and WARPM codes built by previous students ([109, 110, 52, 72, 99]) to an unstructured framework. This allows for simulation of more complex geometries for which the discontinuous Galerkin method is amenable. The code is also parallelized into subdomains and patches, upon which a problem can be broken up into multiple MPI processes across multiple host machines. In broad terms, the code consists of an unstructured library, a set of host actions, and a set of patch processes. The unstructured library handles incorporation of an external mesh file into usable information as well as handling geometric information associated with the mesh that is required by solvers. Hostactions are calculations over the the entire domain, such as time integration, while patch processes are calculations taking place within the patch level, such as the discontinuous Galerkin solver. A synchronizer is also used to transfer information between neighboring patches when required.

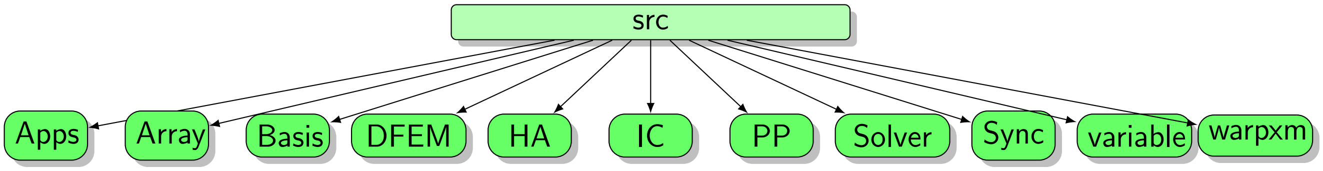
The rest of this chapter gives an overview of the general solver algorithm followed by a high level description of these main sections of the code.

4.1 High-Level Structure



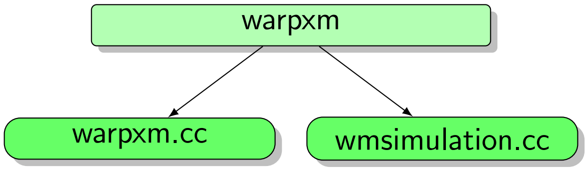
At the highest level, the code has a source directory upon which the meat of the code sits. There is an examples directory with example input files. The "extra" directory holds the Unstructured Converter (UC) library, which translates meshes into information that WARPXM can read and also performs patch and subdomain decomposition. The tools directory holds Warpy, which is a python suite developed to generate and run input files in WARPXM.

4.2 *Source Directory*



In the source directory are the integral pieces of the code. Physics applications, solvers, finite element bases, and variable arrays are all developed here.

4.2.1 warpxm



In this section, the basic workflow of WARPXM is overviewed. The entry point of the code is in the file `src/warpxm/warpxm.cc`. Here all of the required components are setup and run. A general outline of `src/warpxm/warpxm.cc` is as follows

- `main()`
    - `warpxm_init()`
        - initialize MPI
        - initialize petsc
    - `warpxm_main()`
        - initialize simulation
            - read input file
            - create cryptset
            - setup simulation
            - run simulation
    - `warpxm_finalize()`

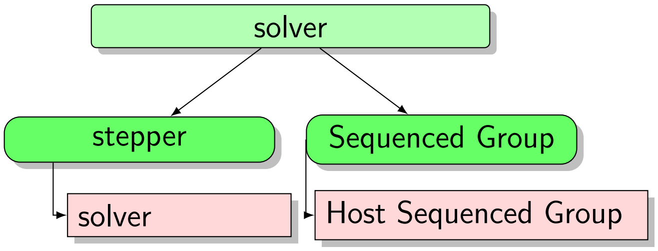
        - finalize petsc
        - finalize MPI

It can be seen that MPI and petsc are integrated at the highest level. Then the code reads in the input file using a cryptset object before setting up and running the simulation. The cryptset class is the module that translates the input file into the simulation parameters. The simulation is itself an object written in its own file `wmsimulation.cc`, detailed next.

- `setup()`
    - determines a run name and sets up various logging streams for output messaging to the user
    - creates a solver based on cryptset parameters
    - sets up solver
- `simulate()`
    - runs `solver.solve()`

Overall, this shows that the simulation sets up and runs the solver of the actual problem and relays information between the user and the simulation. An overview of `wmsolver.cc` is described next.

4.2.2 Solver

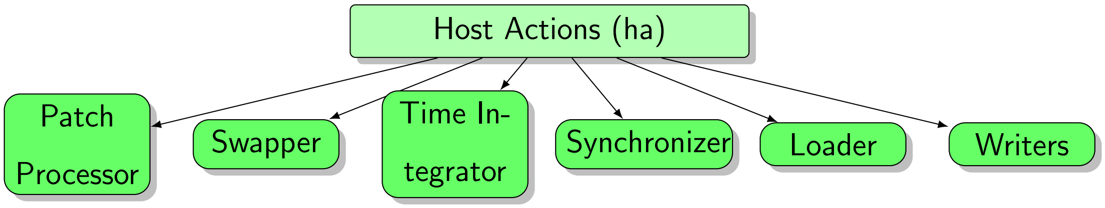


- `setup()`

  - setup `WmDomain` object
    - Sets up mesh object, which uses the UC Domain object to set up unstructured patches, which are then used to generate unstructured geometry objects to develop relevant mesh information
  - Read in variables
    - initialize variables
    - setup variables
  - Initialize host actions
    - for all host actions
      - initialize host action
      - setup host action
  - compile sequence groups
    - `startOnly`
    - `endOnly`
    - `perStep`
- `solve()`
  - `presolve()`
    - initialize hostactions
    - initialize startOnly or restart sequence
  - for each writeout frame to final time
    - `advance()` solution to next frame
      - while time less than time of next frame, run `step_dt()`, which runs `step()` function of the time integrator host action.

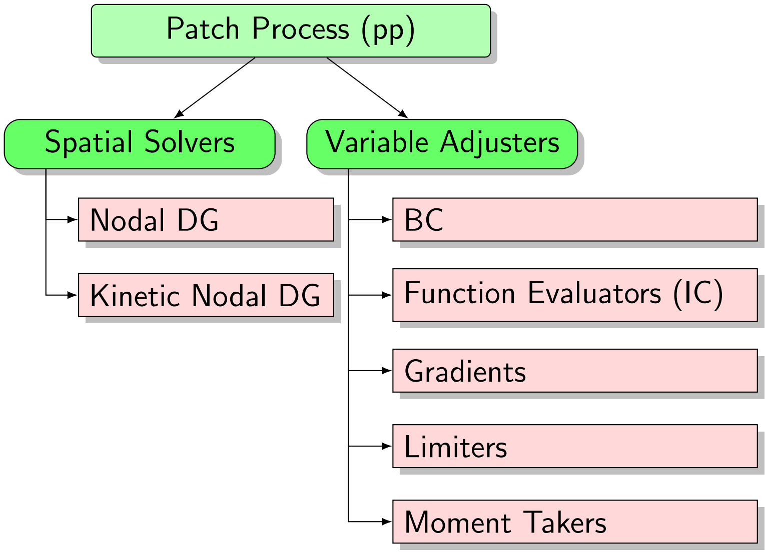
This shows that the solver is responsible for running the simulation as a whole. In the `setup()` function a `WmDomain` object is initialized, which in turn uses a `mesh` object to interface with the unstructured converter library, translating the mesh into usable information for WARPXM. It then reads in variables and sets up host actions, such as the time integrator. Finally it arranges tasks into various groups, such as `startOnly` occurring at the beginning of the simulation, `endOnly` occurring only at the end and `perStep` occurring at each timestep. The solution is advanced in the `solve()` function which runs the `advance()` function to call the time integrator from frame to frame. The time integrator itself is responsible for continuously advancing the solution within these frames. It also handles the spatial solvers and variable adjusters to update the right hand side of the calculation, occurring as patch processes. These components used by the solver (unstructured converter, host actions, and patch processes) are discussed further in following sections.

4.2.3 Host Actions



Host actions are procedures occurring across the entire domain of the calculation. Important examples include the time integrator (also known as the temporal solver), synchronizer which copies variables at interface elements between patches on different MPI processes, variable loader which loads variables from input files, swapper which swaps like variables between time integrator stages, writers which write variables to output files, and the patch-processor, which coordinates various patchprocesses within the domain. They should all have a `step()` function delineating procedures to be performed at each time step. At this level various time integration methods could be written. In this work explicit Runge-Kutta methods as described in Sec. 3.4 are employed.

4.2.4 Patch Processes



These are processes that occur at the patch level and can be called upon by a host action. In the case of the time integrator host action for example, the spatial solver and variable adjusters are called. The spatial solver effectively calculates the right hand side of the discretized partial differential equation in question. In WARPXM, this is the DG method given in Eq. (3.1.37). Variable adjusters, as their name suggests, adjusts the variable being solved for in some way before the time integration step occurs. This is the effect of boundary conditions applying a given value to a "ghost boundary node", gradient solvers (e.g. the application of Eq. (3.2.20)), and limiters. Initial conditions are calculated in a similar manner, but which are only applied as a startOnly step at the beginning of a simulation. Some more detail of these variables adjusters are given below.

- Boundary Conditions

  Boundary conditions specific to equation sets can be written by the end user. If non-periodic boundary conditions are required, WARPXM creates and extra layer of "ghost" elements around the domain, upon which boundary conditions can be set on nodes just exterior to the domain boundary. The user can call these node locations and apply equation-set dependent boundary conditions. Additionally, "virtual" boundary conditions between subdomains can be applied. This is useful for simulations solving different equation sets on different subdomains of the domain, but need interface conditions to stitch them together on the subdomain interfaces, as is needed by the domain-decomposed hybrid method. In these situations, the “ghost” layer is simply the first layer on the adjacent subdomain.

- Initial Conditions

  This module uses applications developed by the end user to create a set of variable adjusters known as function evaluators on the appropriate subdomains managed by a host action (called variable adjuster runner) that applies these function evaluators only at the beginning of the simulation. The user can write applications just like other physics applications for this purpose, where the initial conditions are set on variables through the `evaluate_function()` method that exists in the application class.

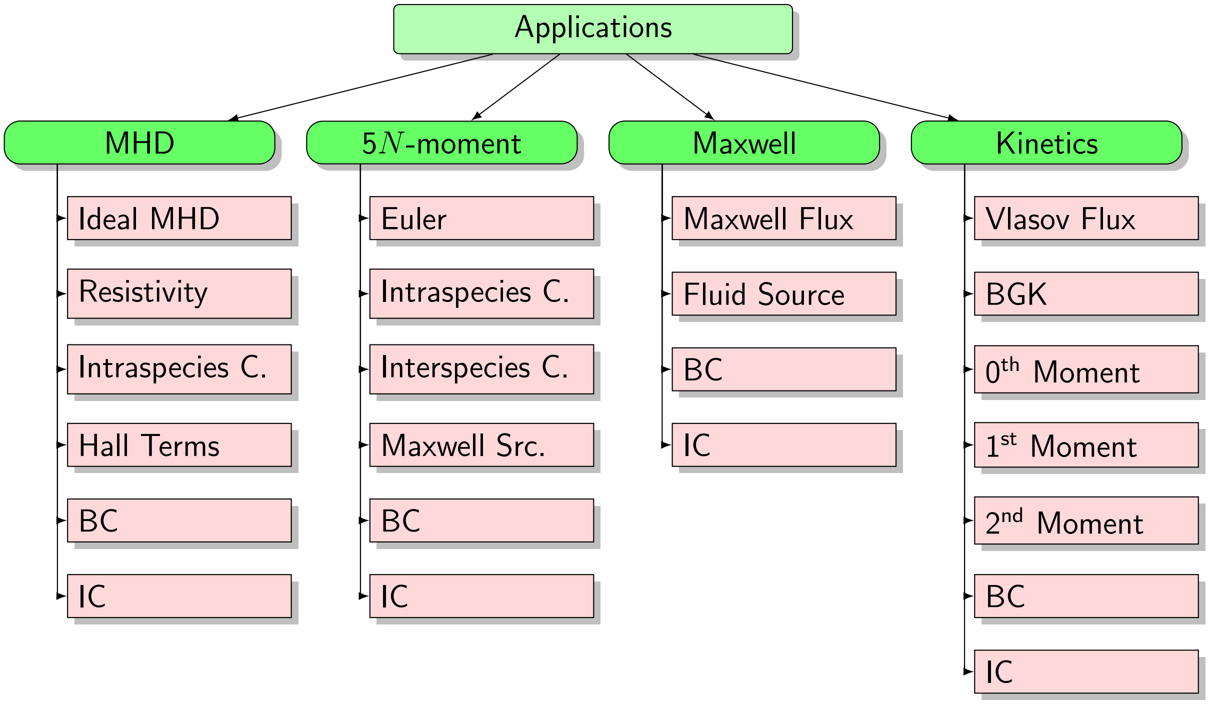
- Gradients

  This module is an application of Eq. (3.2.20). In a problem involving higher order derivatives, for each timestep one applies this gradient calculation, followed by a relevant boundary condition on them, before applying the spatial solver equation. In this work, the local discontinuous Galerkin (LDG) formulation has been implemented through Eqs. (3.2.27) and (3.2.28) as well as the interior penalty (IP) method in Eqs. (3.2.35), (3.2.36), and (3.2.43).

- Limiters

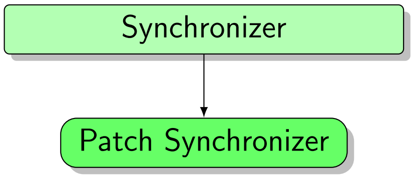
  A few limiter implementations of the slope-moment type limiters mentioned in Sec. 3.3 have been implemented as variable adjusters. These include limiters described by Moe et al. [81] and Tu et al. [122]. An artificial viscosity limiter has also been implemented, though as an application for the spatial solver instead of a variable adjuster, as it effectively adds a term to the equation set.

4.2.5 Applications



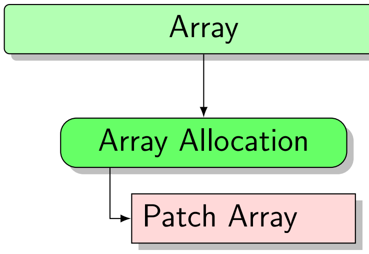
The actual implementation of the physics models as discussed in Chapter 2 happens at this level. Specifically, for the DG solver this happens when applying Eq. (3.1.37) where $\hat{\mathcal{F}}^\lambda$, $\tilde{\mathcal{F}}^\lambda$, and $\hat{\mathcal{S}}^\lambda$ are determined by the physics model. These are implemented through a set of applications which the end user writes.

4.2.6 Synchronizer



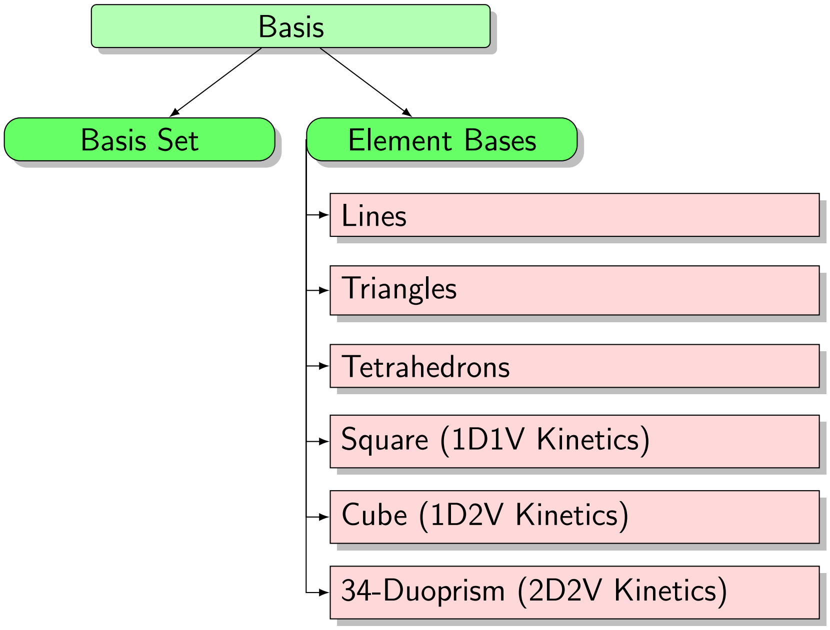
The synchronizer manages the information passing between patches within the domain. Normally, running WARPXM with multiple processes results in the subdomains subdividing into separate patches, one for each MPI process. The synchronizer holds the element numbers between patches that need to be synced at particular points in the simulation. It is held by a host action, such as a time integrator, which uses it to sync data along patch boundaries between timesteps or variable adjuster applications. Asynchronous sends and receives are used, where data is sent between adjacent patches, and then each adjacent patch waits to receive its data from the other patch before moving on in the calculation.

4.2.7 Array



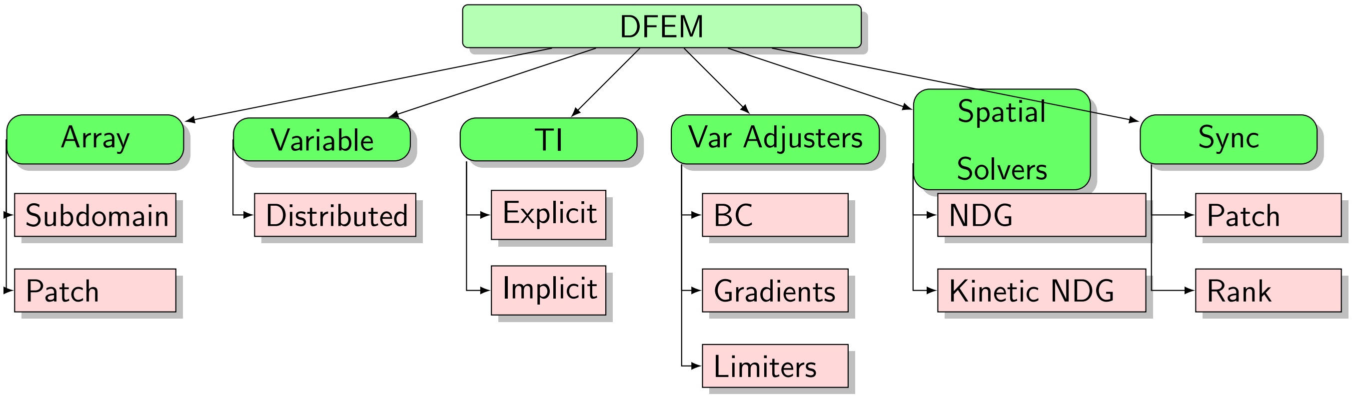
The patch array class here sets up a C++ vector in which data is to be stored. Full implementation is done in the DFEM directory to write this specifically for DG. This array is called by the DG method to operate on variables at node locations. It also gets written to the hdf5 output on writeout steps.

4.2.8 Basis



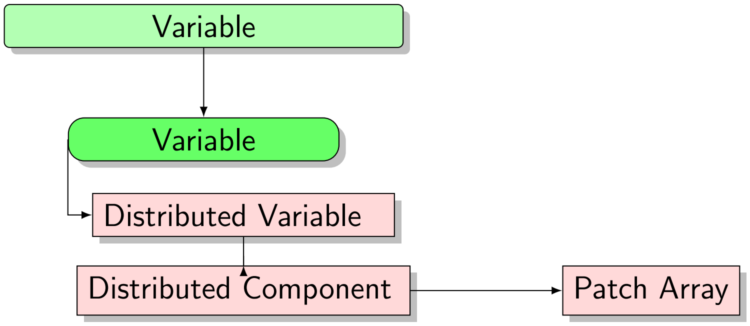
This holds element basis information in the form of text files for various element types with various order. These files are precomputed using mathematica scripts and yield information such as node locations, basis functions, inverse mass matrices, advection matrices, etc. For kinetics, python scripts are used to compute bases using tensor products of lower dimensional elements as described in Sec. 3.6. The role of the basis set is to translate these text files into variables/arrays/matrices for use in WARPXM.

4.2.9 DFEM



The DFEM (Discontinuous Finite Elements) module is where the actual implementations of various host actions, patch processes, variables, arrays, etc. for the DG formulation are written. From the perspective of object-oriented programming, the base (parent) classes of these modules are implemented elsewhere but the children classes specific to DFEM implementation are defined here.

4.2.10 Variable



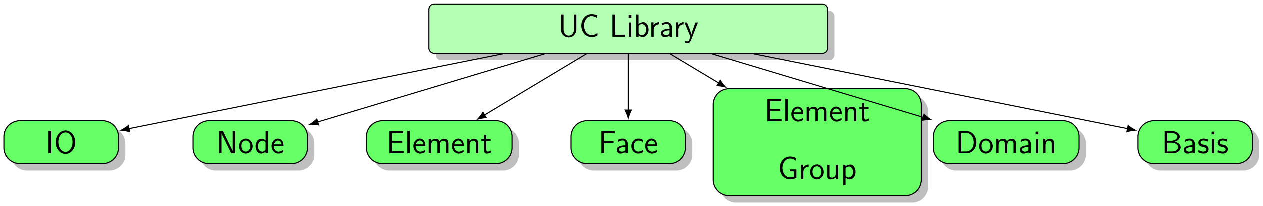
The variable class holds actual patch arrays for various model components and is used when writing to and reading from output files. A distributed variable may be of an "MHD Fluid" which then would have 8 distributed components ([ρ, ρv_x, ρv_y, ρv_z, e, B_x, B_y, B_z]). Each of these distributed components holds the patch arrays with their values at various nodes in the patch.

4.3 Unstructured Framework

The unstructured framework in WARPXM can be thought of as having two primary components, consisting of

1. Unstructured Converter (UC) Library
2. Geometric Calculations (Unstructured Geometry Object)

4.3.1 Unstructured Converter Library



The UC library handles incorporation of the mesh file into geometric information that WARPXM can use and handles subdomain and patch decomposition. It does this through a series of objects that handle different aspects of the domain decomposition and geometric calculations.

At the highest level, the domain object is the interface class responsible for reading, partitioning, accessing and writing unstructured meshes. It holds an element group which is an object that represents various groups of elements, including full domains, subdomains, and patches. Information from the meshfile is converted into these structures. The Metis graph partitioning library [61] is then invoked by the highest-level element group representing the entire domain to perform patch partitioning. Further details of how this library is used is in Sec. 4.3.2.

Contained by the element groups are nodes, elements, and faces, which are also determined from the meshfile. Nodes are the actual points with coordinates given by the meshfile while elements are collections of nodes connected by a connectivity mapping. They also hold face objects, also determined by connections between nodes. The faces are used to give elements local neighborhoods, mapping them to neighboring elements through these faces. They also have an orientation, which determine how a neighboring face is oriented with respect to it, which determines the opposite node numbers used in numerical flux calculations.

Additionally, a layering calculation is also performed on each element, setting a distance between an element and the edge of the element group. This collects elements within the element groups into separate layers, which can be used by the numerical method to apply certain calculations on specific layers of the element group. External layers are also added, specifying the elements that must be synchronized with those from an adjacent patch, usually on a different MPI process, during various stages in simulations. Finally, periodic boundary conditions are enabled by a determination of conjoining nodes on the domain.

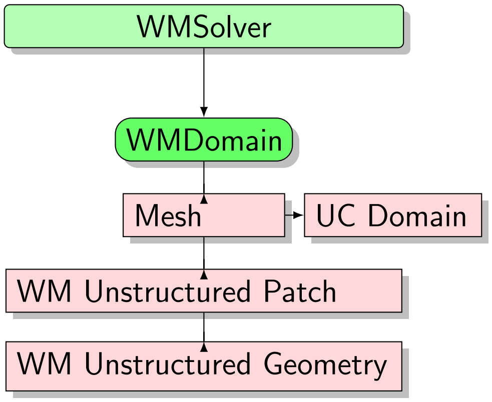
4.3.2 Metis Partitioning

The goal of the patch partition is to split the domain (or subdomain) up into roughly equivalent sizes, or weightings, between patches, often with each patch being worked on by a separate MPI process. Current usage involves a call to the `METIS_PartGraphRecursive()` function in the Metis library. This function requires the mesh to be organized into a graph, where "vertices" are the elements and the "edges" are the element neighbors. Normally, an input mesh file contains 3 sources of information, the node coordinates, the element connectivities which give nodes that make up each element, and nodesets which specify boundary nodes. The UC library uses this information to calculate element neighborhoods which give the neighboring elements along each element face, as well as the orientation between the faces. The neighborhood information is then called into the `METIS_PartGraphRecursive()` function. The information needed by the function needs to be put in a CSR format where the value array corresponds to locations of an element × element matrix (though not explicitly given to the function) where we have 1's (or true's) where the element in a row has a neighbor at a column. The column array (called `adjncy`) corresponds to the neighbors and the row pointer array (called `xadj`) allows for indexing into the neighbors for given elements. The function then partitions the mesh according to a specified number of partitions (the number of patches the domain is to be decomposed into) and weights for each partition. Current usage exhibits default behavior in which all patches (often one for each compute device available which are CPUs in the present system) are weighted evenly so that each patch should contain roughly the same number of nodes. However, weightings can be given to different processes for load balancing in case certain compute cores are faster than others, for example if GPUs are additionally used. Also, if different models are used in different patches, weights can also be assigned due the difference in computational cost between models.

Metis also provides the functionality to take mesh information directly in the form of connectivities instead of neighborhoods. In this case, the matrix that the CSR arrays holds can be thought of an element × node matrix where 1's (or true's) are where the element in a row has a node at a column. The function `METIS_PartMeshDual()` could be used for this where the column array becomes instead an array of connectivity nodes per element (denoted as `eind`) and the rowpointer array becomes the array to these connectivities per element (denoted as `eptr`). One specifies the number of connecting nodes that specify an edge between elements (1 for lines in 1D, 2 for triangles in 2D, or 3 for tetrahedrons in 3D), and the Metis library determines neighborhoods on its own before then performing the graph partition. The advantage of this method is that it can be used directly with a meshfile that only gives connectivity information and that neighborhoods do not have to be calculated (though it is calculated in WARPXM for use in the DG algorithm to find neighboring elements). However at this time, the graph function `METIS_PartGraphRecursive()` is used to partition domains and subdomains. Future work should consider adjustments of weights according to model usages in hybrid simulations for load balancing between partitions.

An example of usage of the Metis partitioning is shown in Appendix E.

4.3.3 Geometric Calculations



The output of the unstructured converter library is used to develop elemental geometry information, such as Jacobians, element centroid locations, area/volumes, etc. These are specific calculations for various element types, and as element types are added into WARPXM, these calculations must be made. It is created through the mesh which uses the UC Domain object to first develop an unstructured patch object, which then has the necessary information to populate the unstructured geometry object with the relevant information. This is then further expanded upon by the unstructured DG object which adds DG information on to this, such as basis information and other relevant information required by the DG solver. At the time of writing, the unstructured geometry object contains the geometric calculations for 1D line, 2D triangle, and 3D tetrahedron elements based on the information in the mesh file as interpreted by the UC library.

As an example of a Jacobian calculation, consider a 2D isoparametric triangle with three nodes at $(0,0)$, $(1,0)$ and $(1,1)$, respectively, in a 2D isoparametric space (ξ_1, ξ_2). Assuming basis expansion as given in Eq. (3.1.15), the position in real space for some element λ can be written as

$$\boldsymbol{x}^{\lambda} = \hat{\boldsymbol{x}}_1^{\lambda}\psi_1(\boldsymbol{\xi}(\boldsymbol{x})) + \hat{\boldsymbol{x}}_2^{\lambda}\psi_2(\boldsymbol{\xi}(\boldsymbol{x})) + \hat{\boldsymbol{x}}_3^{\lambda}\psi_3(\boldsymbol{\xi}(\boldsymbol{x})), \tag{4.3.1}$$

which has a three-node second-order basis. With a nodal basis this becomes

$$\boldsymbol{x}^{\lambda} = \boldsymbol{x}_1^{\lambda}\left(1 - \xi_1(\boldsymbol{x}) - \xi_2(\boldsymbol{x})\right) + \boldsymbol{x}_2^{\lambda}\xi_1(\boldsymbol{x}) + \boldsymbol{x}_3^{\lambda}\xi_2(\boldsymbol{x}) \tag{4.3.2}$$

assuming $\boldsymbol{\xi} \in (0,1)$. So for example, the Jacobian terms are

$$J^\lambda = \begin{pmatrix} \frac{\partial x_1^\lambda}{\partial \xi_1} = x_2^\lambda - x_1^\lambda & \frac{\partial x_1^\lambda}{\partial \xi_2} = x_3^\lambda - x_1^\lambda \\ \frac{\partial y_1^\lambda}{\partial \xi_1} = y_2^\lambda - y_1^\lambda & \frac{\partial y_1^\lambda}{\partial \xi_2} = y_3^\lambda - y_1^\lambda \end{pmatrix}. \tag{4.3.3}$$

Thus, given coordinates from the mesh of an element, λ, the Jacobian can be readily calculated when required.

4.4 Kinetic Framework

Kinetic calculations in WARPXM are performed by stacking a velocity space on top the physical space that already exists. Functionally, this means the distributed variable array, holding variable values on element nodes are extended into "super elements" that stack all nodes in the velocity space onto the element for kinetic variables. For example, in the simple case of second-order 1D1V, nominally there are two nodes per each line element that is allocated into the distributed array. However, for phase space variables, nodes are added according to the number of nodes in the corresponding phase space element and the extent of velocity space. In second-order 1D1V, the phase space element is square. So if velocity space extends 10 elements in v_x, then the phase space super element will consist of $4 \times 10 = 40$ nodes per super element. The kinetic DG implementation then must correctly calculate node numbers when in phase space. In this way, any arbitrary geometry in physical space can be used.

4.5 Domain-Decomposed Hybrid Method

The domain-decomposed hybrid method as described in Sec. 3.7 can be thought of as a boundary condition between models at a subdomain interface. The direct variable translation method as described in Sec. 3.7.2 directly translates variables at these subdomain interfaces to the appropriate model before calculating consistent numerical fluxes. This is performed through virtual boundary conditions, which set values of variables on internal edges of subdomains that are consistent with adjacent subdomains' models. The numerical flux can then be calculated at each subdomain interface using the values consistent with each model. For the composite distribution function method as described in Sec. 3.7.3, the boundary condition calculations can be sidestepped and the numerical fluxes can be directly calculated, such as by summation of the sided moments of the distribution functions on either side of a subdomain interface that construct the composite distribution function.

The domain-decomposed hybrid method allows for reduction in required memory and simulation time due to the fact that certain subdomains can use reduced models with smaller sets of distributed variables and less computationally-intensive calculations. Future work could make the domain decomposition dynamic or adaptive, in which the mesh is remapped at periodic intervals using the Metis partitioning tools as described in Sec. 4.3.2. Such dynamic remapping of subdomains can be useful in simulations in which the physical accuracy of models in different regions change over time, which can be determined using metrics such as χ in Eq. (3.7.17).

Chapter 5

FLUID MODELING

5.1 MHD

5.1.1 1D Brio-Wu Shocktube

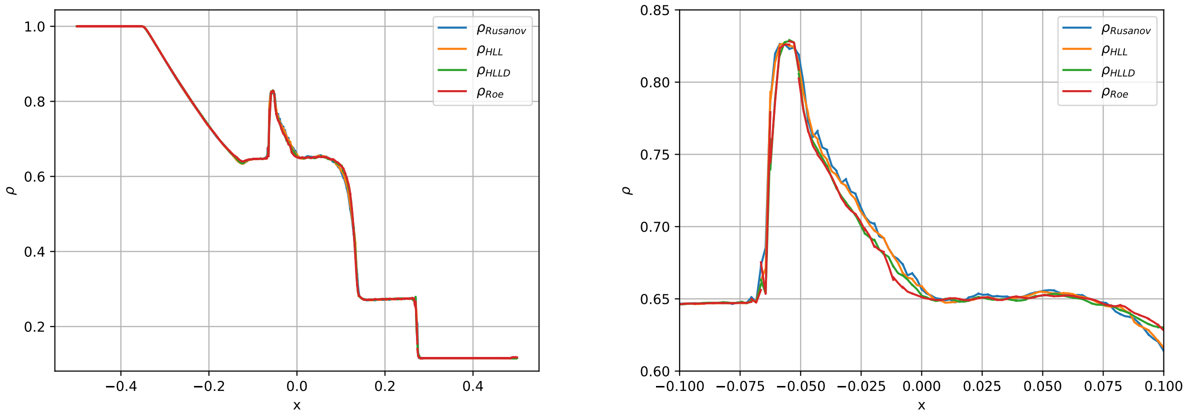
An early task of this work was to implement the ideal MHD model in WARPXM. The Brio-Wu shocktube problem is a standard 1D Riemann test case for ideal MHD [13] and is used to test various numerical flux algorithms. The initial condition is given by

$$(\rho, v_x, v_y, v_z, p, B_x, B_y, B_z)_{t=0} = \begin{cases} (1.0, 0.0, 0.0, 0.0, 1.0, 0.75, 1.0, 0.0) \\ (0.125, 0.0, 0.0, 0.0, 0.1, 0.75, -1.0, 0.0) \end{cases} \tag{5.1.1}$$

In this problem, $\gamma = 5/3$ is chosen and let run to $t = 0.2$. 256 third order elements (second order polynomials) with third order SSPRK3 timestepping is used, as shown in Eqs. (3.4.4)-(3.4.6). Dirichlet boundary conditions are employed, though the problem is not run long enough for them to interact with waves propagating from the discontinuity. The problem is also run with the slope-moment limiter [81] for stability. Figure 5.1 shows the density at $t = 0.2$ for the various fluxes that were implemented for ideal MHD, including Rusanov, HLL, HLLD, and Roe. These plots show good agreement between methods. No analytical solution is calculated upon which to compare, though work by Torrilhon suggests an exact Riemann solver for Ideal MHD [120, 121] which could possibly be added in the future for testing MHD problems such as this.

5.1.2 Orszag-Tang

The Orszag-Tang vortex problem is a classic 2D test case for ideal MHD [86]. Results from this problem are given in Fig. 5.2, which is run with 288 triangular first order elements using



(a) Density

(b) Density Closeup

Figure 5.1: MHD Brio-Wu shocktube at t=0.2 Plot of Density for Various fluxes. This is run with with 256 third order elements with SSPRK3. 5.1a shows the whole domain while figure 5.1b shows a closeup. (Simulations of the Brio-Wu shocktube are performed with WARPXM version 0.0.2 / flux_bc branch using input file brio_wu_shocktube.py).

ideal MHD with an HLLD flux. Initial results as seen in Figs. 5.2a and 5.2c show a high level of divergence error, which causes numerical instability and stops the simulation after some time. A divergence cleaning procedure as described by Dedner [37] was implemented to remove this, for which results are given in Figs. 5.2b and 5.2d. These show much better behavior and a diminished divergence error.

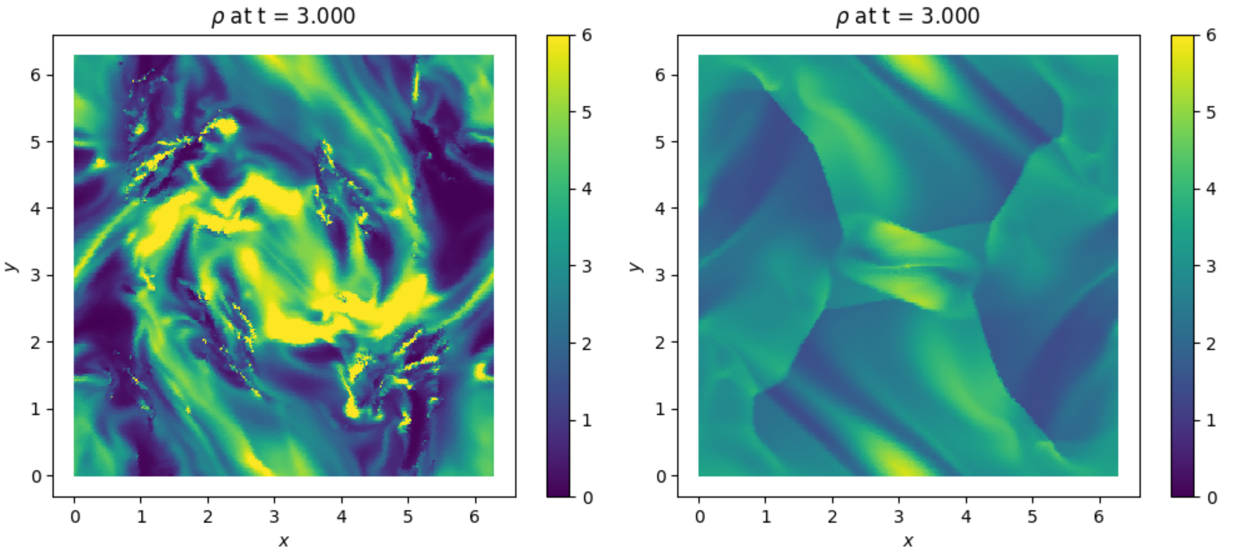
5.1.3 Hartmann Flow

The Hartmann flow is a test case for MHD with the addition of resistive and viscous terms. To incorporate these effects, the local discontinuous Galerkin and interior penalty formulation are required. The geometry of the problem is shown in Fig. 5.3a. Figure 5.3b shows a magnetized Couette equilibrium case with normalized $v_{\text{wall}} = 1$ for $\frac{\delta_p}{L} = 1/3$, $\nu_p \tau = 1$ corresponding to Hartmann Number, Ha = 3 and Reynolds number, Re = 1. The initial condition is set to the expected equilibrium as calculated in Appendix F and the MHD equations with the addition of the resistive and viscous terms implemented using the interior penalty scheme are solved to $t = 0.5$ with the energy equation turned off. The HLLD flux is used for ideal MHD terms. 64 fourth order elements are used with RK4 time integration. It can be seen that in this situation, the equilibrium holds, as the black-dashed lines in the plot corresponding to the equilibrium solution are matched to the DG solution, which does not deviate from the initial condition.

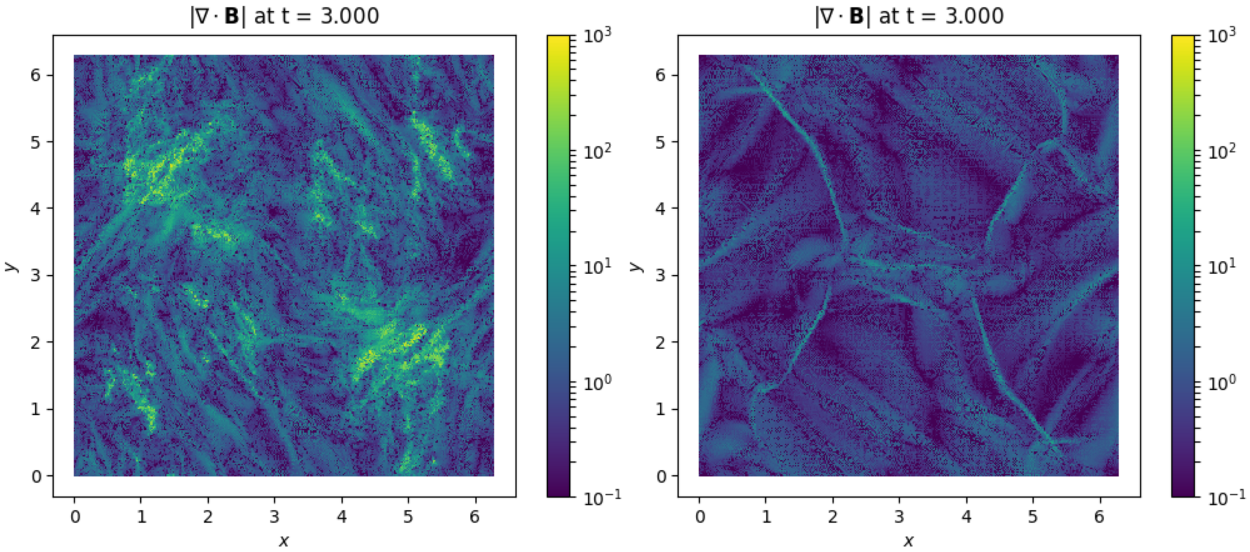
5.2 5N-moment

5.2.1 Sod Shocktube Limiter Comparison

The moment-slope [81] and artificial viscosity limiters described in this work are compared with each other using the Euler Sod Shocktube test case. This is a standard neutral fluid shock-capturing Riemann problem which can be used to assess numerical methods. The

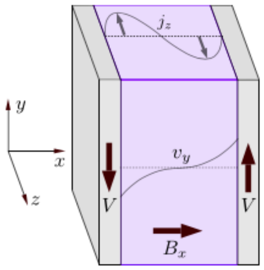


(a) ρ: without cleaning (b) ρ: with cleaning

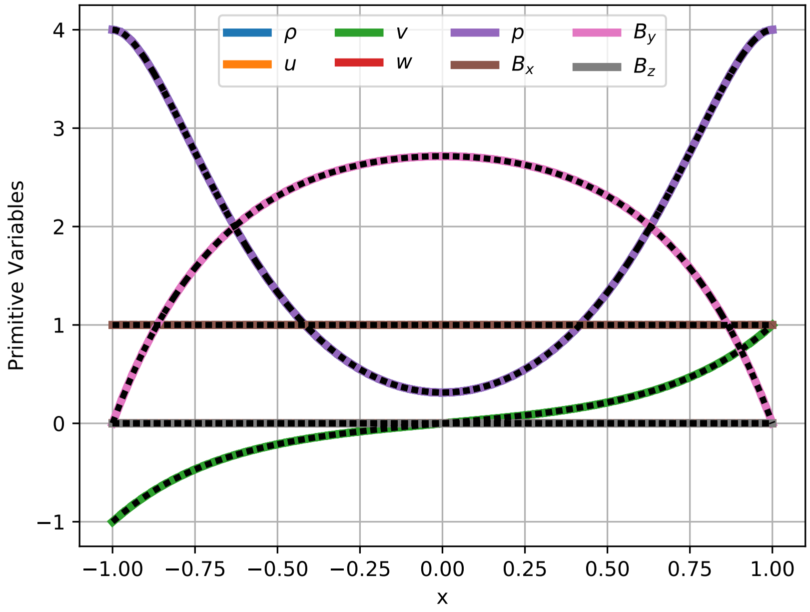


(c) $|\nabla \cdot \boldsymbol{B}|$: without cleaning (d) $|\nabla \cdot \boldsymbol{B}|$: with cleaning

Figure 5.2: Orszag-Tang problem with and without divergence cleaning. (Simulations of the Orszag-Tang problem are performed with WARPXM using input file orszag_tang.py).



(a) Hartmann Flow Geometry



(b) Hartmann Flow at $t = 0.5$ after initializing to equilibrium (Energy equation off).

Figure 5.3: Hartmann flow problem (Simulations of the Hartmann flow problem are performed with WARPXM version 0.0.2 / flux_bc branch using input file hartmann_flow_mhd.py).

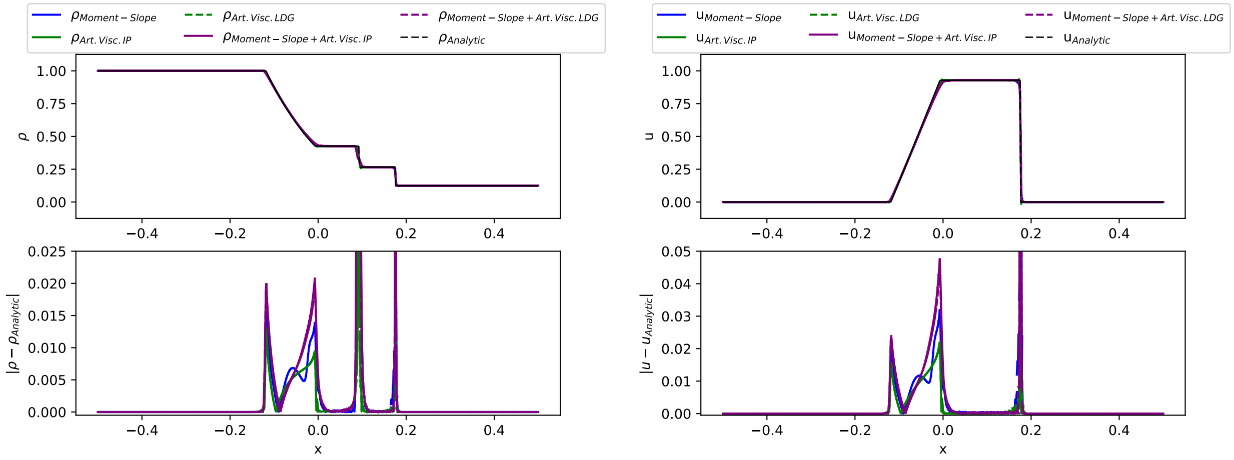
Riemann problem is given by

$$(\rho, v_x, p) = \begin{cases} (1.0, 0.0, 1.0) \text{ for } x < 0.0 \\ (0.125, 0.0, 0.1) \text{ for } x \geq 0.0. \end{cases} \tag{5.2.1}$$

The simulation is run on a domain $x \in [-0.5, 0.5]$ with 400 third order elements (2nd order polynomials) with SSPRK3 timestepping as shown in Eqs. (3.4.4)-(3.4.6), to a time of $t = 0.1$ with a fixed $dt = 10^{-5}$. The Euler equations are solved using Roe fluxes and are subject to Dirichlet boundary conditions, though boundary waves do not interfere with the waves emanating from the Riemann problem in the time frame of the simulation. Results are shown in Figs. 5.4a, 5.4b, and 5.4c. Five different cases are shown along with the analytic solution, calculated using the algorithm detailed by Leveque [70]. These five case are

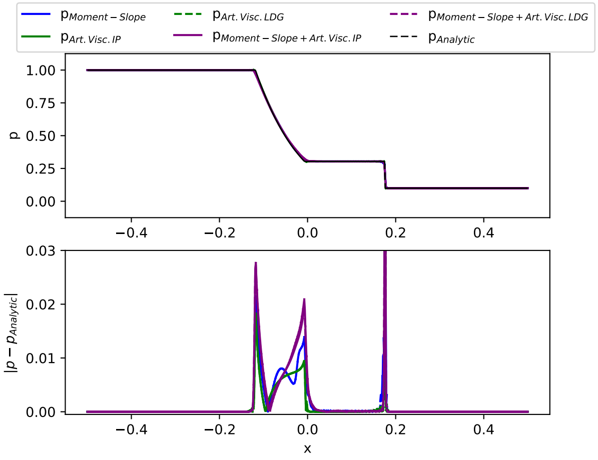
1. moment-slope limiter only
2. artificial viscosity with interior penalty (IP)
3. artificial viscosity with local discontinuous Galerkin (LDG)
4. moment-slope limiter combined with artificial viscosity with interior penalty
5. moment-slope limiter combined with artificial viscosity with local discontinuous Galerkin

The error at each point is also shown in these plots, which show that in this case, the IP and LDG methods exhibit little difference in solution, and that the use of the artificial viscosity limiter alone seems to yield less error than either the case of only the moment-slope limiter or the combination case.



(a) Density

(b) X-Velocity



(c) Pressure

Figure 5.4: Sod shocktube with limiting. (Simulations of the Sod shocktube problem are performed with WARPXM using input file sod_shocktube.py).

5.2.2 Fast Magnetic Reconnection Problem

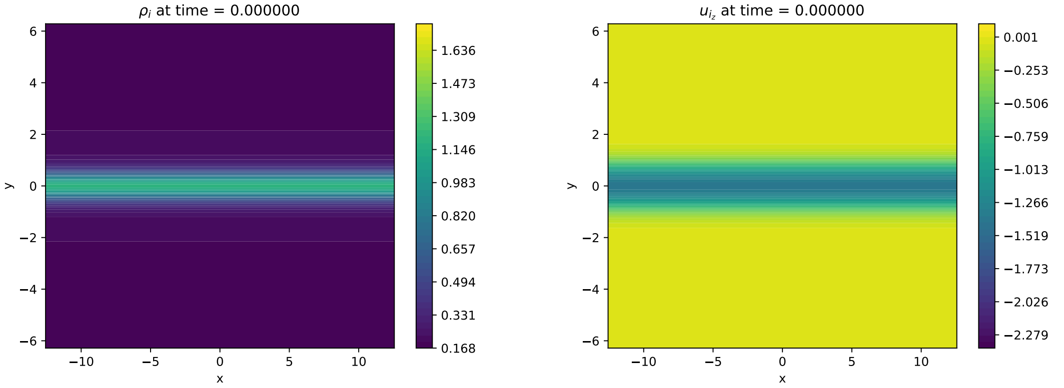
The Geospace Environmental Modeling (GEM) collisionless magnetic reconnection challenge is a test case assessing a code's ability resolve problems involving magnetic reconnection [8, 9]. The problem setup follows the equilibrium and conditions given by Birn [8] in which a current sheet is initialized with a shearing B_x across the x-axis and out-of-page current. A perturbing magnetic field is added to start the simulation. The equilibrium setup derivation for the $5N$-moment model is given in Appendix G, with the equilibrium initial conditions along with a perturbing magnetic field as given in Eqs. (G.2.1) - (G.2.5). The problem is run on a domain of $x \in [-4\pi, 4\pi]$ and $y \in [-2\pi, 2\pi]$, with $\lambda = 0.5$, $n_\infty/n_0 = 0.2$, $T_i/T_e = 0.2$, and $A_\mathrm{i}/A_\mathrm{e} = 25$ (Birn parameters). Additionally, the constants $\frac{\delta_p}{L} = 1$ and $\omega_p \tau = 20$ are used. This sets the speed of light to reference speed $c/v_0 = 20$, which helps to reduce the computational cost of the simulation. The initial conditions are plotted in Fig. 5.5.

The simulation is run to $t = 40\tau_c$. The metric for comparison with other results is the reconnected flux of magnetic field strength across the x-axis (on half the domain):

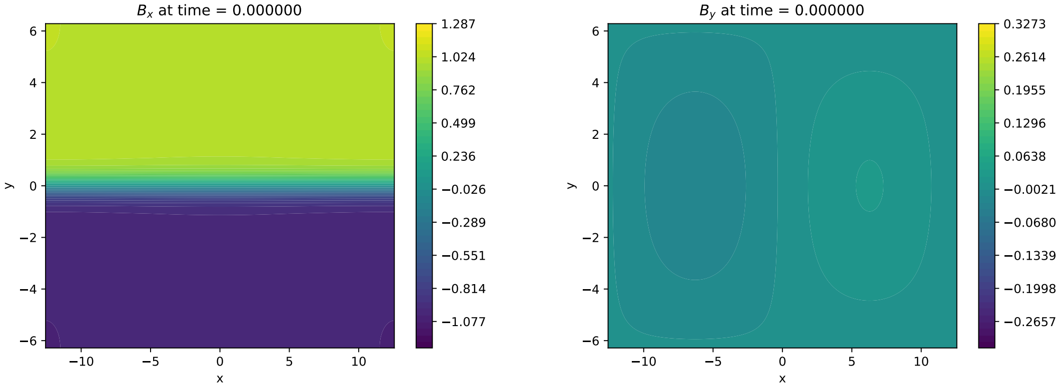
$$\phi = \frac{1}{2} \int_{-4\pi}^{4\pi} |B_y(x, y = 0)| \, dx \tag{5.2.2}$$

Figure 5.6 shows results for the slope-moment limiter with $\alpha = 5$ at $t = 19.6$ and $t = 23.6$ after which the simulation fails, while Fig. 5.8a shows the reconnected flux of a few cases with the slope-moment limiter, compared with results published by Birn [8, 9]. For $\alpha = 500$, the results follow the curve but the limiter is not aggressive enough and the simulation fails fairly early. With $\alpha = 5$, the simulation does run farther but does fail eventually. Future work could involve understanding how to suppress instability and run the simulation for longer times.

A similar scenario plays out with the artificial viscosity limiter, with particular results shown in Fig. 5.7 with reconnected fluxes shown in Fig. 5.8b. A few different spatial resolutions, polynomial orders, and viscosity coefficients were tried, though the simulation does not run to the desired end time. The reconnected flux profile does seem to follow the published

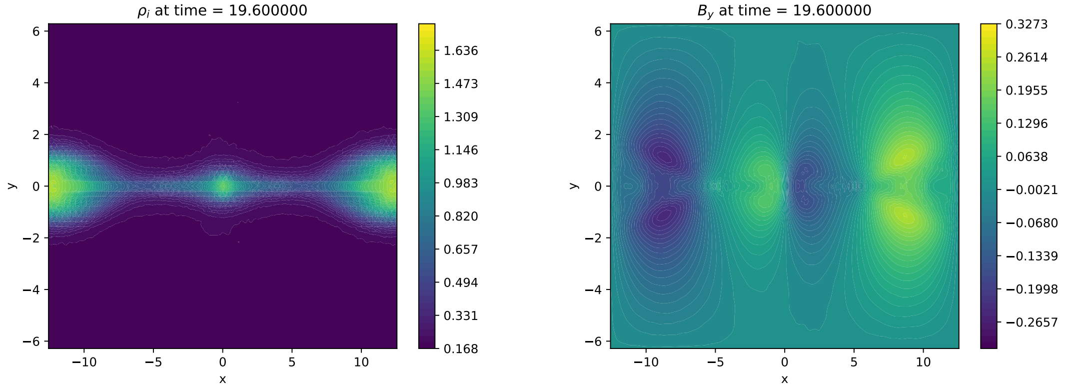


(a) ρ_i (b) u_{i_z}

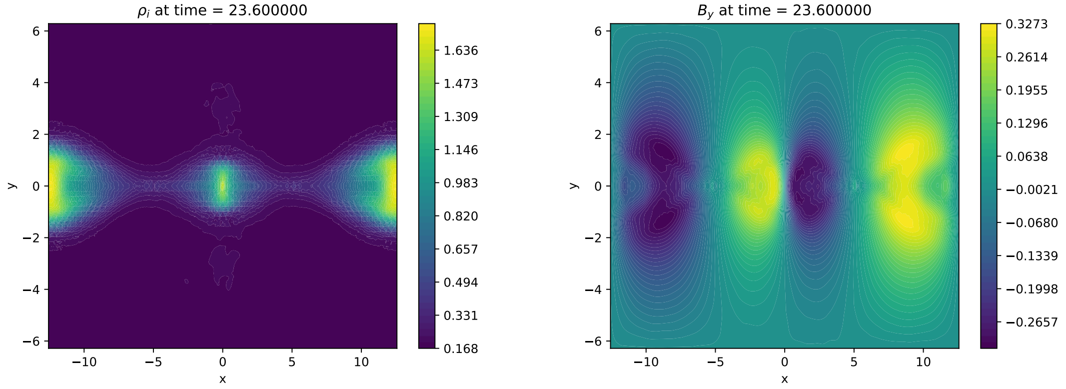


(c) B_x (d) B_y

Figure 5.5: Gem Challenge Initial Conditions

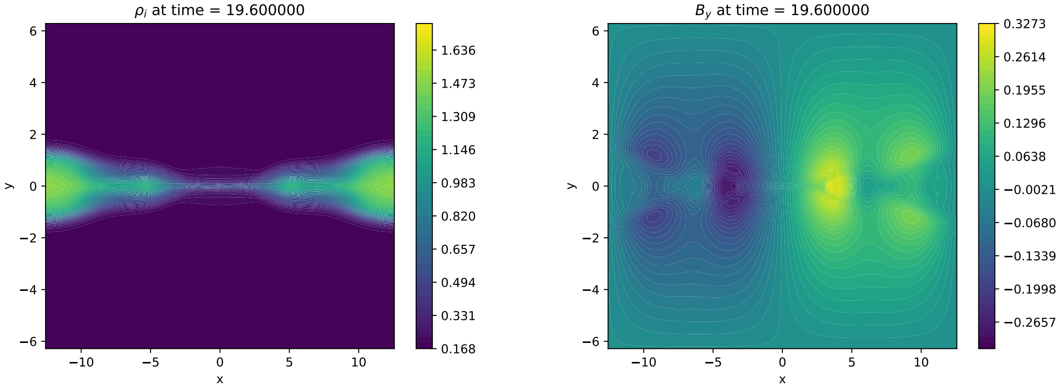


(a) 3rd Order 128x64 $\rho_i @ t = 19.6$ (b) 3rd Order 128x64 $B_y @ t = 19.6$



(c) 3rd Order 128x64 $\rho_i @ t = 23.6$ (d) 3rd Order 128x64 $B_y @ t = 23.6$

Figure 5.6: GEM Challenge with Moment-Slope Limiter at 25:1 Mass Ratio. (Simulations of the GEM Challange problem are performed with WARPXM version 0.0.2 / id_spatial_solver_sequential branch using input file gem_5moment_iman.py).

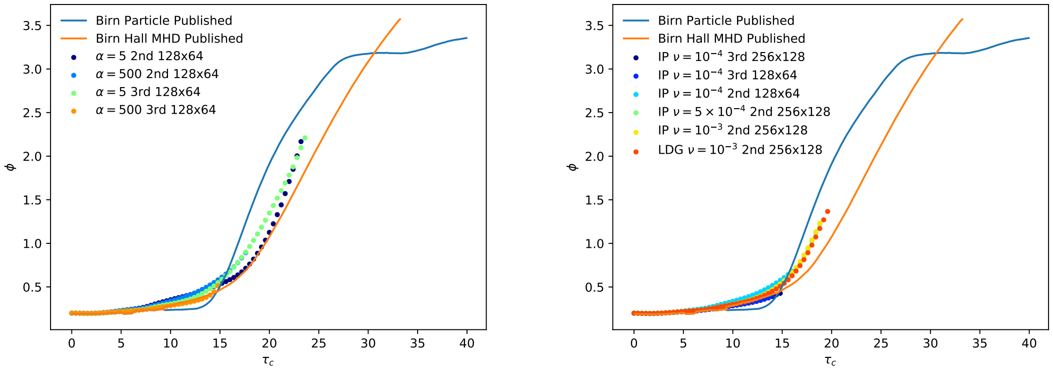


(a) 2nd Order 256x128 ρ_i

(b) 2nd Order 256x128 B_y

Figure 5.7: GEM Challenge with Artificial Viscosity Limiter at 25:1 Mass Ratio at $t = 19.6$

results, however. It is also interesting to note a major difference between the simulations. In the case where the moment-slope limiter is used, a plasmoid appears in the middle of the domain. However, in the simulation using the artificial viscosity limiter, no plasmoid forms. As with the moment-slope limiter, continued work on limiting could help overcome the instabilities and converge to a unique solution.



(a) Moe Limiter

(b) Artificial Viscosity Limiter

Figure 5.8: Gem Challenge reconnected flux with different limiters

Chapter 6

KINETIC MODELING

6.1 1D1V Landau Damping

Landau damping is a standard validation case for 1D1V kinetics implementations. It has been used in this work as well to check the initial kinetics implementation. In this problem, a Maxwellian is setup with a sinusoidal perturbation in position space, given by the form

$$f(x,v_x) = \frac{n_0}{(2\pi)^{\frac{1}{2}}\, v_{\text{th}}} \exp\left(-\frac{v_x^2}{2v_{\text{th}}^2}\right)\left(1+\alpha\cos\left(kx\right)\right), \tag{6.1.1}$$

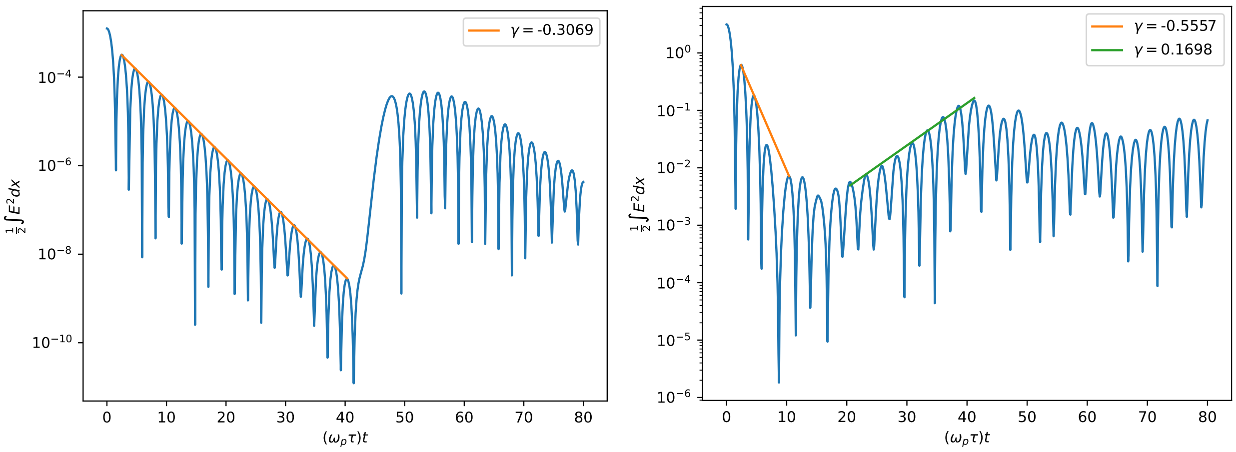
where $v_{\text{th}} = \sqrt{T/A}$. The simulation is performed on a domain of $x \in [-2\pi, 2\pi]$ and $v \in [-5, 5]$ subject to periodic boundary conditions in x and a no flux condition on the velocity boundaries. A consistent electric field is initialized by solving the Poisson equation subject to the periodic boundary condition

$$-\frac{1}{\left(\omega_p\tau\right)^2}\left(\frac{L}{\delta_p}\right)\frac{\partial^2\phi}{\partial x^2} = Z_i n - Z_i n_0 = Z_i n_0\left(1+\alpha\cos\left(kx\right)\right) - Z_i n_0, \tag{6.1.2}$$

where a uniformly distributed oppositely charged distribution is set to enforce net charge neutrality. Solving for the electrostatic potential and using the electric field definition gives

$$E_x(x) = +\, Z_i n_0 \left(\omega_p\tau\right)^2\left(\frac{\delta_p}{L}\right)\frac{\alpha}{k}\sin\left(kx\right). \tag{6.1.3}$$

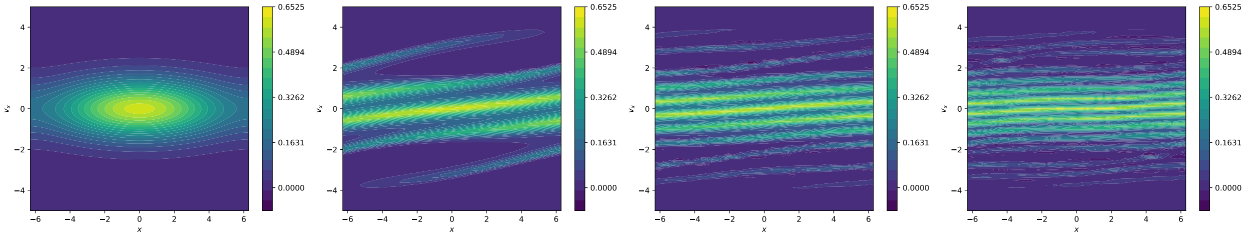
With these initial conditions the Vlasov-Maxwell system is solved using the DG method as described in Sec. 3.6. The parameters for this simulation are $n_0 = 1$, $v_{\text{th}} = 1$, $\alpha = 0.01$, $k = 0.5$, $\omega_p\tau = 1$, $\delta_p/L = 1$, which is run for 80 plasma periods using 40×40 elements with second order polynomials for the spatial and velocity domains, using the 4th order explicit Runge-Kutta timestepping scheme as shown in Eqs. (3.4.7) - (3.4.10). The damping rate is measured by plotting an integral of the electric field energy over time until recurrence occurs,



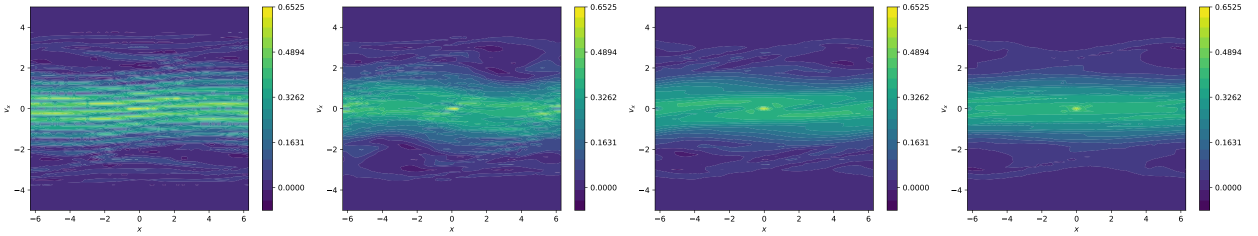
(a) Weak Landau Damping

(b) Strong Landau Damping

Figure 6.1: Evaluation of electric field energy for the weak and strong Landau damping problem in 1D1V. For weak Landau damping, a rate of −0.3069 is measured, compared with a theoretical rate of −0.3066. For strong Landau damping, a damping and growth rate of −0.5557 and 0.1698 are measured, compared with −0.562 and 0.168 respectively from published results. (Simulations of 1D1V Landau damping are performed with WARPXM version 1.2.6 / hybrid_kinetics branch using input file landau_damping.py).



(a) $(\omega_p \tau)\, t = 0$ (b) $(\omega_p \tau)\, t = 8$ (c) $(\omega_p \tau)\, t = 16$ (d) $(\omega_p \tau)\, t = 24$



(e) $(\omega_p \tau)\, t = 32$ (f) $(\omega_p \tau)\, t = 40$ (g) $(\omega_p \tau)\, t = 60$ (h) $(\omega_p \tau)\, t = 80$

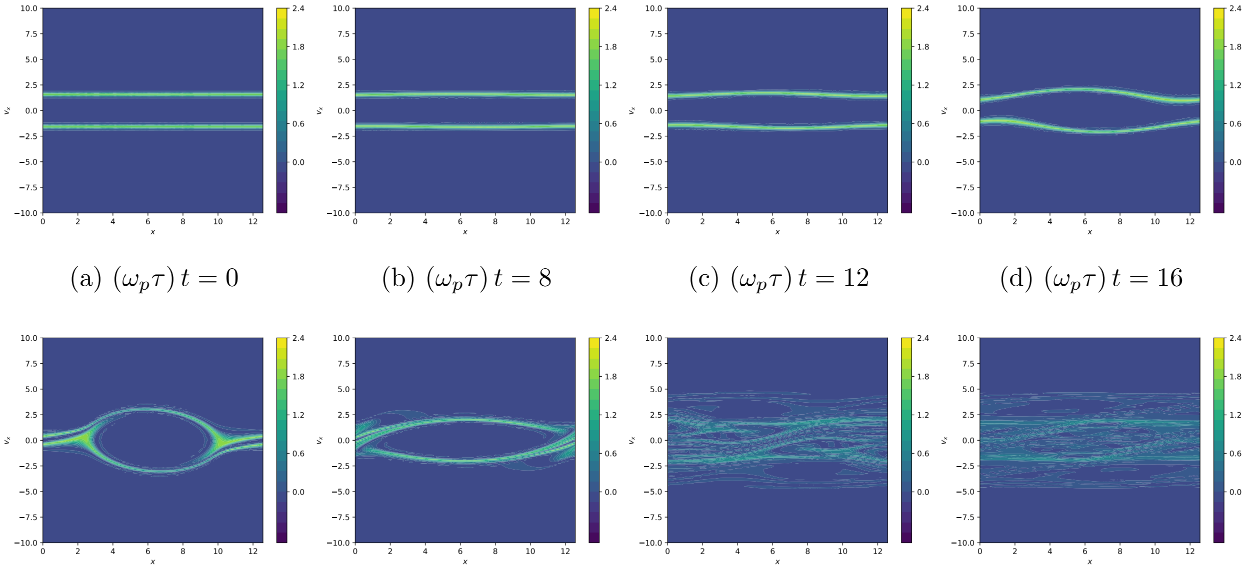
Figure 6.2: Distribution function f for the strong Landau Damping problem at different times for simulation with phase-space of $x \in [-2\pi, 2\pi]$ and $v_x \in [-5, 5]$ using $N_x \times N_{v_x} : 40 \times 40$ second order polynomial square elements and RK4 timestepping. Damping followed by growth occurs, as seen in Fig. 6.1b.

as shown in Fig. 6.1a. The calculated rate of -0.3069 compares well to the theoretical result of -0.3066 [23, 99].

For strong Landau damping, the perturbation is increased to $\alpha = 0.5$. The damping rate of -0.5557 and growth rate of 0.1698 shown in Fig. 6.1b match well with values from the published literature of -0.562 and 0.168, respectively [23]. Phase-space plots are shown in Fig. 6.2.

6.2 Two-Stream Instability

The two-stream instability is another electrostatic 1D1V test problem with a theoretical growth rate that can be used to validate the implementation of the continuum kinetic plasma



(a) $(\omega_p \tau)\, t = 0$ (b) $(\omega_p \tau)\, t = 8$ (c) $(\omega_p \tau)\, t = 12$ (d) $(\omega_p \tau)\, t = 16$

(e) $(\omega_p \tau)\, t = 20$ (f) $(\omega_p \tau)\, t = 24$ (g) $(\omega_p \tau)\, t = 60$ (h) $(\omega_p \tau)\, t = 80$

Figure 6.3: Distribution function f for the two-stream instability problem for $v_{\text{th}} = 0.1$ with separation $v' = \pi/2$ at different times for simulation with phase-space of $x \in [0, 4\pi]$ and $v_x \in [-10, 10]$ using $N_x \times N_{v_x} : 80 \times 160$ second order polynomial square elements and RK4 timestepping. Growth of the instability occurs at the expected rate until saturation at $(\omega_p \tau)\, t = 20$. See Fig. 6.4. (Simulations of the two-stream instability are performed with WARPXM version 1.2.6 / hybrid_kinetics branch using input file two_stream_instability.py).

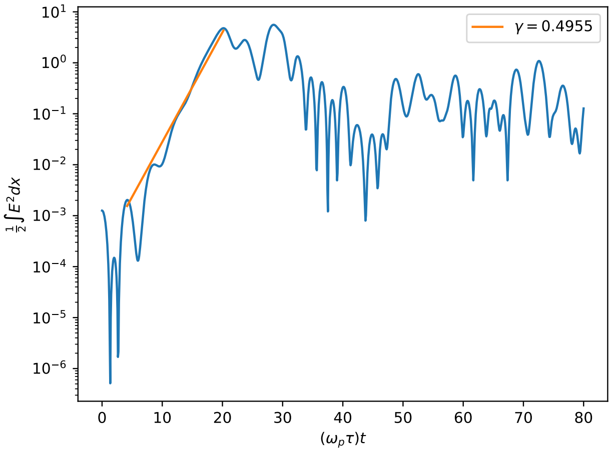


Figure 6.4: Evaluation of electric field energy for the two-stream instability problem. A growth rate of 0.4955 is measured, compared with the theoretical rate of 0.4952.

model. Two counter-streaming plasma beams are initialized as

$$f(x, v_x) = \frac{1}{2}\frac{n_0}{(2\pi)^{\frac{1}{2}}\, v_{\text{th}}}\left[\exp\left(-\frac{1}{2}\frac{(v_x - v')^2}{v_{\text{th}}^2}\right) + \exp\left(-\frac{1}{2}\frac{(v_x + v')^2}{v_{\text{th}}^2}\right)\right](1 + \alpha\cos(kx)). \tag{6.2.1}$$

With this form of f, the same initial electric field is used, Eq. (6.1.3). The separation of each beam from the axis is $v' = \pi/2$ with a perturbation $\alpha = 0.01$. The beams have $v_{\text{th}} = 0.1$, which is an approximation to cold beams ($v_{\text{th}} = 0$), for which the dispersion relation for the two-stream instability is

$$\frac{1}{(Y - X)^2} + \frac{1}{(Y + X)^2} = 1, \tag{6.2.2}$$

where $X = kv'/\omega_{ps}$, $Y = \omega/\omega_{ps}$, and $\omega_{ps}^2 = n_0 Z_s^2/A_s$ [7] for a single species s. The solution to Eq. (6.2.2) is

$$Y = \pm\sqrt{X^2 + 1 \pm \sqrt{4X^2 + 1}}. \tag{6.2.3}$$

The characteristics of this solution for Y are

$$\begin{cases} 0 \leq X < \sqrt{2} \begin{cases} \text{two roots are real} \\ \text{two roots are imaginary} \end{cases} \\ \sqrt{2} < X \text{ all four roots are real} \end{cases} . \tag{6.2.4}$$

For this problem, $n_0 = 1$, $k = 1/2$, $Z_s = 1$, $A_s = 1$ are set, leading to $\omega_{ps} = 1$ and $X = \pi/4$. The growing solution to Eq. (6.2.2) corresponds to imaginary Y in Eq. (6.2.3), found to be $Y = 0.4952i$.

A simulation is run at this unstable condition with an initial distribution as given in Eq. (6.2.1). 80×160 elements are used on $x \in [0, 2\pi/k]$ and $v_x \in [-10, 10]$ with second order polynomials and 4th order explicit Runge-Kutta timestepping as given in Eqs. (3.4.7) - (3.4.10). Normalization parameters $\omega_p \tau = 1$ and $\delta_p/L = 1$ are used. The calculated growth rate of this simulation, as shown in Fig. 6.4 is 0.4955, near the theoretical rate corresponding to cold beams. Figure 6.3 shows the evolution of the distribution function in phase space. Growth of the instability occurs until a characteristic time of 20 after which there is the expected saturation.

6.3 1D2V Weibel Instability

The Weibel instability is used to validate the 1D2V continuum kinetic implementation in WARPXM. This instability is usually described as being driven by an anisotropy in the distribution function in which a magnetic perturbation is made to grow [22]. Weibel first derived the theoretical growth [133]. Fried showed that the anisotropy could be understood as a superposition of counter-streaming beams with a growing transverse magnetic perturbation [45]. It is this case that is used for validation, though in the original analysis by Fried, he arrives at the same growth rate as Weibel assuming cold counter-streaming beams. For the case of two-counterstreaming electron beams, each with densities, $n_\text{e} = 1/2$ with opposing velocities ($\pm u_y$) in a background of non-moving ions with density $n_\text{i} = 1$ for quasineutrality, the growth rate of the magnetic field ($\gamma = \omega_i$) is derived as

$$\gamma = \frac{(\omega_c \tau)}{(\omega_p \tau)} \frac{\omega_{pe} u_y}{\left[1 + (\omega_c \tau)^2 \frac{\omega_{pe}^2}{k^2}\right]^{\frac{1}{2}}}, \tag{6.3.1}$$

where $\omega_{pe}^2 = \frac{2 n_e Z_e^2}{A_e}$ and it is assumed $\frac{u_y}{c/v_0} << 1$.

Cagas rederived this dispersion relation accounting for a thermal velocity of the beams and no assumption of $\frac{u_y}{c/v_0} << 1$ [19, 18]. Assuming two counterstreaming electron beams with $v_{\text{th}} = \sqrt{\frac{T_e}{A_e}}$ with the same opposing drift velocities, $\pm u_y$, the following electron distribution is initialized:

$$f_e = \frac{n_e}{2\pi v_{\text{th}}^2} \left[\exp\left(-\frac{v_x^2 + (v_y - u_y)^2}{2 v_{\text{th}}^2}\right) + \exp\left(-\frac{v_x^2 + (v_y + u_y)^2}{2 v_{\text{th}}^2}\right)\right]. \tag{6.3.2}$$

A neutralizing non-evolving background ion distribution with $n_i = 2n_e$ and the same thermal velocity is also initialized

$$f_i = \frac{n_i}{2\pi v_{\text{th}}^2} \exp\left(-\frac{v_x^2 + v_y^2}{2 v_{\text{th}}^2}\right). \tag{6.3.3}$$

The dispersion relation derived by Cagas under these conditions is

$$0 = 1 - \frac{(\omega_c \tau)^2 \omega_{pe}^2}{k_x^2} \left[\zeta Z(\zeta) \left(1 + \frac{u_y^2}{v_{\text{th}}^2}\right) + \frac{u_y^2}{v_{\text{th}}^2}\right] - \frac{2(\omega_c \tau)^2}{(\omega_p \tau)^2} v_{\text{th}}^2 \zeta^2. \tag{6.3.4}$$

where as with the cold beam case:

$$\omega_{pe}^2 = \frac{2 n_e Z_e^2}{A_e}. \tag{6.3.5}$$

Also

$$\zeta = \frac{\omega / k_x}{\sqrt{2} v_{\text{th}}} \tag{6.3.6}$$

and the plasma dispersion function is

$$Z(\zeta) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \frac{\exp(-x^2)}{x - \zeta} dx. \tag{6.3.7}$$

One can run this instability by perturbing the magnetic field, given by

$$B_z = B_0 \sin(k_x x). \tag{6.3.8}$$

Cases are run to match that of Cagas [19], where $k_x = 0.4$, $B_0 = 1 \times 10^{-4}$, $u_y = 0.15$ and 3 cases of $v_{\text{th}} = 0.15, 0.09, 0.03$. $(\omega_p \tau) = 1$ and $(\omega_c \tau) = 1$ are used in these simulations where the parameters are referenced to the electron and the ions were not evolved. To find the growth rate, Eq. (6.3.4) is solved for $\omega = \omega_r + i\omega_i$, where ω_r is the oscillation frequency and ω_i is the growth or damping rate. To handle poles in the plasma dispersion function in Eq. (6.3.7), it is calculated using analytic continuation, in which it is written as

$$Z(\zeta) = \frac{1}{\sqrt{\pi}} \left[\text{PV} \int_{-\infty}^{\infty} \frac{h(x)}{x - \zeta} dx \right] + i\pi h(\zeta) \times \begin{cases} 0, \text{Im}(\zeta) > 0 \\ 1, \text{Im}(\zeta) = 0 \\ 2, \text{Im}(\zeta) < 0 \end{cases}, \tag{6.3.9}$$

where $h(x) = \exp(-x^2)$ is the numerator in Eq. (6.3.7). Figures 6.5a, 6.5b, and 6.5c show the right hand side of Eq. (6.3.4). The solutions corresponding to growth ($+\omega_i = \gamma$) happen where $\omega_r = 0$. Newton iteration is used to solve for these solutions starting from a nearby point. This is aided with a derivative of Eq. (6.3.4) calculated as

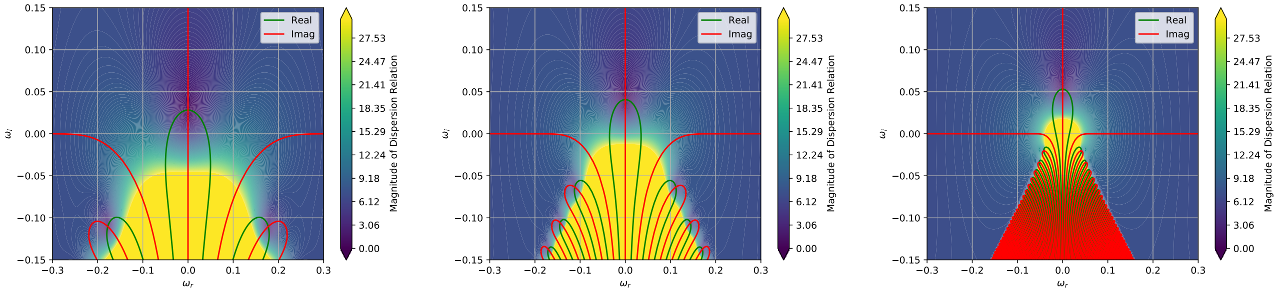
$$\frac{\partial F(\omega)}{\partial \omega} = -(\omega_c \tau)^2 \frac{\omega_{\text{pe}}^2}{\sqrt{2} v_{\text{th}} k_x^3} \left(1 + \frac{u_y^2}{v_{\text{th}}^2}\right) \left[Z(\zeta(\omega)) + \zeta(\omega) Z'(\zeta(\omega))\right] - \frac{(\omega_c \tau)^2}{(\omega_p \tau)^2} \frac{4 v_{\text{th}} \zeta(\omega)}{\sqrt{2} k_x}, \tag{6.3.10}$$

where

$$Z'(\zeta) = -\frac{2}{\sqrt{\pi}} \int_{-\infty}^{\infty} \frac{x \exp(-x^2)}{x - \zeta} dx. \tag{6.3.11}$$

Thus, the roots are found to be where

- for $v_{\text{th}} = 0.15$, $\gamma = 0.028167$
- for $v_{\text{th}} = 0.09$, $\gamma = 0.040592$



(a) $v_{\text{th}} = 0.15$ (b) $v_{\text{th}} = 0.09$ (c) $v_{\text{th}} = 0.03$

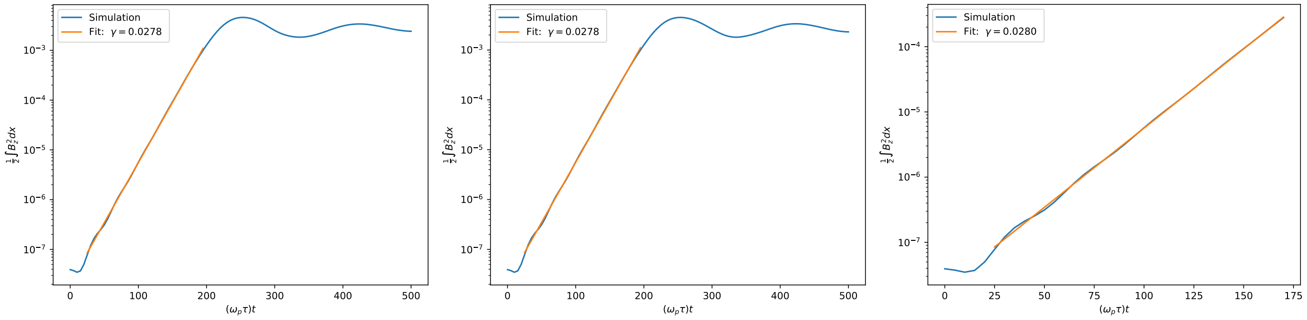
Figure 6.5: Magnitude of thermal dispersion relation for various v_{th} plotted at 250×250 resolution. For $v_{\text{th}} = 0.15$, $\gamma = 0.028167$, for $v_{\text{th}} = 0.09$, $\gamma = 0.040592$, and for $v_{\text{th}} = 0.03$, $\gamma = 0.053114$.

- for $v_{\text{th}} = 0.03$, $\gamma = 0.053114$

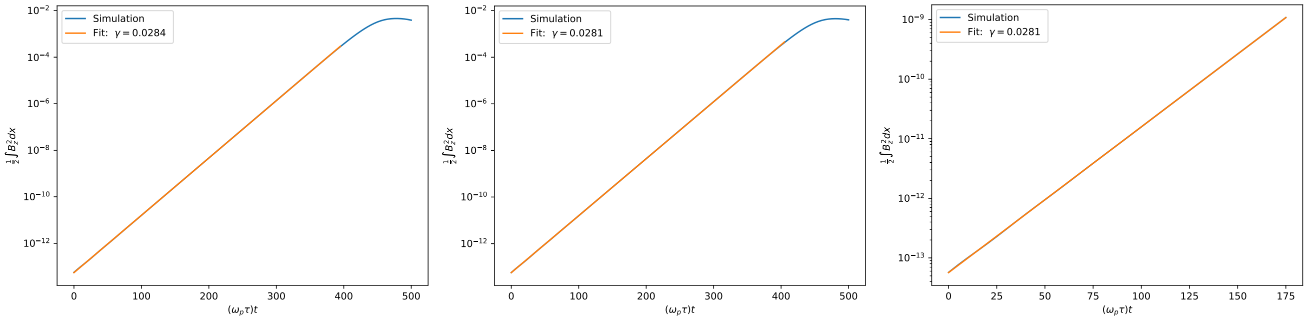
A convergence study is performed two ways:

- Case 1: The initial conditions presented in the above equations.
- Case 2: A restart after some time ($(\omega_p \tau)t = 50$ for $v_{\text{th}} = 0.03$ and $(\omega_p \tau)t = 100$ for other v_{th}) with scaled perturbations to allow for the initial condition to develop the dominant eigenmode.

The convergence test is performed on an $x \times v_x \times v_y$ grid with second order polynomial elements and RK4 timestepping with periodic boundary conditions in x and the outer boundaries in velocity space not being evolved. The domain is $x \in [0, 2\pi/k_x]$, $v_x \in [-10v_{\text{th}}, 10v_{\text{th}}]$, $v_y \in [-10v_{\text{th}}, 10v_{\text{th}}]$ (as already stated $k_x = 0.4$ and $u_y = \pm 0.15$ in these simulations). The convergence study increases the resolutions from $N_x \times N_{v_x} \times N_{v_y}$: $64 \times 32 \times 32$ to $128 \times 64 \times 64$ to $256 \times 128 \times 128$. The convergence studies for each case and for each thermal velocity are shown in Figs. 6.6, 6.7, and 6.8. Case 2 which starts from the eigenmode appears to converge faster than case 1 which starts from the Maxwellian distributions, though for $v_{\text{th}} = 0.03$



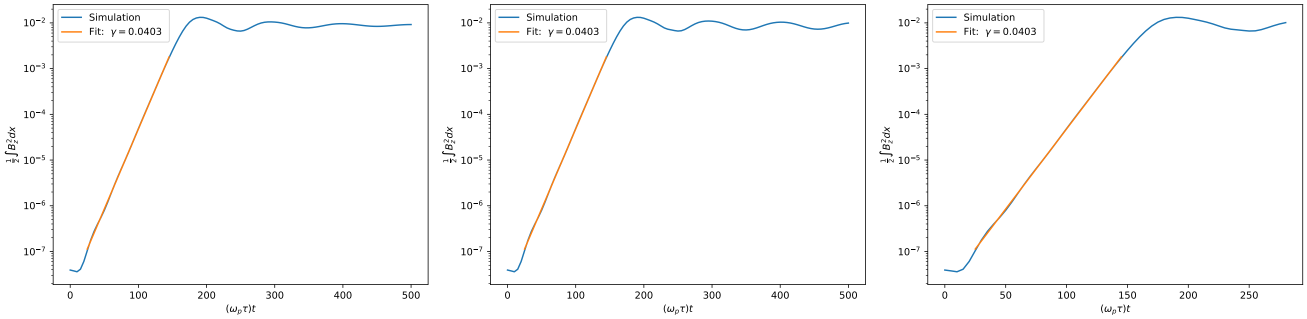
(a) 64 × 32 × 32 case 1 (b) 128 × 64 × 64 case 1 (c) 256 × 128 × 128 case 1



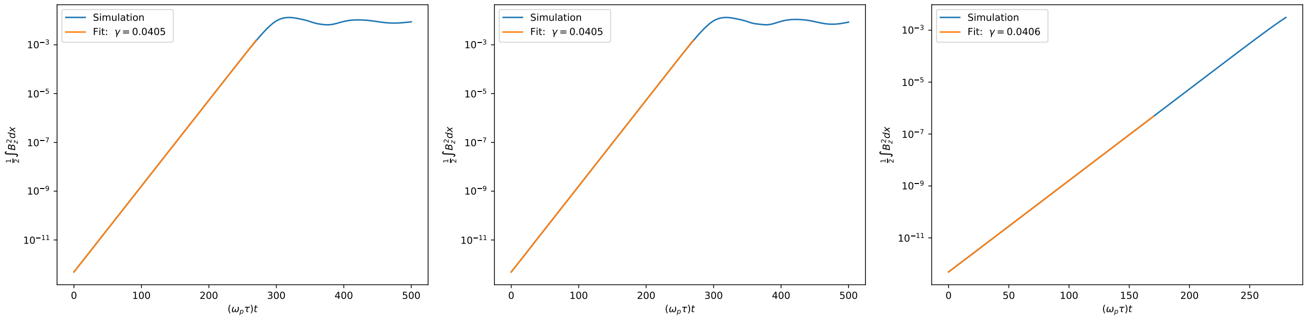
(d) 64 × 32 × 32 case 2 (e) 128 × 64 × 64 case 2 (f) 256 × 128 × 128 case 2

Figure 6.6: Convergence test for Weibel instability for $v_{\text{th}} = 0.15$. The theoretical growth rate is $\gamma_{\text{theory}} = 0.028167$. (Simulations of the Weibel instability are performed with WARPXM version 0.0.2 / new_kinetics branch using input files weibel_instability.py and weibel_instability_data_loading.py).

another mode seems to dominate after some time. Performing an FFT on B_z while this eigenmode is growing ($(\omega_p \tau)\, t = 200$), it is found that this mode has 3 times the wavenumber of the initial wave ($k_x = 1.2$, see Fig. 6.9) which shows both $k_x = 0.4$ and this eigenmode at this time. Solving the dispersion relation at $k_x = 1.2$ again using Newton iteration yields a growth rate of $\gamma = 0.102098$, close to the growth rate that is seen of roughly 0.095. Figure 6.9a shows the profile of B_z at $(\omega_p \tau)\, t = 200$. The $k_x = 1.2$ wavenumber appears to be the dominant mode at that time.

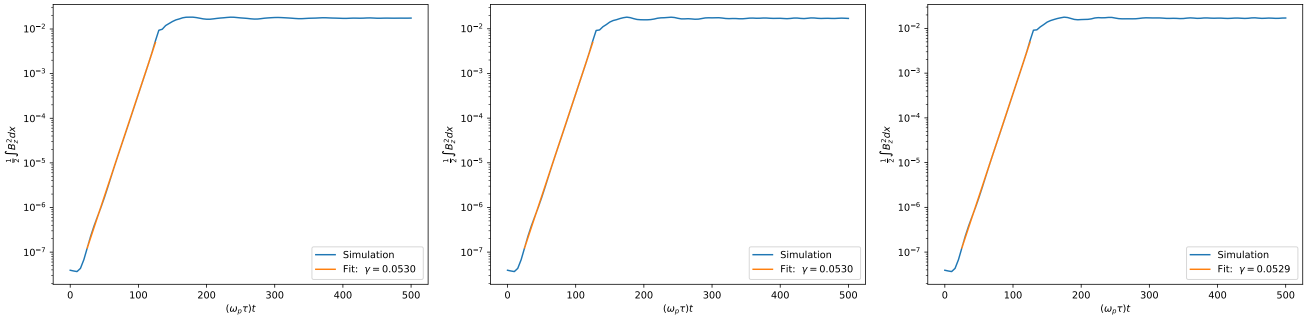


(a) $64 \times 32 \times 32$ case 1 (b) $128 \times 64 \times 64$ case 1 (c) $256 \times 128 \times 128$ case 1

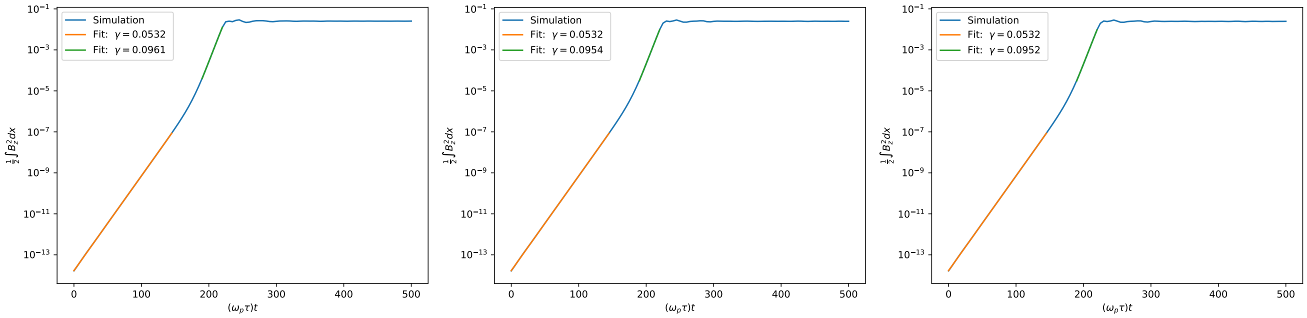


(d) $64 \times 32 \times 32$ case 2 (e) $128 \times 64 \times 64$ case 2 (f) $256 \times 128 \times 128$ case 2

Figure 6.7: Convergence test for Weibel instability for $v_{\text{th}} = 0.09$. The theoretical growth rate is $\gamma_{\text{theory}} = 0.040592$.

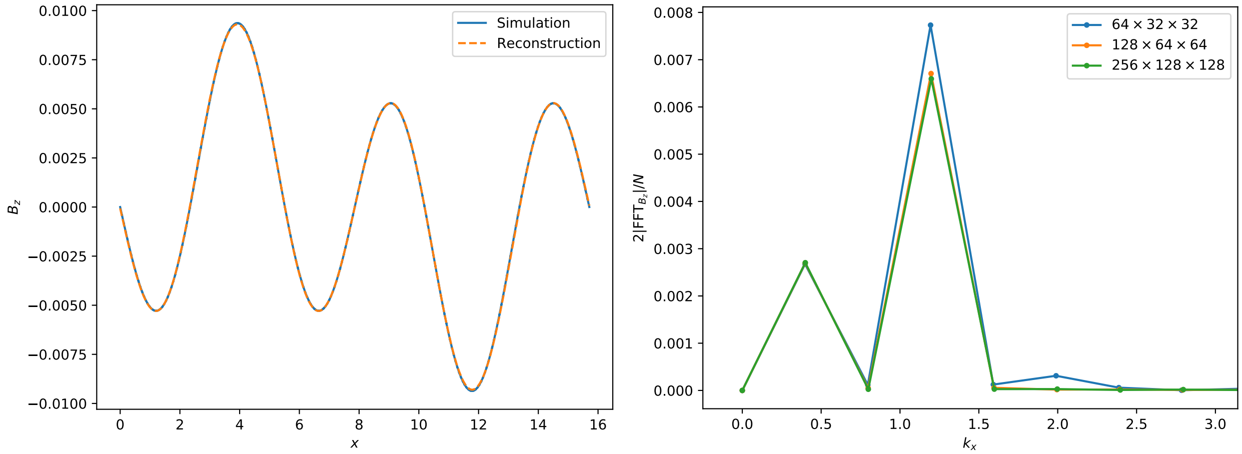


(a) 64 × 32 × 32 case 1 (b) 128 × 64 × 64 case 1 (c) 256 × 128 × 128 case 1



(d) 64 × 32 × 32 case 2 (e) 128 × 64 × 64 case 2 (f) 256 × 128 × 128 case 2

Figure 6.8: Convergence test for Weibel instability for $v_{\text{th}} = 0.03$. The theoretical growth rate is $\gamma_{\text{theory}} = 0.053114$.



(a) B_z for $256 \times 128 \times 128$ at $(\omega_p \tau)\, t = 200$.

(b) Magnitude of FFT for positive wavenumbers scaled by 2/N of B_z for various resolutions at $(\omega_p \tau)\, t = 200$. N is the number of samples taken for the FFT. The 3rd order DG representation is resampled by linear interpolation onto a uniformly distributed x coordinate where $N = 192$, 384, and 768 for $64 \times 32 \times 32$, $128 \times 64 \times 64$, and $256 \times 128 \times 128$ resolutions respectively.

Figure 6.9: Figure 6.9a shows B_z for $v_{\mathrm{th}} = 0.03$ for $256 \times 128 \times 128$ at $(\omega_p \tau)\, t = 200$. A reconstruction of this profile is also shown using the two strongest terms of the corresponding Fourier series at wavenumbers $k_x = 1.2$ and $k_x = 0.4$. Figure 6.9b shows the FFT of this B_z for $v_{\mathrm{th}} = 0.03$ for $(\omega_p \tau)\, t = 200$ for various resolutions for case 2, starting from a perturbed distribution. It can be seen here that these two wavenumbers are dominant.

6.4 1D2V Dory-Guest-Harris Instability

Electromagnetic analysis of the Dory-Guest-Harris (DGH) instability in 1D2V addresses a dearth of benchmark problems for electromagnetic theory used for Vlasov-Maxwell solvers. The DGH instability is known to arise from electrostatic waves propagating perpendicular to a magnetic field for certain probability distribution functions of a plasma, causing strong resonances at ion and electron cyclotron frequencies [116, 118]. The probability distribution functions are often of loss cone or ring-type in velocity space and are found in various situations ranging from magnetic mirror confinement devices [39, 91] to Earth's magnetosphere [59, 89, 108]. The excitation of waves at harmonics of the cyclotron frequency arising from the DGH instability can also be used to heat plasmas in magnetic confinement devices such as tokamaks [87] as well as for diagnostics to measure electron temperatures in these types of devices [117] or the magnetic field strength in the ionosphere [36].

Previous work used the DGH instability as a benchmark for a 1D2V continuum Vlasov-Poisson finite volume method by deriving a closed-form integral representation of the dispersion relation for perpendicular electrostatic waves and then comparing numerical solutions to quantify agreement and convergence [125]. This study is expanded to include electromagnetic effects by analyzing the DGH instability in the context of a Vlasov-Maxwell plasma model. The results show where the instability deviates from electrostatic behavior and produces electromagnetic effects associated with perturbations to the magnetic field. The rest of this section is outlined as follows. Section 6.4.1 derives the electromagnetic dispersion relation associated with perpendicular waves for probability distribution functions of the form $f(v_\perp)$. The electromagnetic and electrostatic dispersion relations are compared for several illustrative cases in Sec. 6.4.2. Section 6.4.3 highlights results from simulations and comparisons with the electromagnetic and electrostatic dispersion relations. Conclusions are given in Sec. 6.4.4. Note that in this section, s refers to a plasma species to avoid confusion with other terms involving the symbol α related to the DGH instability.

6.4.1 *Linear analysis of electromagnetic $k_{\parallel} = 0$ waves in uniformly magnetized plasma*

The electromagnetic dispersion relation for the DGH instability is derived by perturbing the Vlasov-Maxwell system about a spatially uniform equilibrium plasma state, $f_s^0(\boldsymbol{v})$, and a uniform magnetic field, $\boldsymbol{B}^0$, which leads to equilibrium cyclotron motion. The response of this equilibrium can be analyzed using the linearized form of the governing Eqs. (2.3.7), (2.3.10), and (2.3.11). In the linearization procedure, the fields and probability distribution function are expressed as a sum of the equilibrium and perturbation components: $\boldsymbol{E} = \boldsymbol{E}^0 + \boldsymbol{E}^1$, $\boldsymbol{B} = \boldsymbol{B}^0 + \boldsymbol{B}^1$, $f_s = f_s^0 + f_s^1$, where the perturbed quantities are assumed to be much smaller in magnitude than the equilibrium quantities such that nonlinear products of perturbations can be neglected [125]. For the electric field, it is sufficient to assume there is no equilibrium component ($\boldsymbol{E} = \boldsymbol{E}^1$). Except for strong equilibrium electric fields ($\boldsymbol{E}^0 \geq \frac{c}{v_0}\boldsymbol{B}^0$), it is always possible to set $\boldsymbol{E}^0 = 0$ to any arbitrary value by transforming to a different frame of reference [49] using the relationship $\boldsymbol{E}^0 + \boldsymbol{v} \times \boldsymbol{B}^0 = 0$, where $\boldsymbol{v}$ is the parameter for the Galilean transform of the distribution function. With the probability distribution function and fields being expressed through the summation of the equilibrium and perturbation components, the first-order linearized expansions of Eqs. (2.3.7), (2.3.10), and (2.3.11) are

$$\frac{\partial f_s^1}{\partial t} + \boldsymbol{v} \cdot \nabla f_s^1 + (\omega_c \tau) \frac{Z_s}{A_s} \left(\boldsymbol{v} \times \boldsymbol{B}^0\right) \cdot \nabla_{\boldsymbol{v}} f_s^1 + (\omega_c \tau) \frac{Z_s}{A_s} \left(\boldsymbol{E}^1 + \boldsymbol{v} \times \boldsymbol{B}^1\right) \cdot \nabla_{\boldsymbol{v}} f_s^0 = 0, \quad (6.4.1)$$

$$\frac{\partial \boldsymbol{E}^1}{\partial t} - \frac{(\omega_p \tau)^2}{(\omega_c \tau)^2} \nabla \times \boldsymbol{B}^1 = -\frac{(\omega_p \tau)^2}{(\omega_c \tau)} \boldsymbol{j}^1 = -\frac{(\omega_p \tau)^2}{(\omega_c \tau)} \sum_s Z_s \int \boldsymbol{v} f_s^1 (\boldsymbol{x}, \boldsymbol{v}, t)\, d\boldsymbol{v}, \quad (6.4.2)$$

$$\frac{\partial \boldsymbol{B}^1}{\partial t} + \nabla \times \boldsymbol{E}^1 = 0. \quad (6.4.3)$$

The azimuthal symmetry established by the equilibrium cyclotron motion about $\boldsymbol{B}^0$ leads to a more convenient formulation of the governing equations based on cylindrical velocity-space coordinates $(v_\perp, \phi, v_\parallel)$ with $\boldsymbol{B}^0 = B^0 \hat{\boldsymbol{z}}$ along the z-axis. This cylindrical velocity space can also be written using Cartesian coordinates through the transformation

$$\boldsymbol{v} = v_\perp \cos\phi \hat{\boldsymbol{x}} + v_\perp \sin\phi \hat{\boldsymbol{y}} + v_\parallel \hat{\boldsymbol{z}}, \quad (6.4.4)$$

which is aligned without loss of generality such that waves emanating from the perturbed system can be described by a wave vector given by $\boldsymbol{k} = k_\perp \hat{\boldsymbol{x}} + k_\parallel \hat{\boldsymbol{z}}$. Following the transformation shown in Ref. [49], the third term in Eq. (6.4.1) can be written in terms of the ϕ coordinate, yielding a modified version of this equation given by

$$\frac{\partial f_s^1}{\partial t} + \boldsymbol{v} \cdot \nabla f_s^1 - \omega_{c_s} (\omega_c \tau) \frac{\partial f_s^1}{\partial \phi} + (\omega_c \tau) \frac{Z_s}{A_s} \left(\boldsymbol{E}^1 + \boldsymbol{v} \times \boldsymbol{B}^1 \right) \cdot \nabla_{\boldsymbol{v}} f_s^0 = 0, \tag{6.4.5}$$

where a species cyclotron frequency is defined as

$$\omega_{c_s} \equiv \frac{Z_s}{A_s} B^0. \tag{6.4.6}$$

All perturbed quantities can be expressed in terms of Fourier-transformed plane waves at a particular wave number and frequency, e.g. $f_s^1 = \tilde{f}_s^1 \exp\left[i\left(\boldsymbol{k} \cdot \boldsymbol{r} - \omega t\right)\right]$, $\boldsymbol{E}^1 = \tilde{\boldsymbol{E}}^1 \exp\left[i\left(\boldsymbol{k} \cdot \boldsymbol{r} - \omega t\right)\right]$, where $\boldsymbol{r} = x\hat{\boldsymbol{x}} + y\hat{\boldsymbol{y}} + z\hat{\boldsymbol{z}}$. Substitution of these forms of the perturbations into Eqs. (6.4.5), (6.4.2), and (6.4.3) leads to

$$\left(-i\omega + i\boldsymbol{k} \cdot \boldsymbol{v}\right) \tilde{f}_s^1 - \omega_{c_s} (\omega_c \tau) \frac{\partial \tilde{f}_s^1}{\partial \phi} + \frac{Z_s}{A_s} (\omega_c \tau) \left(\tilde{\boldsymbol{E}}^1 + \boldsymbol{v} \times \tilde{\boldsymbol{B}}^1 \right) \cdot \nabla_{\boldsymbol{v}} f_s^0 = 0, \tag{6.4.7}$$

$$-i\omega \tilde{\boldsymbol{E}}^1 = \frac{(\omega_p \tau)^2}{(\omega_c \tau)^2} i\boldsymbol{k} \times \tilde{\boldsymbol{B}}^1 - \frac{(\omega_p \tau)^2}{(\omega_c \tau)} \tilde{\boldsymbol{j}}^1, \tag{6.4.8}$$

$$-i\omega \tilde{\boldsymbol{B}}^1 = -i\boldsymbol{k} \times \tilde{\boldsymbol{E}}^1. \tag{6.4.9}$$

The dispersion relation is obtained by eliminating $\tilde{\boldsymbol{B}}^1$ from Eqs. (6.4.7) and (6.4.8) using Eq. (6.4.9), solving Eq. (6.4.7) for $\tilde{f}_s^1$, and substituting into Eq. (6.4.8) through the current density term, calculated as

$$\tilde{\boldsymbol{j}}^1 = \sum_s Z_s \int_0^{2\pi} \int_{-\infty}^{\infty} \int_0^{\infty} \boldsymbol{v} \tilde{f}_s^1 v_\perp dv_\perp dv_\parallel d\phi \equiv \overline{\overline{\sigma}} \cdot \tilde{\boldsymbol{E}}^1. \tag{6.4.10}$$

This leads to an equation of the form

$$\left(\overline{\overline{I}} + \overline{\overline{\chi}} \right) \cdot \tilde{\boldsymbol{E}}^1 + \frac{(\omega_p \tau)^2}{(\omega_c \tau)^2} \frac{\boldsymbol{k} \times \boldsymbol{k} \times \tilde{\boldsymbol{E}}^1}{\omega^2} = 0, \tag{6.4.11}$$

where $\overline{\overline{\chi}}$ is the susceptibility tensor and is related to the conductivity tensor $\overline{\overline{\sigma}}$ through the relation

$$\overline{\overline{\chi}} = -\frac{(\omega_p \tau)^2}{i\omega\,(\omega_c \tau)} \overline{\overline{\sigma}}. \tag{6.4.12}$$

Evaluation of Eq. (6.4.11) requires the solution to Eq. (6.4.7). In this work, the problem of interest is described by an equilibrium distribution of the form $f_s^0(v_\perp)$ which is independent of $v_\parallel$ such that $\frac{\partial f_s^0}{\partial v_\parallel} = 0$. In addition, only waves propagating perpendicular to $\boldsymbol{B}^0$ are considered such that $k_\parallel = 0$. These simplifications, combined with the elimination of $\tilde{\boldsymbol{B}}^1$ through Eq. (6.4.9) as discussed above and some mathematical manipulation leads to a form of Eq. (6.4.7) given by

$$\frac{\partial \tilde{f}_s^1}{\partial \phi} - i\left(-\alpha_s + \beta_s \cos\phi\right) \tilde{f}_s^1 = \frac{\omega_{p_s}^2}{Z_s \omega_{c_s}} \frac{\partial F_s^0}{\partial v_\perp} \left[\left(\tilde{E}_x^1 + \frac{k_\perp v_\parallel}{\omega} \tilde{E}_z^1 \right) \cos\phi + \tilde{E}_y^1 \sin\phi \right], \tag{6.4.13}$$

where the following variables have been defined:

$$\alpha_s \equiv \frac{\omega}{(\omega_c \tau)\,\omega_{c_s}}, \tag{6.4.14a}$$

$$\beta_s \equiv \frac{k_\perp v_\perp}{(\omega_c \tau)\,\omega_{c_s}}, \tag{6.4.14b}$$

$$F_s^0 \equiv \frac{f_s^0}{n_s}, \tag{6.4.14c}$$

$$\omega_{p_s}^2 \equiv \frac{Z_s^2 n_s}{A_s}. \tag{6.4.14d}$$

After multiplying Eq. (6.4.13) with an integrating factor, the expression is then integrated over an unperturbed cyclotron orbit. The resulting integral is evaluated by expanding exponential terms into infinite series of Bessel functions using the identity

$$\exp\left(-i\beta_s \sin\phi\right) = \sum_{n=-\infty}^{\infty} \exp\left(-in\phi\right) J_n\left(\beta_s\right), \tag{6.4.15}$$

where J_n is the Bessel function of the first kind of order n. The expansion simplifies the expression for the perturbed distribution function through the orthogonality property of the

Bessel functions [49, 112]. The result of the integration is

$$\tilde{f}_s^1 = i\frac{\omega_{p_s}^2}{Z_s\omega_{c_s}}\exp\left(i\beta_s\sin\phi\right)\sum_{n=-\infty}^{\infty}\frac{\exp\left(-in\phi\right)}{-\alpha_s+n}\frac{\partial F_s^0}{\partial v_\perp}\left[\frac{n}{\beta_s}J_n\left(\beta_s\right)\tilde{E}_x^1 + iJ_n^{'}\left(\beta_s\right)\tilde{E}_y^1 + \frac{n}{\beta_s}\frac{k_\perp v_\parallel}{\omega}J_n\left(\beta_s\right)\tilde{E}_z^1\right]. \tag{6.4.16}$$

Equation (6.4.16) is then substituted into Eq. (6.4.10) and combined with Eq. (6.4.12) to yield

$$\overline{\overline{\chi}} = -\frac{2\pi\left(\omega_p\tau\right)^2}{\omega\left(\omega_c\tau\right)}\sum_s\frac{\omega_{p_s}^2}{\omega_{c_s}}\sum_{n=-\infty}^{\infty}\int\limits_0^\infty\frac{\partial F_s^0}{\partial v_\perp}\begin{pmatrix}\frac{n^2}{\beta_s^2}J_n^2\left(\beta_s\right) & i\frac{n}{\beta_s}J_n\left(\beta_s\right)J_n^{'}\left(\beta_s\right)\\ -i\frac{n}{\beta_s}J_n\left(\beta_s\right)J_n^{'}\left(\beta_s\right) & J_n^{'}\left(\beta_s\right)J_n^{'}\left(\beta_s\right)\end{pmatrix}\frac{v_\perp^2}{-\alpha_s+n}dv_\perp, \tag{6.4.17}$$

where in this work, all physics of interest is independent of $v_\parallel$, which is thus set to 0. Substitution of Eq. (6.4.17) into Eq. (6.4.11) yields

$$\begin{pmatrix}K_{11} & K_{12} & 0\\ K_{21} & K_{22}-\frac{\left(\omega_p\tau\right)^2}{\left(\omega_c\tau\right)^2}\frac{k_\perp^2}{\omega^2} & 0\\ 0 & 0 & 1-\frac{\left(\omega_p\tau\right)^2}{\left(\omega_c\tau\right)^2}\frac{k_\perp^2}{\omega^2}\end{pmatrix}\cdot\tilde{\boldsymbol{E}}_1 = 0, \tag{6.4.18}$$

where

$$K_{11} = 1+\overline{\overline{\chi}}_{11}, \tag{6.4.19a}$$

$$K_{12} = \overline{\overline{\chi}}_{12}, \tag{6.4.19b}$$

$$K_{21} = \overline{\overline{\chi}}_{21}, \tag{6.4.19c}$$

$$K_{22} = 1+\overline{\overline{\chi}}_{22}. \tag{6.4.19d}$$

Note the z-component of Eq. (6.4.18) decouples and produces a dispersion relation describing the propagation of light waves,

$$\frac{\omega^2}{k_\perp^2} = \frac{\left(\omega_p\tau\right)^2}{\left(\omega_c\tau\right)^2} = \frac{\frac{q_0^2 n_0}{\epsilon_0 m_0}}{\frac{q_0^2 B_0^2}{m_0^2}} = \frac{n_0 m_0}{\epsilon_0 B_0^2} = \frac{c^2\mu_0 n_0 m_0}{B_0^2} = \frac{c^2}{v_0^2}. \tag{6.4.20}$$

If the assumption of $v_\parallel = 0$ is not made, the z-component of Eq. (6.4.18) would produce an additional term leading to the "ordinary" wave (O-wave), but would still decouple from the rest of the system [40]. Taking the determinant of the remaining 2×2 matrix in Eq. (6.4.18) yields another dispersion relation given by

$$D(\omega, k_\perp) \equiv K_{11}\left(K_{22} - \frac{\left(\omega_p\tau\right)^2}{\left(\omega_c\tau\right)^2}\frac{k_\perp^2}{\omega^2}\right) - K_{12}K_{21} = 0. \tag{6.4.21}$$

By inspection, Eq. (6.4.21) has poles at $n = \alpha_s$ corresponding to resonances of the cyclotron frequency as well as a double pole at $\omega = 0$. Comparison with the electrostatic dispersion relation presented in Ref. [125] shows that this pole at $\omega = 0$ does not exist in the electrostatic limit.

The solution to Eq. (6.4.21) requires the evaluation of Eq. (6.4.17), for which the infinite series in $\overline{\overline{\chi}}$ must be truncated to some finite number of terms. However, the series can be removed entirely in favor of Bessel functions of specific orders by employing the Lerche-Newberger sum rule, given by

$$\sum_{n=-\infty}^{\infty}\frac{\left(-1\right)^n J_{a-cn}\left(z\right) J_{b+cn}\left(z\right)}{n+d} = \frac{\pi}{\sin\left(\pi d\right)} J_{a+cd}\left(z\right) J_{b-cd}\left(z\right), \tag{6.4.22}$$

where d is any non-integer complex number, $\mathrm{Re}(a+b) > -1$, and $0 < c \leq 1$ [67, 68, 83, 84, 69]. Manipulations of the Lerche-Newberger sum rule to remove each infinite series in Eq. (6.4.17) are shown in Ref. [115]. The invocation of the infinite series could also be avoided entirely in solving for $\tilde{f}_s^1$ in Eq. (6.4.13) by taking advantage of the symmetry in particle orbit around the magnetic field during the integration, as shown in Refs. [94, 95]. Either approach leads to a form of $\overline{\overline{\chi}}$ given by

$$\overline{\overline{\chi}} = -\frac{2\pi\left(\omega_p\tau\right)^2}{\omega\left(\omega_c\tau\right)}\sum_s\frac{\omega_{p_s}^2}{\omega_{c_s}}\int_0^\infty\frac{\partial F_s^0}{\partial v_\perp}\begin{pmatrix}\frac{\alpha_s}{\beta_s^2}\left(1-Q_s\right) & -\frac{i}{2\beta_s}Q_s' \\ \frac{i}{2\beta_s}Q_s' & -\left(\frac{\pi}{\sin(\pi\alpha_s)}J_{-\alpha_s}'\left(\beta_s\right)J_{\alpha_s}'\left(\beta_s\right)+\frac{\alpha_s}{\beta_s^2}\right)\end{pmatrix}v_\perp^2\,dv_\perp, \tag{6.4.23}$$

where

$$Q_s \equiv \frac{\pi\alpha_s}{\sin\left(\pi\alpha_s\right)}J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right), \tag{6.4.24a}$$

$$Q_s' \equiv \frac{\pi \alpha_s}{\sin(\pi \alpha_s)} \frac{\partial}{\partial \beta_s} \left[J_{-\alpha_s}(\beta_s) J_{\alpha_s}(\beta_s) \right]. \tag{6.4.24b}$$

Equation (6.4.23) is identical to Eq. (6.4.17), but has the advantage of removal of the infinite series. Resonances at integer multiples of the cyclotron frequency are also readily apparent through the $\sin(\pi\alpha_s)$ terms in the denominators [94], higher multiples of which would need a large number of terms in the series expansion to accurately resolve $\overline{\overline{\chi}}$.

The expressions for Q_s, Q_s', and $J_{-\alpha_s}' J_{\alpha_s}'$ can be recast in a more convenient form in terms of integrals of real low-order Bessel functions, given by

$$J_{-\alpha_s}(\beta_s) J_{\alpha_s}(\beta_s) = \frac{2}{\pi} \int_0^{\frac{\pi}{2}} J_0(2\beta_s \cos\theta) \cos(2\alpha_s \theta)\, d\theta, \tag{6.4.25a}$$

$$\frac{\partial}{\partial \beta_s} \left[J_{-\alpha_s}(\beta_s) J_{\alpha_s}(\beta_s) \right] = -\frac{4}{\pi} \int_0^{\frac{\pi}{2}} J_1(2\beta_s \cos\theta) \cos\theta \cos(2\alpha_s \theta)\, d\theta, \tag{6.4.25b}$$

$$J_{-\alpha_s}'(\beta_s) J_{\alpha_s}'(\beta_s) = \frac{1}{\pi} \left[\int_0^{\frac{\pi}{2}} \cos(2\alpha_s \theta) \left[J_2(2\beta_s \cos\theta) - J_0(2\beta_s \cos\theta) \cos(2\theta) \right] d\theta - \frac{\alpha_s}{\beta_s^2} \sin(\pi \alpha_s) \right]. \tag{6.4.25c}$$

Details of this derivation can be found in Appendix H.

By specifying the equilibrium distribution functions, specifically $\frac{\partial f_s^0}{\partial v_\perp}$, the system of equations given by Eqs. (6.4.19), (6.4.21), and (6.4.23) provides a closed-form implicit expression for the electromagnetic dispersion relation for the DGH instability. The integrals in Eq. (6.4.25), which are needed by Eq. (6.4.23), can be evaluated to arbitrary accuracy using numerical quadrature.

While the derivation in this section is restricted to $k_\parallel = 0$ waves and to distributions of the form $f_s^0(v_\perp)$, the procedure can be extended to evaluate the full susceptibility tensor, written for infinite sums of real integer order Bessel functions as shown in Ref. [49] or products of complex order Bessel functions as shown in Ref. [94], where these restrictions are not made. The integrals in Eq. (6.4.25) are sufficient to calculate the full susceptibility tensor shown in Ref. [94] where $k_\parallel v_\parallel$ is incorporated into Eq. (6.4.14a) and where a parallel velocity dependence is added to the equilibrium distribution function such that $f_s^0 = f_s^0(v_\perp, v_\parallel)$. The numerical integration of the resulting equation similar to Eq. (6.4.23) would have the added complication of poles resulting from $v_\parallel$ in denominators when there is finite $k_\parallel$, which can be handled using analytic continuation techniques as shown by Landau [65] and Bernstein [5] for unmagnetized and magnetized waves, respectively.

The generalization of the wave vector from $\boldsymbol{k} = k_\perp \hat{\boldsymbol{x}}$ and the distribution function velocity dependence from $f_s^0(v_\perp)$ also provides a natural development path to benchmark kinetic codes in higher dimensions beyond 1D2V. The extension to two spatial dimensions can be realized by generalizing $k_\perp$ to the $x - y$ plane such that $\boldsymbol{k} = k_{\perp,x}\hat{\boldsymbol{x}} + k_{\perp,y}\hat{\boldsymbol{y}}$, which allows for benchmarking of 2D2V codes. Allowing for parallel velocity dependence of the equilibrium distribution function, such that $f_s^0 = f_s^0(v_\perp, v_\parallel) = f_s^0(v_x, v_y, v_z)$, and adding a parallel component to the wave vector such that it lies in the $x-z$ plane where $\boldsymbol{k} = k_\perp\hat{\boldsymbol{x}} + k_\parallel\hat{\boldsymbol{z}}$ leads to a 3×3 susceptibility tensor that can be calculated as described above and can be used to benchmark 2D3V codes simulating oblique wave propagation. Further generalization of the wave vector such that $\boldsymbol{k} = k_{\perp,x}\hat{\boldsymbol{x}} + k_{\perp,y}\hat{\boldsymbol{y}} + k_\parallel\hat{\boldsymbol{z}}$ analogous to the extension from 1D2V to 2D2V can also allow for benchmarking of 3D3V codes.

6.4.2 Calculation of the dispersion relation for specific forms of $f_s^0(v_\perp)$, values of $k_\perp$, and ω_p/ω_c ratio

The electromagnetic dispersion relation derived in Sec. 6.4.1 can be used to determine the linear behavior of an equilibrium ring distribution in conditions leading to the DGH instability. The specific form of the ring distribution studied by Dory et al. [39] is given by

$$f^0(v_\perp) = \frac{1}{\pi\alpha_\perp^2 j!}\left(\frac{v_\perp^2}{\alpha_\perp^2}\right)^j \exp\left(-\frac{v_\perp^2}{\alpha_\perp^2}\right), \tag{6.4.26}$$

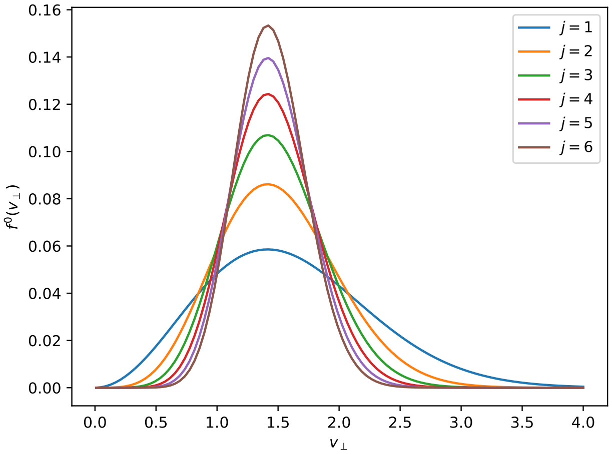


Figure 6.10: Equilibrium ring distribution function for the DGH instability for various j. The distribution is peaked at $v_{\perp 0} = j^{1/2}\alpha_\perp$. The thickness of the distribution can be measured by its half-width at half-maximum, approximated by $\alpha_\perp/2$, which scales with $j^{-1/2}$ for a given $v_{\perp 0}$. Distribution functions with $j = 1, 2, 6$ correspond to cases A-D where $v_{\perp 0} = \sqrt{2}$ as described in Table 6.1.

which has a derivative which can be calculated analytically, yielding

$$\frac{\partial f^0}{\partial v_\perp} = \frac{1}{\pi\alpha_\perp^2 j!}\frac{2v_\perp}{\alpha_\perp^2}\exp\left(-\frac{v_\perp^2}{\alpha_\perp^2}\right)\left(\frac{v_\perp^2}{\alpha_\perp^2}\right)^{j-1}\left[j - \left(\frac{v_\perp^2}{\alpha_\perp^2}\right)\right]. \tag{6.4.27}$$

The peak velocity of the distribution is given by $v_{\perp 0} = j^{1/2}\alpha_\perp$ while the half-width at half-maximum (HWHM) is approximated by $\alpha_\perp/2$[1] which is incorrectly presented in Ref. [125] and in Ref. [39]. This relationship between j and distribution width for a fixed $v_{\perp 0}$ can be seen in Fig. 6.10. The distribution is found to be unstable if sufficiently peaked, the ratio of plasma frequency to cyclotron frequency is above a certain value known as the density threshold, and the wave number falls within a specified range [125, 39].

---

[1] The peak velocity of f^0 is determined by finding the positive root of $f^{0\prime} = \frac{\partial f^0}{\partial v_\perp} = 0$, yielding $v_{\perp 0} \equiv v_\perp(f^{0\prime} = 0) = j^{1/2}\alpha_\perp$. An approximation of the distribution width, which is used for the estimation of the half-width at half-maximum and thermal velocity, is determined by finding the inflection points on either side of the peak, calculated from roots of $f^{0\prime\prime}(v_\perp) = \frac{\partial^2 f^0}{\partial v_\perp^2} = 0$, where the term $\sqrt{16j+1}$ is simplified by assuming $16j >> 1$, yielding $v_\perp(f^{0\prime\prime} = 0) \approx v_{\perp 0} \pm \alpha_\perp/2$.

Case	j	ω_p/ω_c	$\tilde{k}$	$\alpha_\perp$	HWHM	β
A	6	20	3.15	$\sqrt{1/3}$	$\sqrt{1/3}/2$	1/6
B	6	20	4.65	$\sqrt{1/3}$	$\sqrt{1/3}/2$	1/6
C	2	10	2.12	1	1/2	1/2
D	1	1	2.12	$\sqrt{2}$	$\sqrt{2}/2$	1

Table 6.1: Parameters for distribution f^0 for the various cases under consideration, each with $v_{\perp 0} = \sqrt{2}$ and $B^0 = 1$. Cases A-C correspond to cases in Ref. [125] while Case D is introduced with lower j and ω_p/ω_c, which is stable according to both the electromagnetic and electrostatic dispersion relations.

To illustrate the electromagnetic extension of the DGH instability, four specific cases are examined of a ring distribution of electrons in the presence of a static neutralizing background of ions ($A_\mathrm{e} = 1$, $Z_\mathrm{e} = -1$, $f_\mathrm{e}^0 = f^0$, $f_\mathrm{i}^1 = 0$). The parameters of this distribution for these cases are shown in Table 6.1 where $\tilde{k} \equiv k_\perp v_{\perp 0} \frac{\omega_p}{\omega_c}$ is a normalized wave number. For all cases $v_{\perp 0} = \sqrt{2}$ and $B^0 = 1$. Cases A-C are identical to cases examined in Ref. [125], where case A was found to have a purely growing instability, case B was found to be unstable with an oscillatory behavior, and case C was found to be stable according to the electrostatic dispersion relation. These same cases are re-examined using the more general electromagnetic theory. Case D is introduced with decreased j, widening the distribution which reduces peakedness, and a lower plasma to cyclotron frequency ratio which further moves the plasma below the density threshold for instability. The DGH mode is completely stabilized when $j = 0$, corresponding to Maxwellian distribution functions [39, 125]. Table 6.1 also calculates plasma beta of the electrons, given by

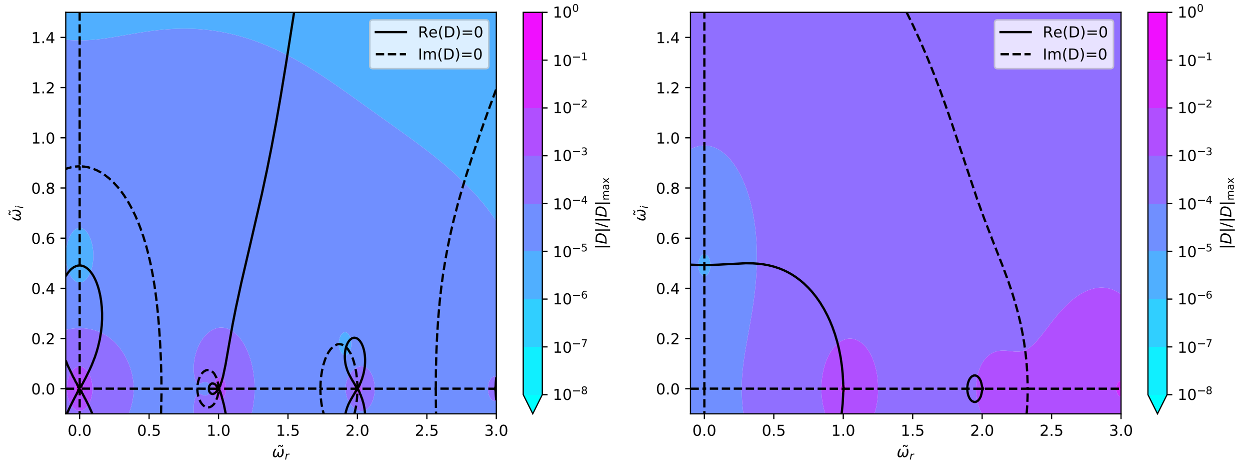
$$\beta = \frac{2nT}{(B^0)^2} = \frac{n\alpha_\perp^2 A}{2\,(B^0)^2}, \tag{6.4.28}$$

where T is calculated from the thermal velocity which is approximated by the half-width at half-maximum of the distribution given by $v_\mathrm{th} = \sqrt{T/A} = \alpha_\perp/2$, and where $n = 1$ and $A = 1$

for electrons. The electrostatic approximation is valid for $\beta << 1$ [77], which inspection of Eq. (6.4.28) shows becomes less accurate for hotter distributions.

The electromagnetic dispersion relation for each case is examined and compared with the electrostatic dispersion relation derived in Ref. [125]. The characteristic time is set to $\tau \equiv \omega_p^{-1}$, leading to $\omega_p \tau = 1$ and $\omega_c \tau = (\omega_p/\omega_c)^{-1}$. Integrations required in the electromagnetic dispersion relation as derived in Sec. 6.4.1 as well as those required in the electrostatic dispersion relation as derived in Ref. [125] are performed using the second-order accurate trapezoidal rule [113] using 100 points. The electromagnetic and electrostatic dispersion relations for cases A, B, C, and D are plotted in Figs. 6.11, 6.12, 6.13, and 6.14, respectively, where the frequencies are normalized to the cyclotron frequency ($\tilde{\omega} = \omega \frac{\omega_p}{\omega_c}$). Shown also are zero contours of the real and imaginary components of the dispersion relation. Solutions exist where the real and imaginary zero contours intersect, satisfying $D(\omega, k_\perp) = 0$, but not at the poles of $D(\omega, k_\perp)$ indicated by $|D| \, / \, |D|_{\max}$ approaching unity. After the general vicinity of a solution is found, more precise calculations are performed with a Newton–Raphson method, using more points in the trapezoidal rule of integration until convergence to $\tilde{\omega}$ within 10^{-4} is reached.

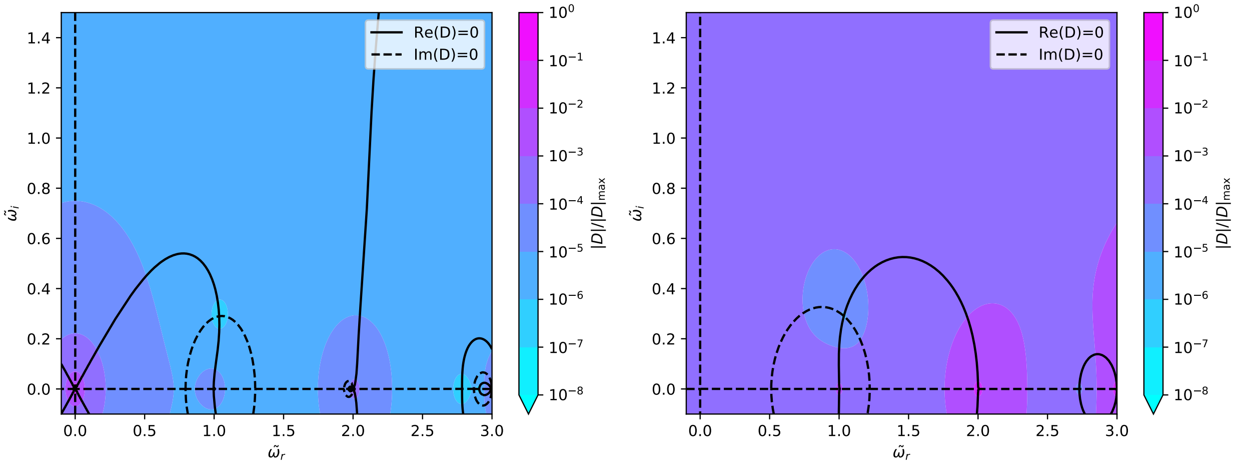
The complex frequencies from the electrostatic and electromagnetic dispersion relations are found to be close for cases A and B. However for case C, the results and behavior characteristics deviate. The electromagnetic dispersion relation shows growth and oscillation while the electrostatic dispersion relation produces multiple purely real frequencies. This discrepancy can by understood by the high plasma beta of the distribution, as seen in Table 6.1, indicating the inaccuracy of the electrostatic theory. Case D yields purely real frequencies for both the electrostatic and electromagnetic dispersion relations. The absence of growth can be explained by the further widening of the distribution and decreased plasma to cyclotron frequency ratio compared to the other cases. Even with the increased plasma beta of this distribution indicating inaccuracy of the electrostatic theory, the combination of increased temperature and reduced frequency ratio makes this case stable even using the electromagnetic dispersion relation, which is not true in case C.



(a) Case A EM, $\tilde{\omega} = 0.4912i$

(b) Case A ES, $\tilde{\omega} = 0.4912i$

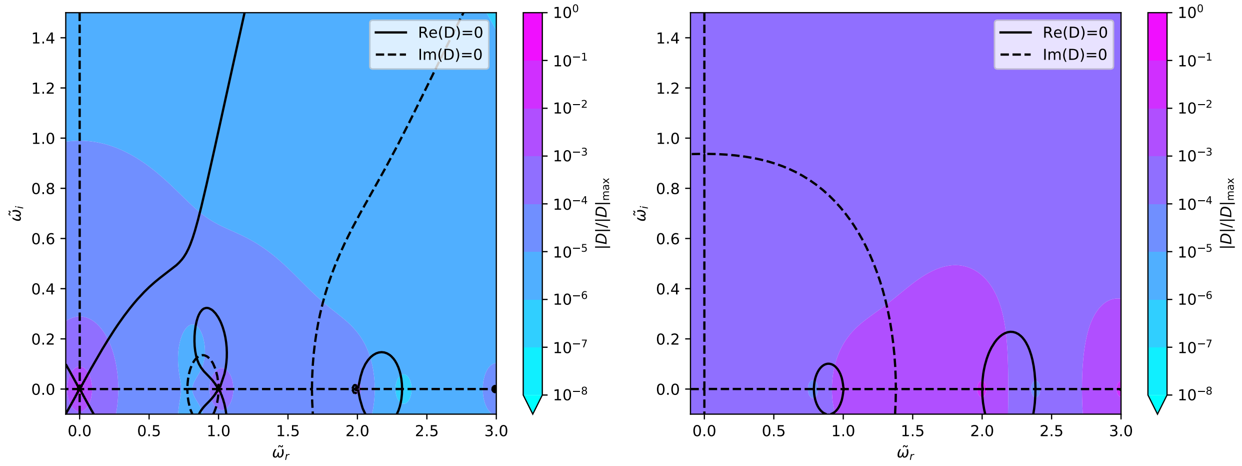
Figure 6.11: Filled contours for normalized magnitudes of the electromagnetic (EM) and electrostatic (ES) dispersion relations for case A plotted on a 320 × 320 grid with $\tilde{\omega}_r \in [-0.1, 3.0]$ and $\tilde{\omega}_i \in [-0.1, 1.5]$ with line contours of $\mathrm{Re}(D) = 0$ and $\mathrm{Im}(D) = 0$ overlaid. These contour lines cross at roots of $D(\omega, k_\perp) = 0$ as well as poles of $D(\omega, k_\perp)$. Solutions to the dispersion relation exist at the roots but not at the poles. The poles exist at real integer multiples of the cyclotron frequency for both the EM and ES dispersion relations, with an extra pole at 0 for the EM dispersion relation. For this case, the largest growing mode written as $\tilde{\omega}$ underneath each contour plot, given by the root of $D(\omega, k_\perp) = 0$ with the largest imaginary component, agrees well for the EM and ES dispersion relations and is purely imaginary.



(a) Case B EM, $\tilde{\omega} = 1.0363 + 0.2900i$

(b) Case B ES, $\tilde{\omega} = 1.0361 + 0.2899i$

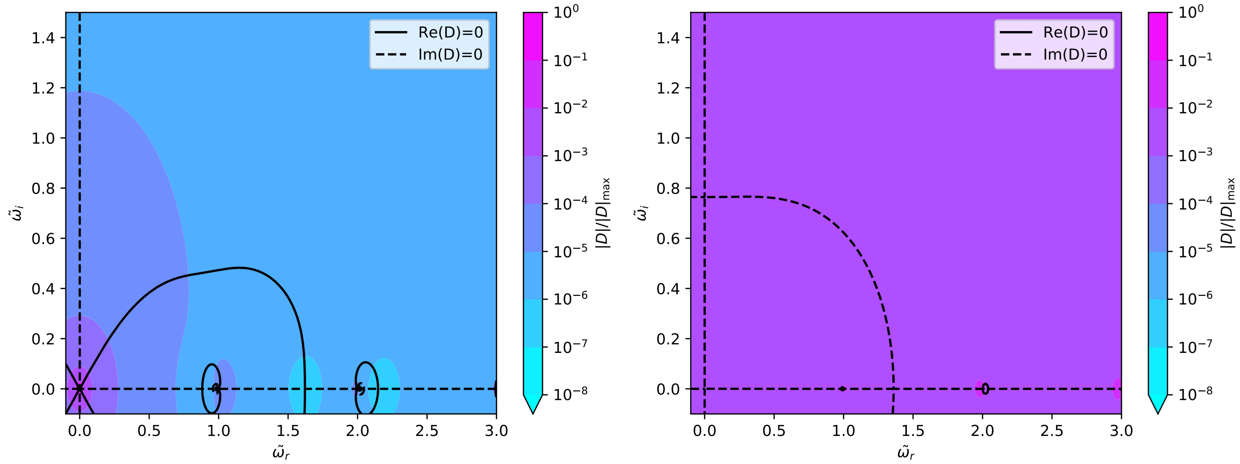
Figure 6.12: Filled contours for normalized magnitudes of the EM and ES dispersion relations for case B with line contours of $\text{Re}(D) = 0$ and $\text{Im}(D) = 0$ overlaid as in Fig. 6.11. For this case, the root of $D(\omega, k_\perp) = 0$ corresponding to the largest growing mode agrees well for the ES and EM dispersion relations and has mixed oscillatory and imaginary components.



(a) Case C EM, $\tilde{\omega} = 0.8426 + 0.1242i$

(b) Case C ES, multiple oscillatory solutions

Figure 6.13: Filled contours for normalized magnitudes of the EM and ES dispersion relations for case C with line contours of $\text{Re}(D) = 0$ and $\text{Im}(D) = 0$ overlaid as in Fig. 6.11. For this case, the root of $D(\omega, k_\perp) = 0$ for the EM dispersion relation corresponding to the largest growing mode has mixed oscillatory and imaginary components while the ES dispersion relation has no growing solutions.



(a) Case D EM, multiple oscillatory solutions

(b) Case D ES, multiple oscillatory solutions

Figure 6.14: Filled contours for normalized magnitudes of the EM and ES dispersion relations for case D with line contours of $\mathrm{Re}(D) = 0$ and $\mathrm{Im}(D) = 0$ overlaid as in Fig. 6.11. For this case, no growing roots of $D(\omega, k_\perp) = 0$ exist for either the EM or ES dispersion relations.

The next sections verify these observations using continuum kinetic simulations of the Vlasov-Maxwell system.

6.4.3 Numerical simulations of the DGH instability and comparison to theory

The four cases described in Sec. 6.4.2 are simulated using the discontinuous Galerkin algorithm described in Chapter 3. For each simulation, the electric field energy, given by

$$U_{\mathrm{E}} = \frac{1}{2} \int E^2 dx \tag{6.4.29}$$

is calculated and plotted on a semi-log scale from which the growth rate, $\tilde{\omega}_i$ and oscillation frequency, $\tilde{\omega}_r$ can be measured. The ring distribution given in Eq. (6.4.26) is initialized on a Cartesian grid of $x \in [0, \frac{2\pi}{k_\perp}]$ and $v_x, v_y \in [-4, 4]$ with a perturbation such that the initial electron distribution is given by

$$f(x, v_x, v_y)|_{t=0} = \frac{1}{\pi \alpha_\perp^2 j!} \left(\frac{v_x^2 + v_y^2}{\alpha_\perp^2} \right)^j \exp\left(-\frac{v_x^2 + v_y^2}{\alpha_\perp^2} \right) \left[1 + \epsilon \sin\left(4 \arctan\left(\frac{v_y}{v_x} \right) - k_\perp x \right) \right], \tag{6.4.30}$$

where $\epsilon = 10^{-4}$. Reference [125] uses the same perturbed distribution and demonstrated effective excitation of the dominant mode. Simulations are performed on a phase-space grid of resolutions given by $N_x \times N_{v_x} \times N_{v_y}$: $48 \times 24 \times 24$, $56 \times 28 \times 28$, $64 \times 32 \times 32$, and $80 \times 40 \times 40$. Second-order polynomial basis elements characterized by third-order optimal spatial convergence [55] are used with fourth-order ERK time-stepping. The Courant number used in all simulations is 0.235 in accordance with Ref. [33].

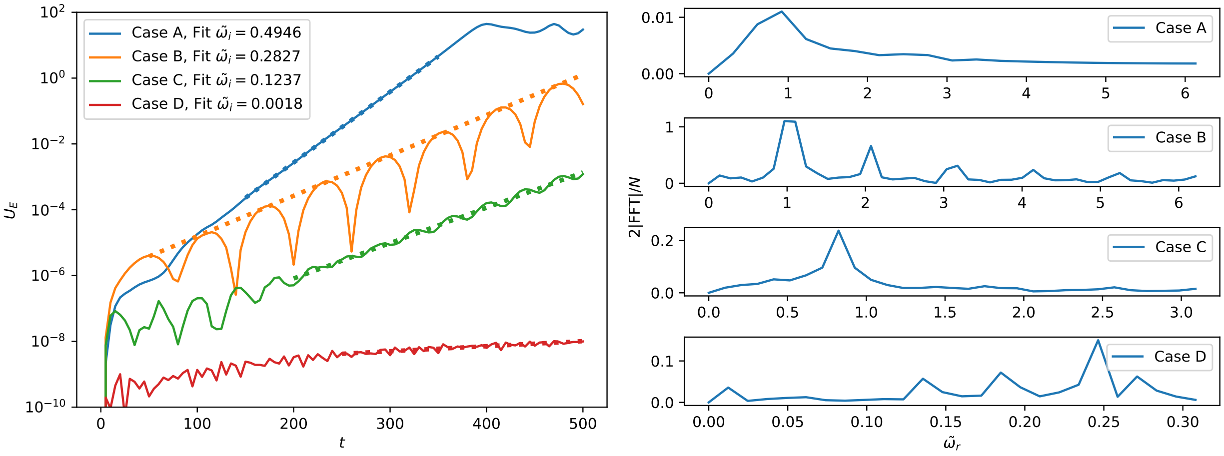
Figure 6.15a shows a plot of the electric field energy measured at every $t = 5$ for the $80 \times 40 \times 40$ resolution simulation for all cases, where the growth rates given by $\tilde{\omega}_i$ are measured using the slopes of line fits through the data. The growth rates for cases A-C are in line with the theoretical values predicted by the electromagnetic dispersion relation, while a small growth is seen for case D despite no prediction of growth from the theory. This small growth seen for case D may be a nonlinear effect due to the coupling of real frequency solutions to the dispersion relation excited from the perturbation profile in Eq. (6.4.30). Fast Fourier transforms (FFTs) are calculated for each case after the linear growth has been subtracted to determine the oscillation frequencies given by $\tilde{\omega}_r$, which are shown in Fig. 6.15b. Mixed growth and oscillation is expected for cases B and C according to the electromagnetic dispersion relation as shown in Sec. 6.4.2. The oscillation frequencies are found to be at the peaks of the FFT spectra. For case B, the oscillation frequency is calculated as the average of the first peak, found to be at $\tilde{\omega}_r^B = 1.0357 \pm 0.0690$ while for case C, the peak is found at $\tilde{\omega}_r^C = 0.8240 \pm 0.0515$, where in both cases half the frequency bin size is specified as the uncertainty. These frequencies are found to be independent of resolution. The growth rates however can be compared for various resolutions to obtain converged values. Figure 6.16 shows the convergence from the simulations performed at the four resolutions described by plotting growth versus velocity-space resolution and line fitting

Case	$\tilde{\omega}_i^{\text{th,EM}}$	$\tilde{\omega}_i^{\text{th,ES}}$	$\tilde{\omega}_i^{\text{num}}$	$\varepsilon_i^{\text{EM}}$	$\tilde{\omega}_r^{\text{th,EM}}$	$\tilde{\omega}_r^{\text{th,ES}}$	$\tilde{\omega}_r^{\text{num}}$	$\varepsilon_r^{\text{EM}}$
A	0.4912	0.4912	0.4956	0.9%	-	-	-	-
B	0.2900	0.2899	0.2965	2.2%	1.0363	1.0361	1.0357 ± 0.0690	<6.7%
C	0.1242	-	0.1239	0.2%	0.8426	-	0.8240 ± 0.0515	<6.1%
D	-	-	0.0018	-	-	-	-	-

Table 6.2: Theoretical growth rates and oscillation frequencies of the electromagnetic and electrostatic dispersion relations as calculated in Sec. 6.4.2 as well as numerical results from discontinuous Galerkin simulations of the Vlasov-Maxwell system. Growth rates are converged values as shown in Fig. 6.16, and oscillation frequencies for cases B and C are calculated from FFTs of the 80 × 40 × 40 simulation. The errors between numerical results and electromagnetic theory are calculated as $\varepsilon_{i,r}^{\text{EM}} = \left|\tilde{\omega}_{i,r}^{\text{num}} - \tilde{\omega}_{i,r}^{\text{th,EM}}\right| / \tilde{\omega}_{i,r}^{\text{th,EM}}$.

to find the limit as element size approaches zero. The converged results are summarized in Table 6.2 along with theoretical values for the electromagnetic and electrostatic dispersion relations calculated in Sec. 6.4.2. Theoretical oscillation frequencies and those measured from the simulation results with the FFTs are also given in Table 6.2. The growth rate convergence is calculated assuming a rate of order 3 for second-order basis functions used in simulations. The growth rate discrepancies for cases A and C are less than 1% while the discrepancy for case B is 2.2%. The error in oscillation frequency as measured by the FFT should be less than 6.7% for case B and less than 6.1% for case C, based on the frequency resolution.

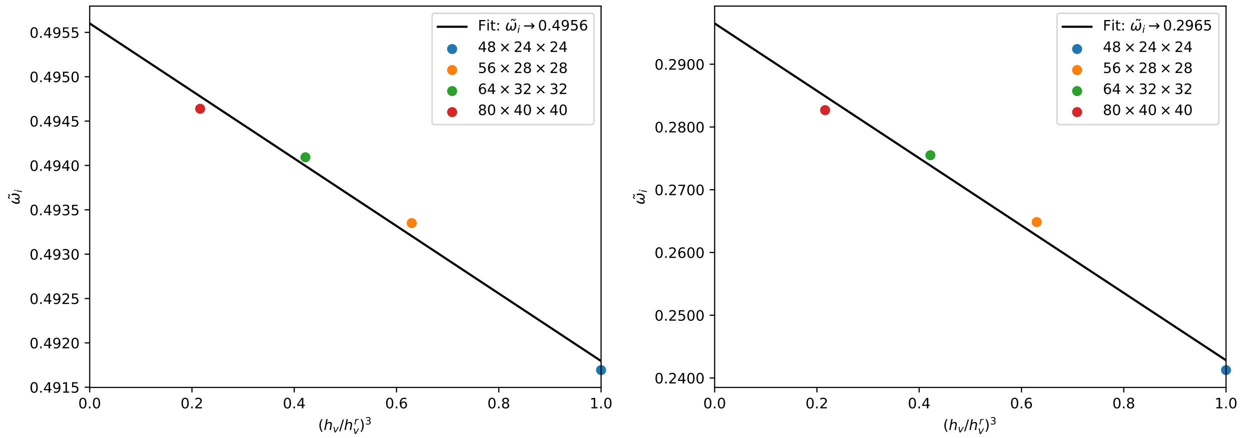
The simulation results show agreement with the theoretical predictions from the electromagnetic dispersion relations. They also confirm the agreement between the electrostatic approximation and the electromagnetic model for the cases of lower plasma beta in A and B. The agreement between electromagnetic theory and simulation in case C shows that with increased plasma beta, the electrostatic approximation is inadequate to accurately predict instabilities involving perpendicular waves. In case D, both theories predict stability and a



(a) Electric Field Energy

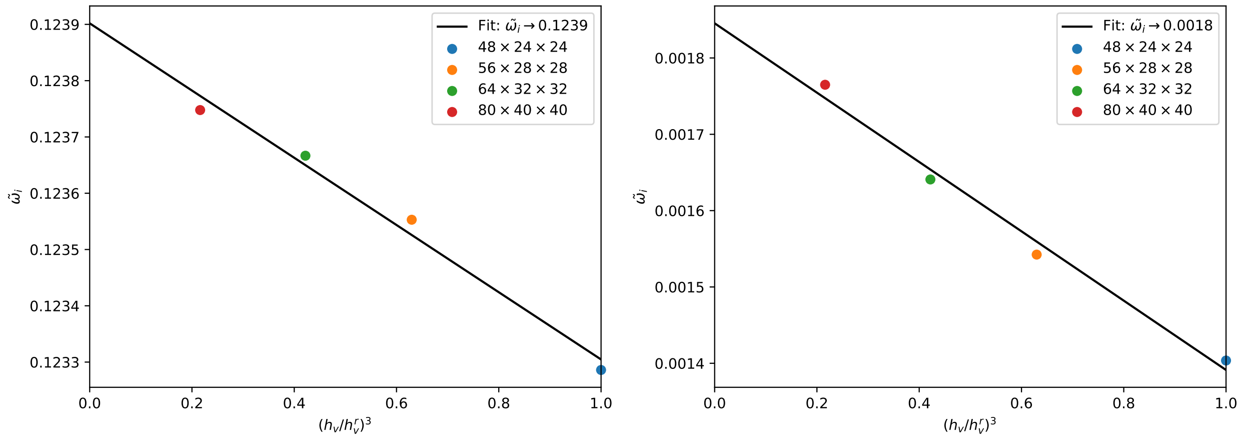
(b) FFT of Electric Field Energy

Figure 6.15: Simulations of the DGH instability with the Vlasov-Maxwell system using $80 \times 40 \times 40$ second-order polynomial elements. Figure 6.15a shows the growth of the electric field energy for cases A-D, where t is normalized by ω_p. Line fits showing the slope yielding $\tilde{\omega}_i$ are also shown for the times during which the linear growth is measured. The measurement durations are $t \in [150, 350]$, $[50, 500]$, $[200, 500]$, and $[250, 500]$ for cases A, B, C, and D, respectively. For case B, the line fit is based on the peaks of the oscillation. Figure 6.15b shows absolute values of the fast Fourier transforms for each case for these times, after the linear growth has been subtracted to remove the zero-frequency component and the result at negative frequencies are combined with those at positive frequencies. The peaks in the FFT plots correspond to the oscillation frequency, $\tilde{\omega}_r$. For case B, the oscillation frequency is calculated as the average of the first peak, yielding $\tilde{\omega}_r^B = 1.0357 \pm 0.0690$. For case C, the oscillation frequency is found at the first peak to be $\tilde{\omega}_r^C = 0.8240 \pm 0.0515$. (Simulations of the DGH instability are performed with WARPXM version 1.2.6 / hybrid_kinetics branch using input file dory_guest_harris_instability.py).



(a) Case A

(b) Case B



(c) Case C

(d) Case D

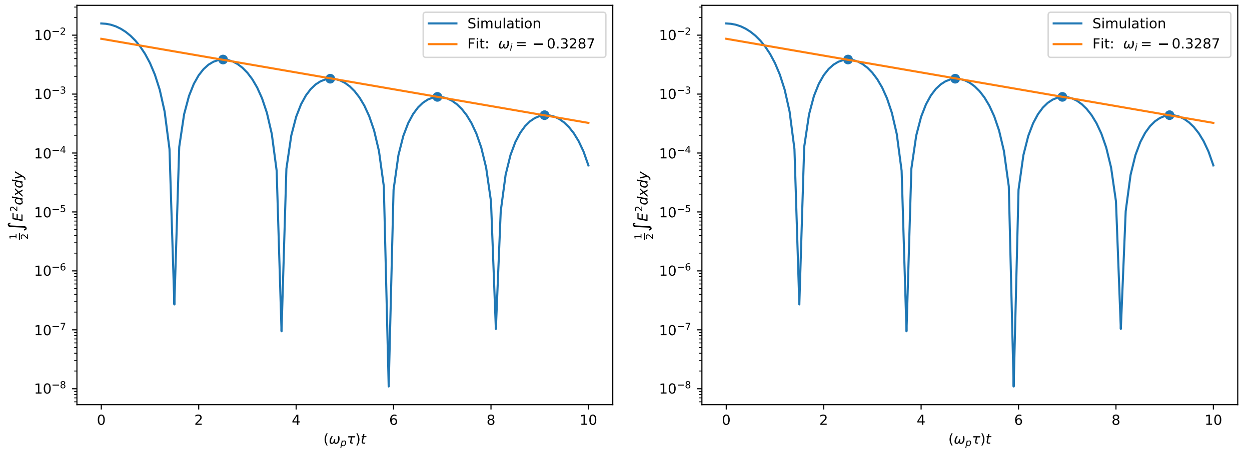
Figure 6.16: Convergence of growth rates $\tilde{\omega}_i$ for cases A-D. The growth rates are plotted for various resolutions against a velocity-space element length (h_v) normalized to the reference element length of a $48 \times 24 \times 24$ velocity-space element (h_v^r) with the expected cubic spatial convergence. Line fits are constructed to determine the y-intercepts corresponding to the $\lim_{h_v \rightarrow 0} \tilde{\omega}_i$.

small growth is seen in simulation, possibly due to nonlinear mode coupling.

6.4.4 Conclusions

A closed-form integral representation of the electromagnetic dispersion relation for the DGH instability is derived using electromagnetic theory, which is more complete than the electrostatic approximation that is only valid for low-beta plasmas. Theoretical growth rates and oscillation frequencies are computed for four cases of equilibrium electron ring distributions using electromagnetic and electrostatic dispersion relations. As predicted, agreement between the electromagnetic and electrostatic dispersion relations is found in the growth rates and oscillation frequencies for cases characterized by low plasma beta. A case with increased temperature (and corresponding increased plasma beta) and reduced plasma to cyclotron frequency ratio to further move the plasma below the density threshold for instability was predicted by the electrostatic dispersion relation to be stable but unstable according to electromagnetic theory. Simulations using WARPXM to solve the continuum kinetic Vlasov-Maxwell system in 1D2V confirm the electromagnetic dispersion results. A case with further increased temperature reducing the peakedness of the distribution and reduced plasma to cyclotron frequency ratio is found to be stable according to both theories, even though less confidence can be given to the electrostatic dispersion relation due to the corresponding increased plasma beta. The Vlasov-Maxwell simulations for this case show a small growth, possibly due to nonlinear effects.

The agreement between simulation results and theory confirm the use of the DGH instability as a viable benchmark for continuum kinetic codes solving the Vlasov-Maxwell system for plasmas subject to dynamic magnetic fields and currents. The electromagnetic generalization extends the analysis of the DGH instability using the electrostatic approximation confined to plasmas characterized by small beta such as those of cold ring distributions to include higher beta plasmas with higher temperatures. In addition, the closed-form integral representation of the electromagnetic dispersion relation derived in this work can be used as a benchmark for other numerical treatments of the Vlasov-Maxwell system, such as



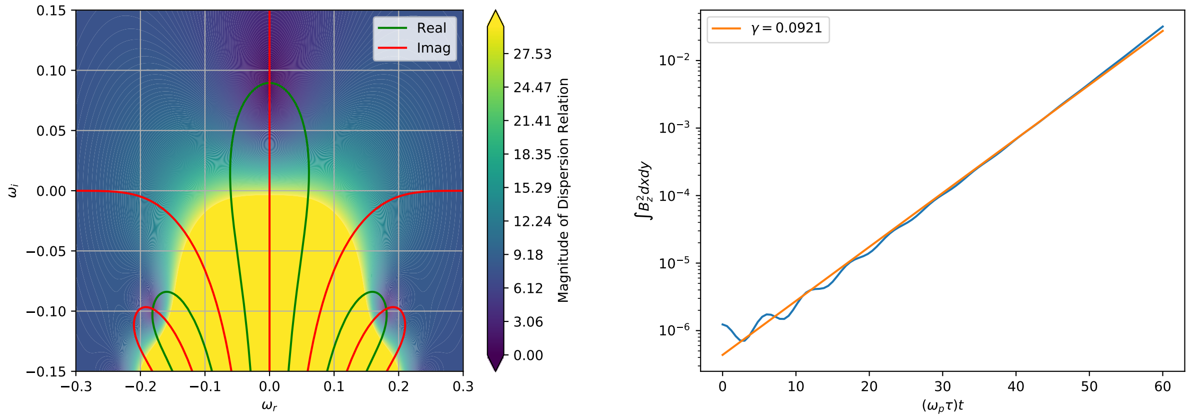
(a) k_x direction (b) k_y direction

Figure 6.17: Energy for 2D2V 32×32×32×32 Landau damping on second order 3-4 duoprism elements (first order polynomials) using a second order RK scheme. (Simulations of 2D2V Landau damping are performed with WARPXM version 1.1.2 / hybrid_kinetics branch using input file landau_damping_2d2v.py).

electromagnetic particle-in-cell methods.

6.5 2D2V Landau Damping

A 2D2V implementation of the kinetic model is implemented in WARPXM, using a 3-4 duoprism phase space element constructed by taking a tensor product of triangular elements in 2D physical space and line elements extending into v_x and v_y. Initial electrostatic validation is performed by running the weak Landau damping problem, initializing waves in either the x or y direction. Simulations are run using second order elements (first order polynomials) and a second order RK method. Details of this setup in 1D1V are discussed in Sec. 6.1. Figures 6.17a and 6.17b, show results for this problem in the 2D2V case for the x and y-directed wave numbers. In each case a growth rate of $\omega_i = -0.3287$ is found, compared to -0.3066 [23, 99].



(a) $T_e = 0.01$ and $k\lambda_D = 0.04$ yields $\gamma = 0.08951$

(b) Plot of$\int B_z^2 dxdy$

Figure 6.18: 2D2V Weibel instability for $T_e = 0.01$ and $k\lambda_D = 0.04$ yields a theoretical growth rate of $\gamma = 0.08951$. A 2D2V Vlasov-Maxwell $64 \times 64 \times 32 \times 32$ simulation starting from equilibrium with a perturbed B_z as described in Eq. (6.3.8) with $B_0 = 10^{-4}$ using second order elements (first order polynomials) and second order timestepping to $(\omega_p \tau)\, t = 60$ shows a growth rate of $\gamma = 0.0921$. (Simulations of the 2D2V Weibel instability are performed with WARPXM version 1.1.2 / hybrid_kinetics branch using input file weibel_instability_2d2v.py).

6.6 2D2V Weibel Instability

Here the Weibel instability condition is reexamined in 2D2V. The same setup as in the 1D2V case is followed, however, the y dimension is added to the simulation with $x, y \in [0, 2\pi/k_x]$. The case from Ref. [17] is simulated in which $T_e = 0.01$ and $k\lambda_D = 0.04$. Solving the dispersion relation as in Sec. 6.3 yields a growth rate of $\gamma = 0.08951$ as seen in Fig. 6.18a. A 2D2V simulation of this problem is then run using second order elements (first order polynomials) and second order RK timestepping on at $64 \times 64 \times 32 \times 32$ grid to $(\omega_p \tau)\, t = 60$. The simulation is started simply from the perturbed magnetic field initial condition (case 1 in Sec. 6.3). Figure 6.18b shows a growth rate near the theoretical number.

Chapter 7

HYBRID MODELING

The direct variable translation and composite distribution function methods described for the coupling of the $5N$-moment multi-fluid plasma model and the continuum kinetic model in Sec. 3.7 are tested by application to 1D1V problems. A neutral gas test and plasma sheath are used as test cases. A 2D2V magnetized Kelvin Helmholtz problem is also simulated.

7.1 1D1V Double Rarefaction Wave with Direct Variable Translation

Exploration of the direct variable translation method for coupling described in Sec. 2.4 between a neutral fluid with no heat flux or viscosity in 1D and the kinetic model in 1D1V is performed. A Riemann problem producing a double rarefaction wave [53, 14, 20] is initialized where

$$(\rho, v_x, p) = \begin{cases} (1.0, -0.2, 0.4) \text{ for } x < 0.5 \\ (1.0, +0.2, 0.4) \text{ for } x \geq 0.5 \end{cases} \tag{7.1.1}$$

on a domain for $x \in [0, 1]$ up to $t = 0.15$ with the ratio of specific heats, $\gamma = 3$. A problem is run in which the domain is split with the $5N$-moment model on the left and the kinetic model on the right, with $v \in [-10, 10]$. No limiter is used in this coupling problem, but may be necessary for larger jumps and the resulting stronger waves. On the kinetic side, these fluid variables are used to initialize Maxwellian distributions using Eq. (2.2.14). Dirichlet conditions are imposed on fluid variables while copy-out is imposed on the distribution function in phase space, acceptable up to this time since waves do not reach the boundary. Two cases are run, one where the subdomain boundary is at the jump ($x = 0.5$) and another where the subdomain boundary is at $x = 2/3$, allowing for the waves to propagate in the fluid region for some time before coming into contact with the kinetic region. In each case, a mesh of 128 × 320 elements (64 physical elements on either side in both cases) are used with first order polynomials on the physical and phase space with second order Runge-Kutta timestepping using Eqs. (3.4.2) - (3.4.3). The species is also charge neutral with $Z = 0$, eliminating electromagnetic source terms in Eqs. (2.3.19) and (2.3.20) as well as the velocity-direction flux in Eq. (2.3.8), which also removes the need to solve Maxwell's equations, Eqs. (2.3.10), and (2.3.11). Rusanov numerical fluxes are used for the fluid side of the domain.

Collisions drive the kinetic distribution function towards a Maxwellian according to Boltzmann's H-theorem [66], which is required for validity of the fluid model, so a collision operator of the form in Eq. (2.3.15) is included where the kinetic model is solved. The relaxation parameter for the collision operator is set to $\nu = 1000$ with normalization parameter $\nu_p \tau = 1$.

Results are given in Fig. 7.1. The solution is shown on the physical space domain with fluid variables (ρ,v_x,p) shown directly on the left. The same variables are shown on the right, but which are the result of the moments taken of the distribution function, again using Eqs. (2.2.8), (2.2.9), and (2.2.11). It can be seen that the direct variable translation method does exhibit the proper matching, at least when the distribution function remains close to a Maxwellian, which is the assumption that has to be made.

7.2 1D1V Double Rarefaction Direct Variable Translation and Composite Distribution Method Comparison

A double rarefaction wave problem involving a single thermalized neutral fluid as described in Sec. 7.1 is used to test the direct variable translation and composite distribution function methods in the limit where Maxwellian distribution functions exist on either side of a subdomain boundary. The problem couples the fluid model in 1D with the kinetic model in 1D1V to test each hybridization method.

The initial condition is characterized by the same Riemann problem described in Sec. 7.1 given by Eq. (7.1.1) for $x_c = 0.5$ in a physical space domain given by $x \in [0, 1]$. A smooth transition function is added to the velocity jump of the form

$$v_x = \frac{v_{x_L}}{2}\left[1 - \tanh\left\{\frac{2\pi\left(x - x_c\right)}{d}\right\}\right] + \frac{v_{x_R}}{2}\left[1 + \tanh\left\{\frac{2\pi\left(x - x_c\right)}{d}\right\}\right], \tag{7.2.1}$$

where $v_{x_L} = -0.2$ and $v_{x_R} = 0.2$ are the two states of v_x in Eq. (7.1.1) on either side of the jump and $d = 0.1$ is a measure of the transition width. The smooth transition ensures adequate resolution of the gradient, enabling accurate grid convergence studies.

Simulations are performed with the domain decomposed into left and right subdomains separated at $x = x_c$. For $x < x_c$, the fluid model is solved and for $x > x_c$, the kinetic model is solved. Both models use a single species with mass $A = 1$. The species is also charge neutral with $Z = 0$, eliminating electromagnetic source terms in Eqs. (2.3.19) and (2.3.20) as well as the velocity-direction flux in Eq. (2.3.8), which also removes the need to solve Maxwell's equations, Eqs. (2.3.10), and (2.3.11). Simulations are performed using the direct variable translation method for coupling of the fluid model with the kinetic model. The simulations are then repeated using the composite distribution function method. In the subdomain where the kinetic model is solved, the initial condition is set by using the density and pressure according to Eq. (7.1.1) and velocity according to Eq. (7.2.1) to define corresponding Maxwellian distribution functions using Eq. (2.3.17). The initial conditions for the distribution function and fluid variables are plotted in Fig. 7.2.

For consistency with the single spatial degree of freedom in a 1D1V kinetic simulation, the adiabatic index is set to $\gamma = 3$ in the fluid subdomain. The simulations are performed to $t = 0.15$ with the velocity-space domain of the kinetic model defined by $v_x \in [-10, 10]$. Second-order polynomial elements are used, comprising of 128 line elements in x, while the velocity-space resolution is varied to yield phase-space elements for resolutions of $N_x \times N_{v_x}$: 128×20, 128×40, 128×80, 128×160, 128×320, and 128×640. A third-order ERK method is used for timestepping and Rusanov numerical fluxes are used in the solution for the fluid model. The higher spatial and temporal orders used compared to simulations run in Sec. 7.1 allows for comparison of the hybridization method to higher precision. A simulation is also performed using the fluid model in both the left and right subdomains, denoted in the rest of this section as the fluid simulation. Simulations at all velocity-space resolutions are also performed using the kinetic model in both subdomains, denoted as the kinetic simulations. Collisions are implemented as in Sec. 7.1. The relaxation parameter is set using $\nu_p \tau = 1$ and varying ν until thermalization is achieved.

Figure 7.3 shows fluid variables, calculated using moments of the distribution function in the kinetic subdomains and as solutions to the fluid model in the fluid subdomains, for all simulations with ν values of 10, 100 and 1000 for the 128×640 resolution case at $t = 0.15$. Agreement between the kinetic model using the collision operator and the fluid model is found with increasing ν. The solutions using the direct variable translation and composite distribution function methods also generally agree well with the fluid and kinetic simulation solutions. However, the level of agreement reduces between the location of the initial jump in velocity at x_c and the rarefaction wave as it moves in time for low collision frequencies. This can be seen more clearly in a closeup of the number density as shown in Fig. 7.4. For lower collision frequencies, the discrepancy increases at x_c, corresponding to the initial velocity jump and the subdomain interface between the fluid and kinetic solutions, which causes a mismatch in the solution using either hybrid method, and is generally larger for the composite distribution function method. This mismatch is diminished for both coupling methods with increasing collision frequency, allowing the fluid moments calculated from the kinetic solution to approach a Maxwellian distribution function that matches the fluid solution.

Figure 7.3d visually confirms the matching distribution function at the subdomain interface for the $\nu = 1000$ case. As described in Sec. 3.7, the direct variable translation and composite distribution function methods are conservative and consistent when this subdomain interface distribution function is Maxwellian. Measurements of χ as defined in Eq. (3.7.17) confirm that the distribution function does approach a Maxwellian with increased collision frequency, as seen in Fig. 7.5, which plots χ for $\nu = 10$, 100, and 1000 for the 128×640 resolution simulations. Figure 7.5 shows that the subdomain interface distribution function approaches a Maxwellian (lower values of χ) for increasing collision frequency. Figure 7.5 also shows that χ measured in the hybrid simulations using the direct variable translation and composite distribution function methods closely matches χ measured in the kinetic simulation for the higher collision frequencies, further indicating similar evolution of the solution using the kinetic and fluid models when the distribution function approaches a Maxwellian.

A further check of the coupling performance of the direct variable translation and composite distribution function methods can be made by measuring mass, momentum, and energy conservation. As described in Sec. 3.7, the composite distribution function method should provide better conservation properties than the direct variable translation method in general, but both should provide conservation when the distribution functions on either side of the subomain interface approach the same Maxwellian. A norm is defined to measure the conservation properties, given by

$$L_2^{q_i} = \sqrt{\sum_{t=0}^{0.15} \left(\int_0^1 q_i(t)dx + \left[\left(\mathcal{F}_{ij\,(q_i(t))} n_j \right)_{x=0} + \left(\mathcal{F}_{ij\,(q_i(t))} n_j \right)_{x=1} \right] \Delta t - \int_0^1 q_i(t=0)dx \right)^2}, \tag{7.2.2}$$

where $q_i = [\rho, p_x, e]$ are the conserved variables in the one-dimensional fluid model in Eqs. (2.3.18), (2.3.19), and (2.3.20), where for x-momentum, $p_x \equiv \rho v_x$. Equation (7.2.2) is the L_2-norm of the difference in the integral of q_i over the physical-space domain at various times compared with $t = 0$, calculated at intervals of $\Delta t = 0.0015$. To account for outflow of mass, momentum, and energy at domain boundaries, the time-integrated flux of q_i is included in the $\mathcal{F}_{ij\,(q_i(t))} n_j$ terms, calculated using Eqs. (3.7.3), (3.7.4), and (3.7.5), where n_j is the unit normal pointing out of the simulation domain. Assuming the distribution functions are Maxwellian and constant at the domain boundaries (the double rarefaction wave does not reach the boundaries at $t = 0.15$), these fluxes can be calculated from the initial condition density, velocity, and pressure as

$$\left(\mathcal{F}_{(\rho)}n_x\right)_{x=0}+\left(\mathcal{F}_{(\rho)}n_x\right)_{x=1}=0.2+0.2=0.4 \tag{7.2.3}$$

$$\left(\mathcal{F}_{(p_x)}n_x\right)_{x=0}+\left(\mathcal{F}_{(p_x)}n_x\right)_{x=1}=-0.44+0.44=0 \tag{7.2.4}$$

$$\left(\mathcal{F}_{(e)}n_x\right)_{x=0}+\left(\mathcal{F}_{(e)}n_x\right)_{x=1}=0.124+0.124=0.248 \tag{7.2.5}$$

Equation (7.2.2) is calculated for the various velocity-space resolutions and plotted in Fig. 7.6 for $\nu = 1000$. Values of L_2 for mass, momentum, and energy converge to floors once velocity space is adequately resolved when $1/\Delta v_x = 4$, corresponding to the 128 × 80 resolution simulation. The converged L_2 values for the simulations using the composite distribution function method are similar to those for the kinetic simulations. This confirms the conservation property of the composite distribution function method. The conservation of mass, momentum, and energy using the direct variable translation method is reduced compared to the kinetic simulations and those using the composite distribution function method, indicated by the higher converged values reached in Fig. 7.6. Similar behavior is observed at lower collision frequencies, but with higher converged values of L_2 for mass, momentum, and energy for each hybrid method. This trend can be seen in Table 7.1, which shows the L_2 values for $\nu = 10$, 100, and 1000 for the simulations at the 128 × 640 resolution. This indicates that even the direct variable translation method does conserve mass, momentum, and energy to a higher degree as the subdomain interface distribution function approaches a Maxwellian, as postulated in Sec. 3.7.

Overall, the double rarefaction wave test indicates good performance using the direct variable translation and composite distribution function methods for coupling between the fluid and kinetic models for a single neutral fluid approaching a Maxwellian. The coupling is performed with minimal error or instability with good conservation properties in the Maxwellian limit.

	ν	F	K	DVT	CD
L_2^{ρ}	10	2×10^{-13}	1×10^{-8}	8×10^{-4}	7×10^{-9}
	100	2×10^{-13}	5×10^{-13}	5×10^{-5}	3×10^{-13}
	1000	2×10^{-13}	7×10^{-13}	1×10^{-5}	3×10^{-13}
$L_2^{p_x}$	10	3×10^{-16}	2×10^{-16}	1×10^{-3}	2×10^{-8}
	100	3×10^{-16}	4×10^{-15}	2×10^{-4}	7×10^{-13}
	1000	3×10^{-16}	1×10^{-13}	2×10^{-6}	5×10^{-14}
L_2^{e}	10	5×10^{-14}	8×10^{-8}	2×10^{-3}	4×10^{-8}
	100	5×10^{-14}	2×10^{-12}	4×10^{-4}	8×10^{-13}
	1000	5×10^{-14}	1×10^{-13}	9×10^{-6}	6×10^{-14}

Table 7.1: The norm $L_2^{\boldsymbol{q}}$ as defined in Eq. (7.2.2) for the double rarefaction wave problem plotted for the 128 × 640 resolution case using second-order polynomial elements for $\nu =$ 10, 100, and 1000. Mass, momentum, and energy are conserved to a higher degree for the composite distribution function method compared with the direct variable translation method, with both exhibiting better conservation as the distribution functions are more closely approximated as Maxwellians with increasing collision frequency.

7.3 1D1V Plasma Sheath with Direct Variable Translation

The direct variable translation method for coupling is tested on a non-neutral plasma using a simulation of a plasma sheath. The Vlasov-Maxwell system is used to solve this problem as described by Cagas [17] with a domain of $x \in [-128\lambda_D, 128\lambda_D]$ with 256 elements. The outer 32 elements from both left and right walls are solved using the Vlasov-Maxwell system for ion and electron distributions while the interior elements are solved with the $5N$-moment model for fluid ions and electrons. A realistic mass ratio is used, such that $A_\mathrm{i} = 1$, $A_\mathrm{e} = 1/1836$, with charges $Z_\mathrm{i} = +1$, $Z_\mathrm{e} = -1$. Phase space is spans $v_\mathrm{i} \in [-6v_{\mathrm{th}_\mathrm{i}}, +6v_{\mathrm{th}_\mathrm{i}}]$ and $v_\mathrm{i} \in [-6v_{\mathrm{th}_\mathrm{e}}, +6v_{\mathrm{th}_\mathrm{e}}]$ for ions and electrons, respectively, using 48 rectangular elements for each physical element. First order polynomials and second order timestepping are used.

The normalization is set by defining a reference length (L), reference number density (n_0), and reference temperature (T_0), from which L/δ_p and $\omega_p\tau$ are calculated. The choice is of a typical number density ($n_0 = 10^{19}$ m^{-3}), $T_0 = 1$ eV, and $L = L_D = \left(\frac{\epsilon_0 T_0}{n_0 q_0^2}\right)^{\frac{1}{2}} =$ 2.35 μm where q_0 is the elementary charge, e and $\epsilon_0 = 8.85 \times 10^{-12}$ F/m. From these parameters, the normalization described in Sec. 2.3 implies $p_0 = n_0 T_0 = \frac{B_0^2}{\mu_0}$ leading to $v_0 = \sqrt{\frac{T_0}{m_0}} = 9.79 \times 10^3$ m/s where reference $m_0 = m_p = 1.67 \times 10^{-27}$ kg (proton mass). The nondimensional parameters then become

$$\omega_p\tau = \sqrt{\frac{q_0^2 n_0}{\epsilon_0 m_0}\frac{\epsilon_0 T_0}{n_0 q_0^2}}\frac{1}{v_0} = \frac{\sqrt{\frac{T_0}{m_0}}}{v_0} = 1, \tag{7.3.1}$$

$$\frac{L}{\delta_p} = \frac{v_0\tau}{c/\omega_p} = (\omega_p\tau)\frac{v_0}{c} = 3.265 \times 10^{-5}. \tag{7.3.2}$$

The initial conditions for ions and electrons are for normalized $n_\mathrm{i} = n_\mathrm{e} = 1$, $v_{x_\mathrm{i}} = v_{x_\mathrm{e}} = 0$, $T_\mathrm{i} = T_\mathrm{e} = 10$. In the fluid subdomains, these variables are set directly, while in the kinetic subdomain they are converted to equivalent Maxwellian distribution functions, using $\gamma = 3$, appropriate for one spatial degree of freedom. The simulations are run to $t\omega_\mathrm{pe} = 200$ where $\omega_\mathrm{pe} = \sqrt{Z_\mathrm{e}^2 n_\mathrm{e}/A_\mathrm{e}}$ is the non-dimensional electron plasma frequency. The boundary condition of the sheath in physical space is that of a conducting wall on Maxwell's equations and are set as outflow conditions on the distribution function in phase space. Additionally, locally-dependent BGK operators are applied on both ion and electron species to replenish electron tails to achieve quasi-steady-state, as shown by Cagas [17], where the form of the BGK operator is given by Eq. (2.3.15) using a locally-dependant collision frequency given by Eq. (2.3.16). For this problem, $\nu_p \tau$ is chosen to be 2, with $\ln \Lambda = 10$.

Results comparing fluid moments at $t\omega_{pe} = 20$ are shown in Figs. 7.7a (ions) and 7.7b (electrons). For comparison, a full kinetic simulation is also performed (dashed black lines). The phase space for both ions and electrons, respectively, are shown in Figs. 7.8a and 7.8b. The solutions for the domain-hybridized plasma model and for the full kinetic model match closely until the electrons deviate significantly from a Maxwellian distribution at model boundaries. This eventually drives the ions away from a Maxwellian distribution as well, leading to to a loss of proper coupling among these species as well, as shown by corresponding Figs. 7.9a, 7.9b, 7.10a, and 7.10b at $t\omega_{pe} = 200$. However, as long as the $5N$-moment plasma model is physically valid at the model interface, i.e. the distribution functions remain Maxwellian, the domain-hybridized plasma model produces accurate results with reduced computational effort. To simulate this particular problem longer in time, one might choose to move the model boundary farther into the middle portion of the domain where the electron distribution function remains closer to a Maxwellian, or to solve the electrons with a full kinetic model while solving the ions with the hybrid approach, which would still reduce the computational effort. The boundary may also be moved dynamically during a simulation using the χ metric as defined in Eq. (3.7.17) determining the deviation of the distribution function from a Maxwellian.

7.4 1D1V Sheath with Direct Variable Translation and Composite Distribution Methods

The performance of the coupling methods described in Sec. 3.7 can be further assessed through application to a realistic problem involving multiple charged species in the presence of fields and where kinetic effects are essential to the dynamics. The 1D plasma sheath development provides such an assessment. As with the double rarefaction wave problem, the plasma sheath is studied in Sec. 7.3 using the direct variable translation method, and is expanded upon to include the composite distribution function method for comparison.

The plasma sheath development is simulated on a one-dimensional domain bounded by grounded electrodes (or walls), as described in Ref. [17]. The normalization used is also as described in Sec. 7.3, and is restated for convenience, with reference values of density $n_0 = 10^{19}$ m^3, temperature $T_0 = 1$ eV, and Debye length scale $L = \sqrt{\frac{\epsilon_0 T_0}{n_0 q_0^2}} = 2.35$ μm. A normalizing proton mass $m_0 = m_p$ is set, leading to $\omega_p \tau = 1$ and $L/\delta_p = v_0/c = 3.265 \times 10^{-5}$ using the normalization described in Sec. 2.3. A realistic mass ratio is used such that $A_\mathrm{i} = 1$ and $A_\mathrm{e} = 1/1836$ while the charges are $Z_\mathrm{i} = 1$ and $Z_\mathrm{e} = -1$. Ions and electrons are simulated with initial conditions of $n_\mathrm{i} = n_\mathrm{e} = 1$, $v_{x_\mathrm{i}} = v_{x_\mathrm{e}} = 0$, and $T_\mathrm{i} = T_\mathrm{e} = 10$ on a 1D1V grid where the physical space is defined by a grid of $x \in [-128, 128]$ where for hybrid simulations, for $x \in [-128, -96]$ and $x \in [96, 128]$, the kinetic model is solved and for $x \in [-96, 96]$, the fluid model is solved. The velocity space spans $v_\alpha = [-6v_{\mathrm{th}_\alpha}, 6v_{\mathrm{th}_\alpha}]$ for each species α where $v_{\mathrm{th}_\alpha} = \sqrt{T_\alpha / A_\alpha}$. A simulation using the kinetic model for both ions and electrons over all subdomains is also performed, denoted the kinetic simulation for the rest of this section. Boundary conditions are outflow for the distribution functions in phase space and conducting walls for the fields. Second-order polynomial spatial and velocity elements are used, with $\Delta x = 1$ and $\Delta v_{x_\alpha} = v_{\mathrm{th}_\alpha}/4$, along with third-order ERK timestepping. The simulations are run to $t\omega_{pe} = 20$, where $\omega_{pe} = \sqrt{Z_\mathrm{e}^2 n_\mathrm{e} / A_\mathrm{e}}$ is the non-dimensional electron plasma frequency. As with the 1D1V double rarefaction wave problem, the adiabatic index is set to $\gamma = 3$ in the fluid subdomain. For the fluid model, Roe fluxes as described in Sec. 3.1.4 are employed [103, 106].

Due to the mass ratio and equivalent initial ion and electron temperatures, the electron dynamics are much faster than for ions. As the highest-velocity electrons leave the domain and become absorbed by the walls, a positive potential forms and the electron distribution function loses symmetry and no longer remains Maxwellian. This happens immediately adjacent to the walls and emphasizes the utility of the domain-decomposed hybrid method, which allows for the kinetic solution in the subdomains adjacent to the walls and the fluid solution away from them. A BGK collision operator as written in Eq. (2.3.15) with locally-dependent collision frequency as given in Eq. (2.3.16) is thus added to the simulations in the kinetic subdomains for the electrons, with $\nu_p\tau = 1$ and $\ln\Lambda = 10$. This collision frequency is highest away from the walls where the fluid model becomes valid, and decreases as the plasma enters the sheath region [17] where the distribution function is expected to be far from Maxwellian. Collisions between ions and electrons are neglected.

Figure 7.11 shows the ion and electron distribution functions at $t\omega_{pe} = 20$ for the kinetic simulation as well as simulations employing the direct variable translation and composite distribution function hybrid methods. For the hybrid simulations, Maxwellian distribution functions constructed from the solutions to the fluid model in the middle subdomain are shown. Figures 7.11d and 7.11f indicate an accumulation of the electron distribution function at the subdomain interfaces for the simulations using the hybrid methods, which is more pronounced for the composite distribution function method. No such accumulation appears to occur in the ions, however.

Figures 7.12 and 7.13 show comparisons of the fluid variables n, v_x, p, and T for ions and electrons, respectively, for each of the simulations as well as χ defined in Eq. (3.7.17). Figure 7.12 shows agreement in the ion solution, which is expected due to the low value of χ at the subdomain interfaces, as illustrated in Fig. 7.12e, as well as the smooth transition at the subdomain interfaces in Figs. 7.11c and 7.11e. The jumps in electron density, pressure, and temperature seen in Figs. 7.13a, 7.13c, and 7.13d however confirm the observed accumulation of the distribution functions in Figs. 7.11d and 7.11f. The electron density and temperature discontinuities are more pronounced, however, for the composite distribution function method compared with the direct variable translation method. Figures 7.13b and 7.13c also show smoother electron velocity and pressure transitions between subdomains for the composite distribution function method compared with the direct variable translation method. Additionally, both hybrid methods increase the amplitude of oscillations in the fluid subdomain for the electrons, as shown in Figs. 7.13a, 7.13b, 7.13c, and 7.13d.

Figure 7.13e shows a value of χ_e of about 0.03 at the subdomain interfaces for the kinetic simulation at $t\omega_{\mathrm{pe}} = 20$. The discontinuities and oscillations present in the simulations using the direct variable translation and composite distribution function methods indicate that at this value of χ_e the fluid model is not valid for the electron species at the subdomain interfaces, and thus application of the fluid model at these locations produces inaccurate results. The further decrease in χ_e towards the center of the domain, as seen in Fig. 7.13e, indicates an advantage in moving the subdomain interface toward the center of the domain.

Simulations are thus performed using the direct variable translation and composite distribution function methods with the subdomain interface at $x_{si} = \pm 64$, which reduces the size of the fluid subdomain while increasing the size of the kinetic subdomains. Plots of fluid variables and χ_e for these simulations including the kinetic simulation are shown in Fig. 7.14. The density and temperature jumps when using the direct variable translation and composite distribution function methods are found to be reduced compared to when the subdomain interfaces are closer to the walls. The hybrid simulation results more closely follow the kinetic simulation results when the kinetic subdomains adjacent to the walls are expanded. The lower value of χ_e at the $x_{si} = \pm 64$ subdomain interface confirms that more accurate solutions are found as the distribution function at subdomain interfaces approaches a Maxwellian.

A measure of the electron mass and energy integrated over the domain of the hybrid simulations compared to the kinetic simulation for both cases of the subdomain interface location are shown in Fig. 7.15. This compares the conservation properties for each hybrid method against the kinetic simulation. The mass and energy are computed according to

$$\frac{\Delta M}{M_K} = \left| \frac{\int \rho(t) dx - \int \rho_K(t) dx}{\int \rho_K(t) dx} \right|, \tag{7.4.1}$$

$$\frac{\Delta E}{E_K} = \left| \frac{\int e(t) dx - \int e_K(t) dx}{\int e_K(t) dx} \right|, \tag{7.4.2}$$

where K refers to the kinetic simulation quantity. Integrated momentum is maintained at zero for all hybrid and kinetic simulations. As described in Sec. 3.7 and seen for the double rarefaction wave problem in Sec. 7.2, the composite distribution function method achieves mass and energy conservation that is closer to the kinetic simulation than by using the direct variable translation method. Moving the subdomain interface away from the wall where χ_{e} is reduced, however, improves mass and energy conservation and solution accuracy for both hybrid methods. The improvements further support the observations in Figs. 7.13 and 7.14, which show the solutions using both hybrid methods that are closer to the kinetic simulation when the kinetic subdomains are expanded.

7.5 Conclusions from 1D1V Tests

Simulations involving the double rarefaction wave in Secs. 7.1 and 7.2 and plasma sheath in Secs. 7.3 and 7.4 show that both the direct variable translation and composite distribution function methods for model coupling between the multi-fluid plasma model and the multi-species kinetic model are viable when the distribution function at the subdomain interface is close to Maxwellian. The plasma sheath simulations show that the direct variable translation method is more robust than the composite distribution function method, with smaller jumps in fluid variables at the subdomain interfaces. Conservation properties are significantly improved by placing the subdomain interfaces where χ is small, which validates the fluid model. The improved conservation properties are observed for both the composite distribution function method and the direct variable translation method. The next section will use the direct variable translation method to simulate the magnetized Kelvin-Helmholtz instability problem in 2D2V, demonstrating the effectiveness of the method for reducing computational costs while maintaining physical accuracy.

7.6 Domain-Decomposed Hybrid Simulations of the Magnetized Kelvin-Helmholtz Instability in 2D2V

The direct variable translation hybrid method is now applied in 2D2V to the magnetized Kelvin-Helmholtz instability. This instability is driven by velocity shear and governs the transport of collisionless low-beta plasmas perpendicular to a background magnetic field.

Previous work systematically studied the magnetized Kelvin-Helmholtz instability through derivation of linear growth rates using the Hall MHD model as well as simulations using higher-fidelity models including the multi-fluid plasma model as described in Sec. 2.2.2 and a multi-species kinetic model using a Vlasov-Poisson formulation for electrostatics [127]. The multi-fluid plasma model captures the physics of charge separation and diamagnetic drift (not captured by single-fluid MHD) in the shear layer of the magnetized Kelvin-Helmholtz instability while the kinetic model additionally resolves effects associated with finite Larmor radii such as non-Maxwellian distribution functions and pressure anisotropies. The aim of this research is to demonstrate the utility of the domain-decomposed hybrid method as described in Sec. 3.7 to capture the relevant physics by solving the kinetic model as described in Sec. 2.2.1 in the shear layer and the fluid model as described in Sec. 2.2.2 where the distribution functions are expected to remain close to Maxwellian. This allows for speedup in simulation times and reduction in computational costs by using the higher fidelity kinetic model only where it is needed.

This work is restricted to the study of the magnetized Kelvin-Helmholtz instability during linear growth. The low-beta property of the plasma studied means that the electrostatic assumption is valid and that results using the Vlasov-Maxwell and Vlasov-Poisson models should be indistinguishable [77]. The next sections describe the setup and results for simulations of the magnetized Kelvin-Helmholtz instability.

7.6.1 Vlasov-Maxwell equilibrium for the magnetized Kelvin-Helmholtz instability

The magnetized Kelvin-Helmholtz instability is simulated by initializing and perturbing an equilibrium that satisfies the Vlasov-Maxwell system as described in Sec. 2.2.1. The determination of the equilibrium for the Vlasov-Maxwell system closely follows the procedure outlined in Ref. [127], which determines an equilibrium for the Vlasov-Poisson system close to an equilibrium that satisfies the multi-fluid plasma model for electrons and ions for a specific density and electric field profile. The procedure outlined in Ref. [127] involves solving a nonlinear ordinary differential equation given by the Poisson equation that closely approximates the specified fluid density and electric field profiles, yielding solutions for the ion and electron distribution functions satisfying an equilibrium for the Vlasov-Poisson system. The determination of the kinetic equilibrium in this way is required for problems such as the magnetized Kelvin-Helmholtz instability, where Larmor radii and gradient scale lengths are comparable. In such cases, the fluid equilibrium becomes a poor approximation of the kinetic equilibrium, and thus initializing kinetic simulations using the fluid equilibrium can introduce spurious dynamics.

The following section summarizes the procedure to develop the kinetic equilibrium satisfying the Vlasov-Poisson system from the desired density and electric field profiles as described in Ref. [127], with a modification to satisfy the Vlasov-Maxwell system for low-beta plasmas.

Determining kinetic equilibria that satisfy the Vlasov-Maxwell system

The procedure for determining kinetic equilibria that satisfy the Vlasov-Poisson system begins by constructing auxiliary ion and electron distribution functions using constants of motion from specified ion density and electric field profiles from a two-fluid equilibrium. These auxiliary distribution functions are then used to solve a nonlinear Poisson equation to obtain exact equilibrium distribution functions, as described in detail in Refs. [127, 126].

The procedure can be understood through consideration of the Vlasov-Poisson system, consisting of Eq. (2.3.7) written for each species, and the Poisson equation, written in normalized form for electrostatic potential, ϕ,

$$-\frac{(\omega_c\tau)}{(\omega_p\tau)^2}\nabla^2\phi = \sum_\alpha Z_\alpha n_\alpha, \tag{7.6.1}$$

where n_α is the number density given by Eq. (2.2.8) for a species distribution function, f_α. The problem of interest involves a two-species low-beta collisionless plasma with a one-dimensional equilibrium such that $\phi = \phi(x)$ with a background magnetic field given by $\boldsymbol{B} = B_z\hat{\boldsymbol{z}}$. The Vlasov-Poisson equilibrium assumes that the background magnetic field is constant, $B_z \equiv B_{z0} = 1$. As described in Ref. [126], in such a plasma configuration, there exists two constants of motion, which are the energy and canonical momentum in the

y-direction, written in normalized forms as

$$\mathcal{E}_\alpha = \frac{A_\alpha \left(v_x^2 + v_y^2\right)}{2T_\alpha} + (\omega_c\tau) \frac{Z_\alpha \phi(x)}{T_\alpha}, \tag{7.6.2}$$

$$\mathcal{P}_{\alpha,y} = \frac{\mathcal{A}_y}{B_{z0}} + \frac{v_y}{\Omega_{c,\alpha}}, \tag{7.6.3}$$

where

$$\mathcal{A}_y = \int B_z dx \tag{7.6.4}$$

is the y-component of the magnetic vector potential, and $\Omega_{c,\alpha} = (\omega_c\tau) \frac{Z_\alpha B_{z0}}{A_\alpha}$ is a normalized cyclotron frequency. The species distribution functions, f_α, will satisfy Eq. (2.3.7) in equilibrium where $\partial f_\alpha / \partial t = 0$ if it can be expressed in terms of the constants of motion. Reference [126] assumes a form of f_α where the canonical momentum and energy dependencies are multiplicatively separable, given by

$$f_\alpha(\mathcal{P}_{\alpha,y}, \mathcal{E}_\alpha) = N_\alpha(\mathcal{P}_{\alpha,y}) \frac{A_\alpha}{2\pi T_\alpha} \exp\left(-\mathcal{E}_\alpha\right) \tag{7.6.5}$$

for some function $N(\mathcal{P}_{\alpha,y})$. If the distribution functions for each species can be formulated in this way and their moments of Eq. (2.2.8) for n_α additionally satisfy Eq. (7.6.1), a kinetic equilibrium for the Vlasov-Poisson system can be found. Investigation of the relationship between n_α and $\mathcal{P}_{\alpha,y}$ in Ref. [126] shows that $N(\mathcal{P}_{\alpha,y})$ can be expressed

$$N_\alpha(\mathcal{P}_{\alpha,y}) = \left[g_\alpha(X) \exp\left((\omega_c\tau) \frac{Z_\alpha \phi^*(X)}{T_\alpha} \right) \right]_{X = \mathcal{P}_{\alpha,y}}, \tag{7.6.6}$$

where $g_\alpha(x)$ and $\phi^*(x)$ are the desired density and electrostatic potential profiles that satisfy the specified fluid equilibrium. The Vlasov-Poisson equilibrium is then given by the solution for $\phi(x)$ in Eq. (7.6.2) that upon substitution along with Eq. (7.6.6) into Eq. (7.6.5), yields plasma densities according to Eq. (2.2.8) that solve Eq. (7.6.1).

Defining auxiliary distribution functions facilitates determination of the solution to Eq. (7.6.1) by allowing an alternative form of Eq. (7.6.5) expressed as

$$f_\alpha = f_\alpha^{\text{aux}} \exp\left((\omega_c\tau) \frac{Z_\alpha \left(\phi^* - \phi\right)}{T_\alpha} \right). \tag{7.6.7}$$

A similar relationship holds for the equilibrium number density, such that

$$n_\alpha = n_\alpha^{\text{aux}} \exp\left(\left(\omega_c\tau\right) \frac{Z_\alpha\left(\phi^* - \phi\right)}{T_\alpha} \right), \tag{7.6.8}$$

where n_α^{aux} is calculated from f_α^{aux} using Eq. (2.2.8). For a two-species ion-electron plasma with a specified equilibrium fluid ion density profile, $g_{\text{i}}(x)$, and electrostatic potential, $\phi^*(x)$, substitution of Eqs. (7.6.5) and (7.6.6) into Eq. (7.6.7) yields

$$f_{\text{i}}^{\text{aux}} = \left[g_{\text{i}}(X) \exp\left(\left(\omega_c\tau\right) \frac{Z_{\text{i}}\phi^*(X)}{T_{\text{i}}} \right) \right]_{X=\mathcal{P}_{\text{i},y}} \frac{A_{\text{i}}}{2\pi T_{\text{i}}} \exp\left(-\frac{A_{\text{i}}\left(v_x^2 + v_y^2\right)}{2T_{\text{i}}} - \left(\omega_c\tau\right) \frac{Z_{\text{i}}\phi^*(x)}{T_{\text{i}}} \right) \tag{7.6.9}$$

for ions, and

$$\begin{aligned} f_{\text{e}}^{\text{aux}} = & \left[-\frac{1}{Z_{\text{e}}} \left(Z_{\text{i}} n_{\text{i}}^{\text{aux, fit}}(X) + \frac{\left(\omega_c\tau\right)}{\left(\omega_p\tau\right)^2} \frac{\partial^2\phi^*(X)}{\partial X^2} \right) \exp\left(\left(\omega_c\tau\right) \frac{Z_{\text{e}}\phi^*(X)}{T_{\text{e}}} \right) \right]_{X=\mathcal{P}_{\text{e},y}} \\ & \times \frac{A_{\text{e}}}{2\pi T_{\text{e}}} \exp\left(-\frac{A_{\text{e}}\left(v_x^2 + v_y^2\right)}{2T_{\text{e}}} - \left(\omega_c\tau\right) \frac{Z_{\text{e}}\phi^*(x)}{T_{\text{e}}} \right) \end{aligned} \tag{7.6.10}$$

for electrons. In Eq. (7.6.10), g_{e} is written as the solution for n_{e} to Eq. (7.6.1) calculated from $\phi^*(X)$ and $n_{\text{i}}^{\text{aux}}(X)$, where $X = \mathcal{P}_{\text{e},y}$. A spline fit as described in Ref. [126] is constructed to evaluate $n_{\text{i}}^{\text{aux}}(X)$, which is evaluated numerically from $f_{\text{i}}^{\text{aux}}$ using Eq. (2.2.8). After a similar evaluation of $n_{\text{e}}^{\text{aux}}$, Eq. (7.6.1) becomes

$$-\frac{\left(\omega_c\tau\right)}{\left(\omega_p\tau\right)^2} \frac{\partial^2\phi}{\partial x^2} = Z_{\text{i}} n_{\text{i}}^{\text{aux}} \exp\left(\left(\omega_c\tau\right) \frac{Z_{\text{i}}\left(\phi^* - \phi\right)}{T_{\text{i}}} \right) + Z_{\text{e}} n_{\text{e}}^{\text{aux}} \exp\left(\left(\omega_c\tau\right) \frac{Z_{\text{e}}\left(\phi^* - \phi\right)}{T_{\text{e}}} \right). \tag{7.6.11}$$

Equation (7.6.11) can be solved numerically for ϕ by discretizing the Laplacian operator and applying Newton's method. Details of entire procedure for finding the solution for $\phi(x)$ are found in Ref. [126]. The computed distribution functions along with $\phi(x)$ define the Vlasov-Poisson equilibrium using the fluid profiles $g_{\text{i}}(x)$ and $\phi^*(x)$ for a constant B_{z0}.

For the Vlasov-Maxwell system to satisfy equilibrium, the steady-state Ampere's Law, Eq. (2.3.10), should additionally be satisfied. Integration of the steady-state form of Eq. (2.3.10) yields the required magnetic field profile,

$$B_z(x) = B_z(x_0) - \left(\omega_c\tau\right) \int_{x_0}^{x} Z_{\text{i}} n_{\text{i}}(x) v_{\text{i}y}(x) + Z_{\text{e}} n_{\text{e}}(x) v_{\text{e}y}(x) dx, \tag{7.6.12}$$

where Eq. (2.2.9) is used and $B_z(x_0) \equiv B_{z,x_0}$ is the magnetic field at a specified location. Evaluation of Eq. (7.6.12) yields a magnetic field consistent with the solution to ϕ solved in Eq. (7.6.11), which, for electrostatic low-beta plasmas, should be close to the constant B_{z0}. However, the $B_z(x)$ profile calculated in Eq. (7.6.12) can be integrated using Eq. (7.6.4) to calculate a corrected, self-consistent profile for $\mathcal{A}_y(x)$ in Eq. (7.6.3) and the process for finding the solution for $\phi(x)$ can be repeated using the corrected $\mathcal{A}_y(x)$. In this way, an equilibrium is found for the Vlasov-Maxwell system for a low-beta plasma for a self-consistent, nonuniform $B_z(x)$ profile that approximates a uniform B_{z,x_0}. The next section applies this procedure for a specific initialization of the magnetized Kelvin-Helmholtz instability.

Vlasov-Maxwell equilibrium for the magnetized Kelvin-Helmholtz instability

The magnetized Kelvin-Helmholtz instability is initialized for the case of a two-species plasma with ions and electrons. The ion number density and electrostatic potential profiles that solve the fluid equilibrium as described in Ref. [127] are given by

$$g_{\mathrm{i}}(x) = \left[1 + \exp\left(\frac{2x}{d}\right)\right]^{\frac{bd}{2}} \tag{7.6.13}$$

and

$$\phi^*(x) = -\frac{E_{x,0}d}{2}\ln\left[1 + \exp\left(\frac{2x}{d}\right)\right], \tag{7.6.14}$$

where b, d and $E_{x,0}$ are simulation parameters specifying an exponential growth factor, shear layer half-width, and electric field strength, respectively.

The procedure outlined in Sec. 7.6.1 to determine an equilibrium consistent with the Vlasov-Maxwell system is applied to case A1 as described in Ref. [127], where $b = -10.0$, $d = 0.05$ and $E_{x,0} = 2.00 \times 10^{-2}$. Additional parameters are $\omega_p\tau = \omega_c\tau = 1$, with $m_0 = m_p$ and $T_{\mathrm{i}} = T_{\mathrm{e}} = 6.25 \times 10^{-4}$. The equilibrium is one-dimensional with $x \in [-L_x/2, L_x/2]$ where $L_x = 1.0$. The magnetic field strength at $x_0 = 0$ is $B_{z,x_0} = 1.00$, which is the starting point for the initial condition calculation in Sec. 7.6.1. The ion to electron mass ratio is set to $A_{\mathrm{i}}/A_{\mathrm{e}} = 25$. A comparison of ϕ and $\mathcal{A}_y$ for the electrostatic case (calculating the equilibrium once based on $B_z = B_{z,x_0}$) and the electromagnetic case (calculating the equilibrium using Eq. (7.6.12) as an input for a second iteration) is shown in Fig. 7.16. The equilibrium is solved using 512 grid points and second-order differencing. Since the equilibrium is for a low-beta plasma, the electrostatic and electromagnetic profiles are similar. Plots of n, v_y, T, E_x, and B_z in Fig. 7.17 demonstrate the similarities between these profiles.

7.6.2 Domain-decomposed hybrid simulations of the magnetized Kelvin-Helmholtz instability

The one-dimensional equilibrium calculated in Sec. 7.6.1 is used to initialize a simulation of the magnetized Kelvin-Helmholtz instability in two dimensions. The simulation parameters and discussion of results are described in the following subsections.

Initial conditions for the magnetized Kelvin-Helmholtz instability

Simulations of the magnetized Kelvin-Helmholtz instability evolved from the perturbed equilibrium derived in Sec. 7.6.1 are performed on a two-dimensional physical-space domain of $x \in [-L_x/2, L_x/2]$ and $y \in [-L_y/2, L_y/2]$ where $L_y = 2\pi/k_y$ with $k_y = 8$. The domain is subdivided into a middle subdomain with $x \in [-L_x/4, L_x/4]$, an outside subdomain to the left with $x \in [-L_x/2, -L_x/4]$, and an outside subdomain to the right with $x \in [L_x/4, L_x/2]$. The physical-space grid for the entire domain is composed of 32×16 second-order polynomial basis function triangular elements, which are created from a subdivision of each element in a 32×8 rectangular grid. Dirichlet boundary conditions on the left and right walls are used, setting all variables to the initial condition, and periodic boundary conditions are used at the top and bottom of the domain. A simulation is performed using the Vlasov-Maxwell kinetic model described in Sec. 2.2.1 over the entire domain for both ions and electrons, which in the remainder of this section is referred to as the kinetic simulation. A simulation is also performed in which the ions are solved using the Vlasov-Maxwell kinetic model in the middle subdomain and the fluid model described in Sec. 2.2.2 in the left and right subdomains while the electrons are solved using the Vlasov-Maxwell kinetic model in all subdomains. This is referred to as the hybrid simulation in the remainder of this section. In the subdomains in which the kinetic model is used, the velocity space is comprised of two velocity dimensions, such that $v_x, v_y \in [-v_{\text{max},\alpha}, v_{\text{max},\alpha}]$ where $v_{\text{max,e}} = 1.0$ and $v_{\text{max,i}} = 0.2$. The velocity space is discretized using 32×32 second-order polynomial basis function rectangular elements. The phase-space elements are constructed from tensor products of the underlying triangular elements in physical space and rectangular elements in velocity space, yielding 3-4 duoprism elements [35]. In the left and right subdomains where the fluid model is solved for the ions, initial values for fluid variables are calculated from velocity moments in Eqs. (2.2.8), (2.2.9), and (2.3.13) of the initialized equilibrium distribution functions. Values of χ_i in the left and right subdomains at the initial condition are found to be below 10^{-4}, validating the fluid model. Rusanov fluxes are used in the fluid subdomains and for consistency with 2 degrees of freedom in a 2D2V simulation, the adiabatic index is set to $\gamma = 2$. To seed the instability, a perturbation is applied to the electron distribution function as described in Ref. [127], which multiplies f_e by a factor $(1 + \epsilon)$, where

$$\epsilon = 2.0 \times 10^{-4} \sin(k_y y) \exp\left(-\frac{x^6}{d^6}\right). \tag{7.6.15}$$

Third-order ERK timestepping is used to advance solutions until a time of $t\omega_s = 60$, for which linear behavior is expected. The shearing rate, ω_s, is related to the ion velocity jump in the shear layer, $\Delta v_{\text{i}y}(t = 0)$ in Fig. 7.17b, through the relationship $\omega_s = |\Delta v_{\text{i}y}(t = 0)| / (2d)$. Reference [127] approximates $\Delta v_{\text{i}y}(t = 0) = -0.02625$ for case A1, leading to $\omega_s = 0.2625$.

A hybrid simulation using the fluid model for both the electrons and ions in the left and right subdomains coupled to the kinetic model in the middle subdomain is also performed. However, this simulation loses accuracy due to the electron distribution function developing large deviations from a Maxwellian over the entire domain as evidenced by large values of χ_e. Therefore, results from this simulation are not included in the investigation.

Simulation results of the magnetized Kelvin-Helmholtz instability

A plot of the evolution of the ion transverse velocity in the x-direction at $(x, y) = (0, 0)$ is shown in Fig. 7.18 for the kinetic and hybrid simulations. The linear growth rate from the hybrid simulation is calculated to be $\omega_i/\omega_s = 0.1485$, compared to $\omega_i/\omega_s = 0.1527$ for the kinetic simulation, based on line fits of the peaks between $t\omega_s = 20$ and $t\omega_s = 50$ before effects of nonlinear saturation are observed. These growth rates are within 5.3% of the growth rate of $\omega_i/\omega_s = 0.145$ found for the Vlasov-Poisson simulation in Ref. [127]. The growth rates shown in Fig. 7.18 approximate the kinetic simulation results markedly better than two-fluid simulation results, which has $\omega_i/\omega_s = 0.174$ [127].

Figure 7.19 shows the ion number density for each simulation measured at $t\omega_s = 60$. There are some discontinuities at the model interfaces at $x/d = \pm 5$ in the hybrid simulation. Figure 7.20 shows χ_i for the kinetic and hybrid simulations at $t\omega_s = 60$. The kinetic simulation shows χ_i to be approximately between $0.01 - 0.02$ at the $x/d = -5$ interface and approximately between $0.02 - 0.03$ at the $x/d = 5$ interface, indicating deviation from Maxwellian at these levels are sufficient to cause the ion number density discontinuities. Values of χ_i in the hybrid simulation show that the differences in the kinetic and Maxwellian distribution functions on the $x/d = 5$ interface where density is low further increases the Maxwellian deviation. Thus, it may be advantageous to extend the kinetic subdomain beyond $x/d = 5$ where χ_i is lower to reduce the discontinuities, as is performed for the plasma sheath in Sec. 7.4.

Figure 7.21 measures mass, y-momentum, and energy for ions integrated over the physical domain for the hybrid simulation relative to the kinetic simulation. Similar to Eqs. (7.4.1) and (7.4.2) in Sec. 7.4, these integrations in two dimensions are

$$\frac{\Delta M}{M_K} = \left| \frac{\int \rho(t) dA - \int \rho_K(t) dA}{\int \rho_K(t) dA} \right|, \tag{7.6.16}$$

$$\frac{\Delta P_y}{P_{y,K}} = \left| \frac{\int p_y(t) dA - \int p_{y,K}(t) dA}{\int p_{y,K}(t) dA} \right|, \tag{7.6.17}$$

$$\frac{\Delta E}{E_K} = \left| \frac{\int e(t) dA - \int e_K(t) dA}{\int e_K(t) dA} \right|. \tag{7.6.18}$$

As with the 1D1V conservation calculations in Sec. 7.4, Eqs. (7.6.16), (7.6.17), and (7.6.18) help to assess the performance of the hybrid simulation using the domain-decomposed hybrid method. Integrated x-momentum is maintained at zero for both the kinetic and hybrid simulations. Figure 7.21 shows that conservation in the hybrid simulation reduces as the simulation progresses and the instability grows. This is due to increasing deviations from Maxwellian in the subdomain interface distribution functions, observed through increasing values of χ_{i} over time. This behavior suggests the need for extending the kinetic subdomain if simulations are run to a longer time. The subdomains could also be changed dynamically during the simulation, where the kinetic subdomain is expanded based on measured values of χ_{i}.

Further simulations using higher order elements and higher resolution may be desirable to further resolve the instability and calculate convergence of the growth rate. However, the simulations presented in this section show the viability of the domain-decomposed hybrid method, particularly using direct variable translation, as a means to reduce the computational cost by using the fluid model in regions where the distribution function remains close to a Maxwellian.

The hybrid simulation, which applies the fluid model for ions on half of the domain, reduces the simulation time by 28% with a memory saving of 25%. In both simulations, the domain is divided into approximately equal areas in physical space on which different compute nodes communicating using MPI advance the solution in parallel. Additional speedup can thus be realized for the hybrid simulation with the use of load balancing measures, such as by applying more compute nodes to the kinetic subdomain, where the more computationally expensive kinetic model is used to advance the solution for ions.

7.7 Conclusions

This research presents the domain-decomposed hybrid method for simulations coupling the multi-fluid plasma model and the multi-species kinetic model governing plasma dynamics in the presence electromagnetic fields, enabling faster simulations with reduced computational resources while maintaining high physical fidelity. The method subdivides a simulation domain into separate subdomains in which the different models are solved. The solution for each model is calculated using the same continuum method, the discontinuous Galerkin finite element method, which facilitates the specification of the interface conditions between the different models on adjacent subdomains. The specification of these interface conditions is defined through surface numerical fluxes, which are a component of the discontinuous Galerkin method. The domain-decomposed hybrid method can be applied separately for different species such as ions and electrons, which is advantageous for many applications where the fluid approximation may be sufficient to govern the dynamics of one species but not another.

Two particular methods are studied for specification of the numerical flux at subdomain interfaces. The first method, called the direct variable translation method, constructs fluid variables from the kinetic distribution function at a subdomain interface. The construction of the fluid variables from the kinetic distribution function allows for determination of the numerical flux for the multi-fluid plasma model in a manner that is consistent with the fluid approximation of Maxwellian distribution functions. The second method, called the composite distribution function method, calculates numerical fluxes based on the construction of a composite distribution function at the subdomain interface, ensuring conservation of the distribution function as well as mass, momentum, and energy.

The direct variable translation and composite distribution function methods are tested using simulations of a double rarefaction wave and a plasma sheath using the multi-fluid plasma model in 1D and the multi-species kinetic model in 1D1V. Results demonstrate the conservation properties of each method and show good coupling results for distribution functions on either side of a subdomain interface approaching a common Maxwellian. The effectiveness of the coupling reduces when the distribution function at the subdomain interface deviates from a Maxwellian, which can be measured using the metric, χ. In such cases, accumulation along with increased oscillatory behavior of the distribution function emanating from the subdomain interfaces can occur. Placing the subdomain interfaces at locations of lower χ improves the solution accuracy and resolves the numerical artifacts. The direct variable translation method, despite having weaker conservation properties, is found to be more robust than the composite distribution function method due to reduced numerical artifacts at subdomain interfaces for a given value of χ.

The direct variable translation method is then used to couple ions simulated using the multi-fluid plasma model in 2D and the multi-species kinetic model in 2D2V for the magnetized Kelvin-Helmholtz instability. An equilibrium is initialized that satisfies the steady-state electromagnetic Vlasov-Maxwell system, modified from a method deriving the Vlasov-Poisson equilibrium for electrostatics. The hybrid simulation yields a linear growth rate of the instability that agrees well with a simulation using the kinetic model with no hybridization, but does so in less time and with memory savings, consistent with the reduced computational complexity of the multi-fluid plasma model compared with the multi-species kinetic model. The ability to capture the relevant physical phenomena with reduced computational resources demonstrates the viability of the domain-decomposed hybrid method for the simulation of plasma dynamics to high physical fidelity.

Further advancement in the method presented in this work for hybridization of the fluid and kinetic models are suggested. The domain decomposition can be made dynamic, such that subdomain interface locations evolve throughout a simulation based on local plasma parameters. The χ metric could be used to determine dynamic decomposition, so that the subdomain interfaces move to where χ is low, applying the fluid model only in regions where it is valid. Furthermore, the interface locations can be replaced with transition regions where the solutions to the fluid and kinetic models transition into each other over some finite distance. This could potentially reduce the numerical artifacts that occur at some subdomain interfaces. Additional load balancing measures such as dedication of more computational resources to subdomains where the more computationally intensive kinetic model is solved can increase computational efficiency and further reduce simulation time. These improvements will help reduce the computational cost of simulating plasma dynamics to high fidelity, allowing for numerical simulation of phenomena beyond the current capabilities.

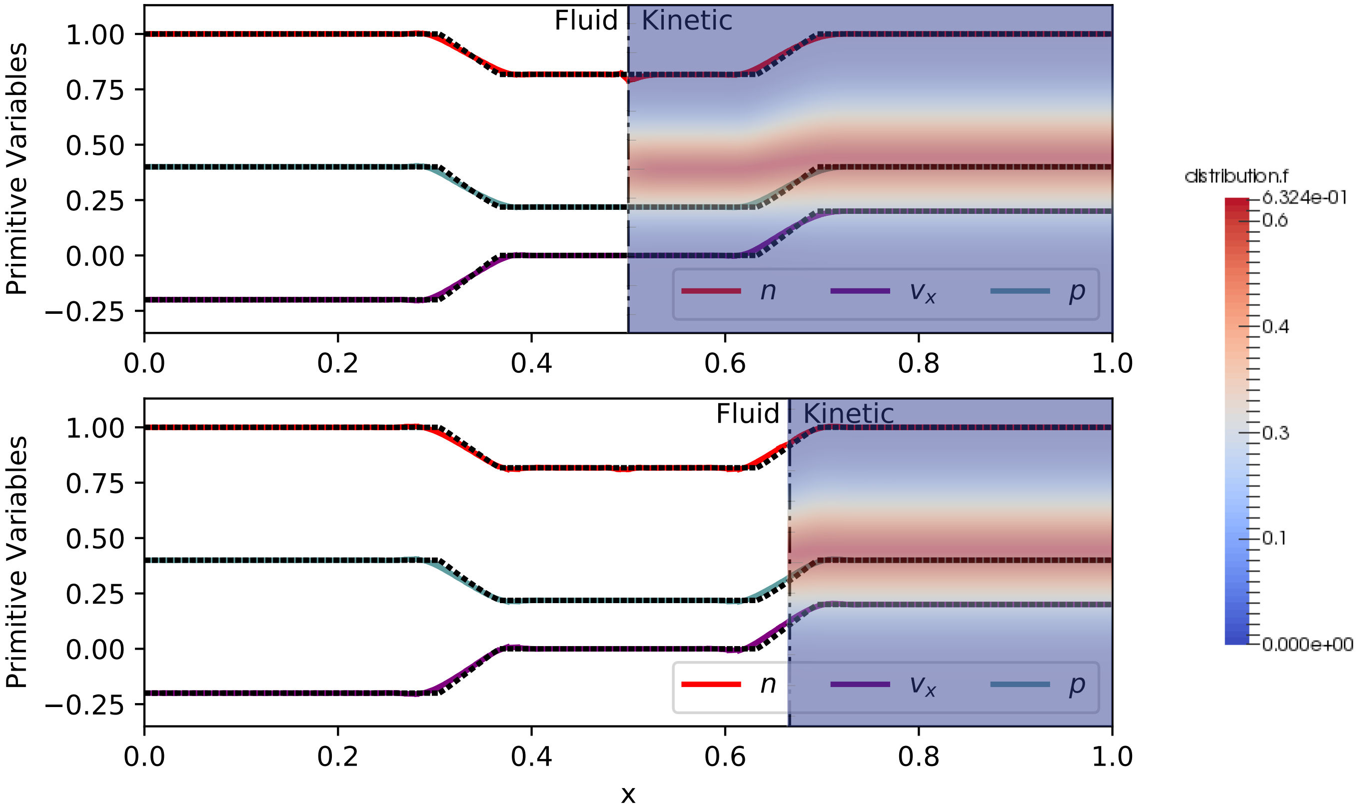
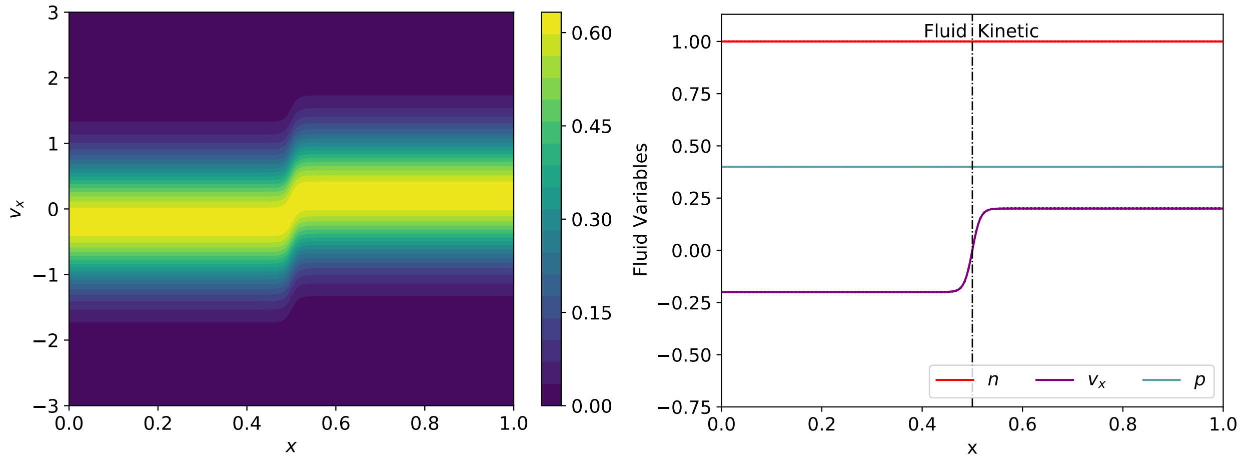


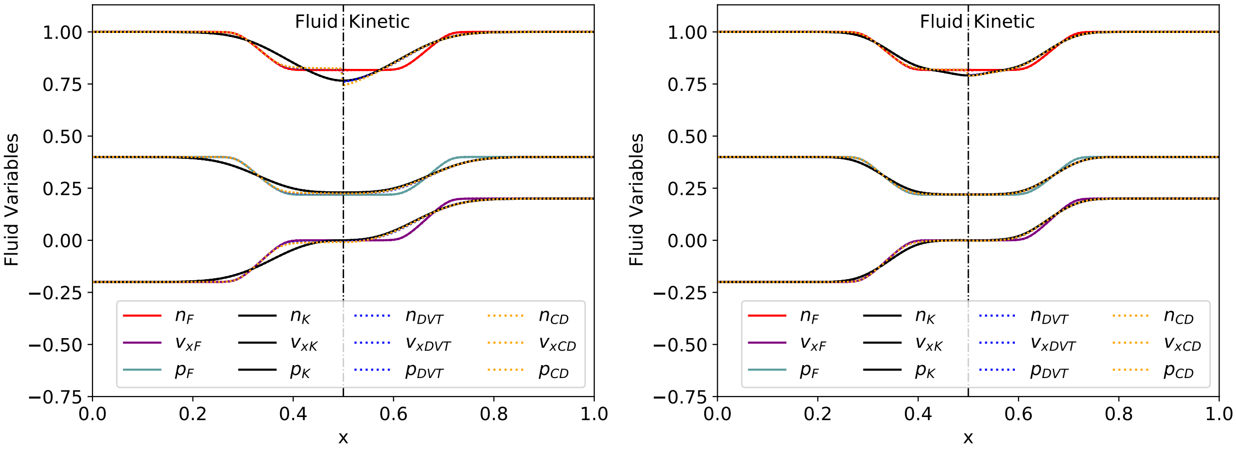
Figure 7.1: Double rarefaction waves problem with $5N$-moment model on left of model interface and continuum kinetic model with BGK operator on right of model interface at $t = 0.15$ on $x \in [0, 1]$ subject to direct variable translation matching. The top case has the model interface at $x = 0.5$ while the bottom case has the model interface at $x = 0.667$. Black dashed lines are the analytic solution to the fluid Riemann problem [70]. The distribution function is also overlaid in the kinetic regimes. (The double rarefaction waves problem is simulated with WARPXM version 1.2.6 / hybrid_kinetics branch using input file double_rarefaction.py for fluid and kinetic simulations as well as those using the direct variable translation method while the input file double_rarefaction_flux_bc_testing.py is used for simulations involving the composite distribution function method).



(a) Distribution function for $x \in [0, 1]$ and $v_x \in [-3, 3]$. The velocity space for the simulations however spans $v_x \in [-10, 10]$

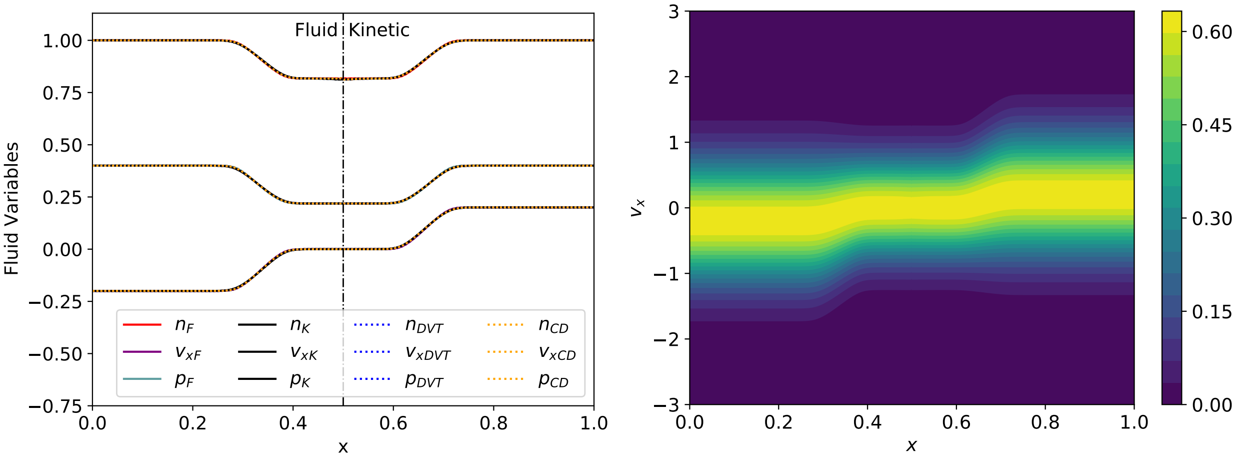
(b) Fluid variables for $x \in [0, 1]$.

Figure 7.2: Initial condition for the double rarefaction wave simulation on a grid of 128×640 second-order polynomial elements. Figure 7.2a shows the initial condition for the distribution function and Fig. 7.2b shows corresponding fluid variables n, v_x, and p.



(a) Fluid variables for $\nu = 10$

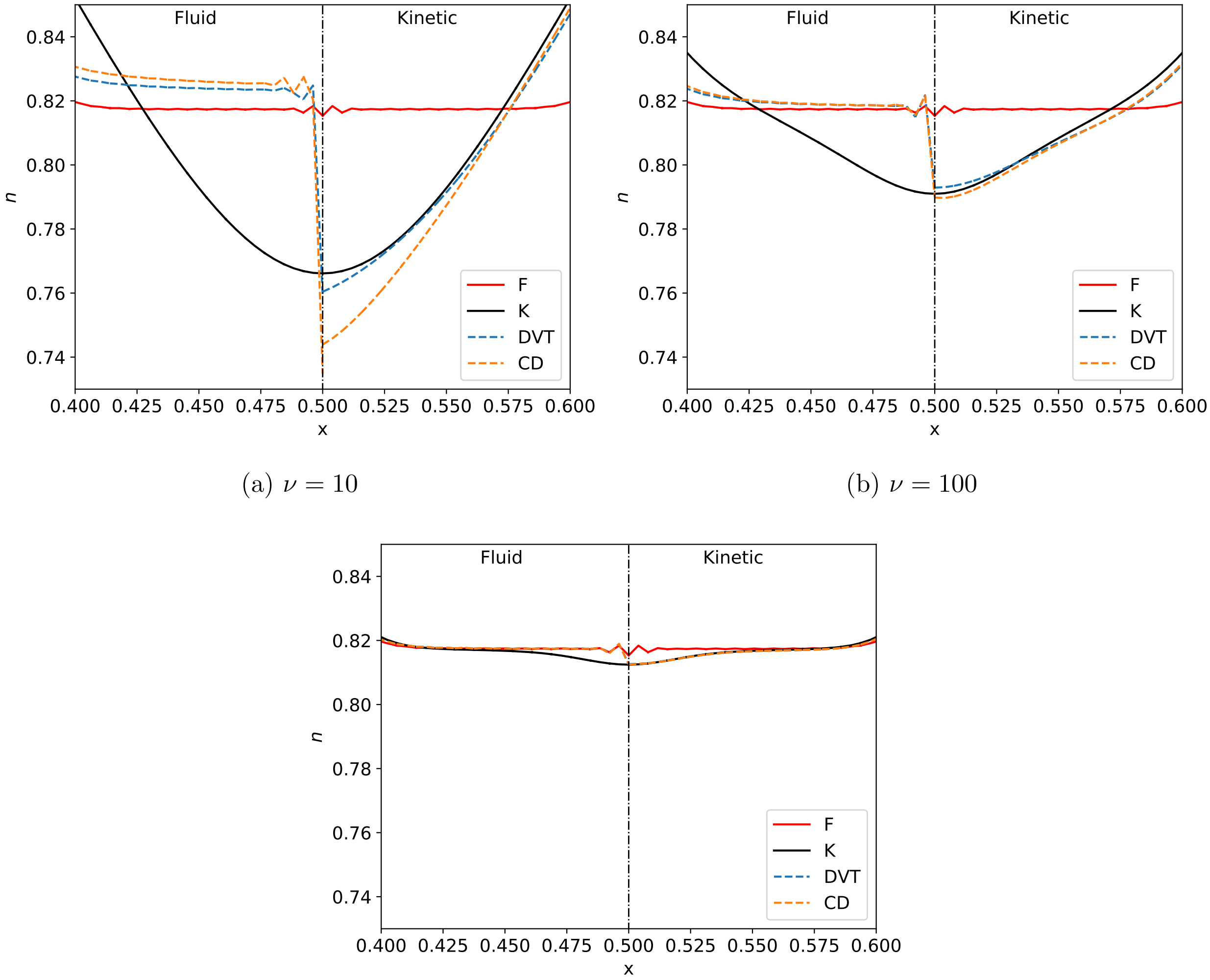
(b) Fluid variables for $\nu = 100$



(c) Fluid variables for $\nu = 1000$

(d) Distribution function for $\nu = 1000$ for the kinetic simulation.

Figure 7.3: Comparison of domain-decomposed hybrid simulations of the double rarefaction wave problem a on grid of 128×640 second-order polynomial elements at $t = 0.15$. Fluid variables, measured by taking moments of the distribution function in the kinetic subdomain, and as solutions to the fluid model in the fluid subdomain, are shown in Figs. 7.3a, 7.3b, and 7.3c for $\nu = 10$, 100, and 1000. F, K, DVT, and CD denote fluid, kinetic, hybrid using the direct variable translation method, and hybrid using the composite distribution function method simulations, respectively. The distribution function for the $\nu = 1000$ case is shown in Fig. 7.3d for the kinetic simulation. The BGK operator with increasing ν relaxes distribution functions towards Maxwellians on subdomains where the kinetic model is solved, allowing for coupling of the fluid and kinetic models using the direct variable translation and composite distribution function methods at the subdomain interface at $x = x_c = 0.5$ with minimal error.

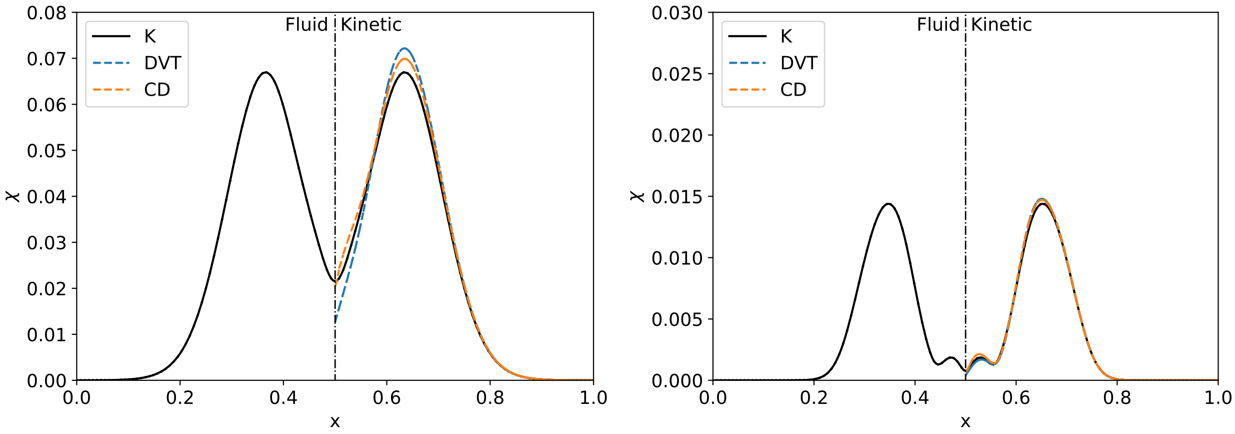


(a) $\nu = 10$

(b) $\nu = 100$

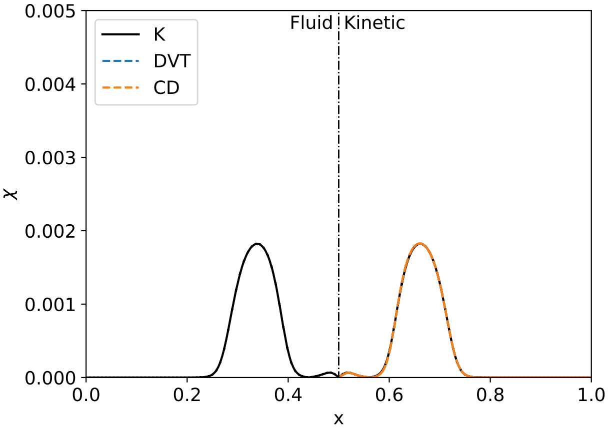
(c) $\nu = 1000$

Figure 7.4: Closeup of n for $\nu = 10$, 100, and 1000 for simulation of the double rarefaction wave problem on a grid of 128×640 second-order polynomial elements at $t = 0.15$. With increasing ν, the fluid and kinetic solutions approach each other, allowing for better matching of the solution in each subdomain using the direct variable translation and composite distribution function methods. Note the DVT and CD lines closely follow each other in Fig. 7.4c for $\nu = 1000$.



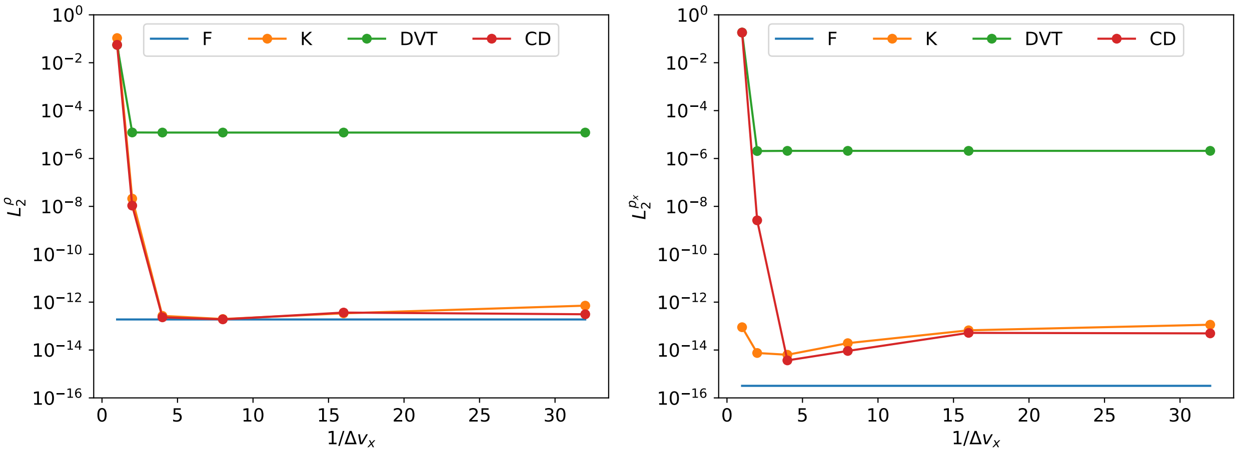
(a) $\nu = 10$

(b) $\nu = 100$



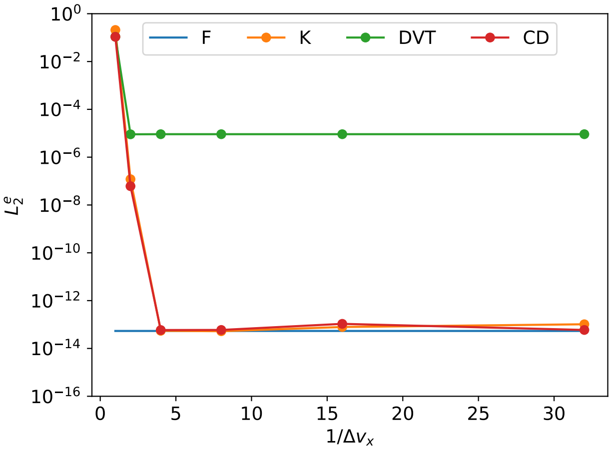
(c) $\nu = 1000$

Figure 7.5: Deviation from a Maxwellian distribution function is indicated by χ, defined in Eq. (3.7.17), for the double rarefaction simulations at $t = 0.15$, measured for a kinetic simulation using 128 × 640 second-order polynomial elements in phase space for $\nu = 10$, 100, and 1000. Values of χ are also plotted on the right subdomain for simulations using the direct variable translation and composite distribution function methods at the same resolution. Values of χ reduce in simulations with higher ν, indicating distribution functions that are closer to Maxwellian. Higher ν also allows for better matching of χ for simulations employing either hybrid method to the kinetic simulation. Note the DVT and CD lines closely follow each other in Fig. 7.5c for $\nu = 1000$. Also note that χ is only plotted on the left subdomain for the kinetic simulation, as it is 0 by definition in the fluid model.



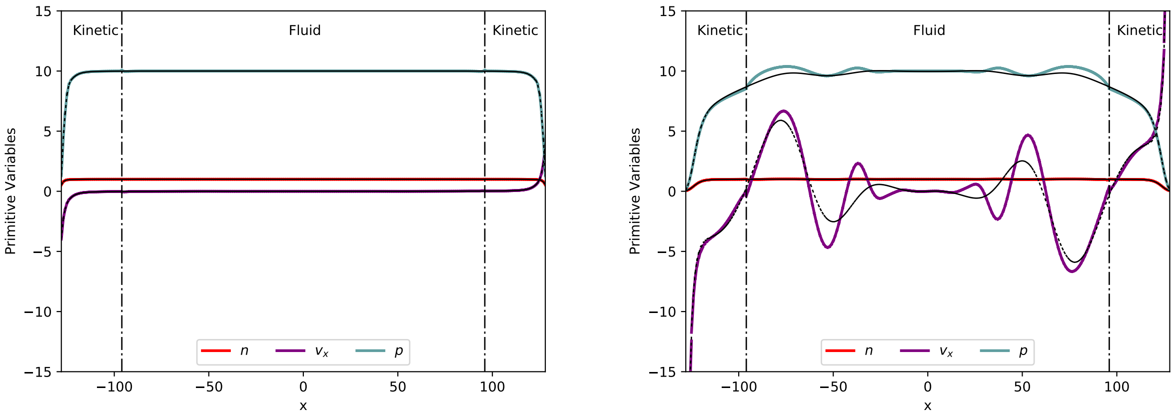
(a) L_2^ρ

(b) $L_2^{p_x}$



(c) L_2^e

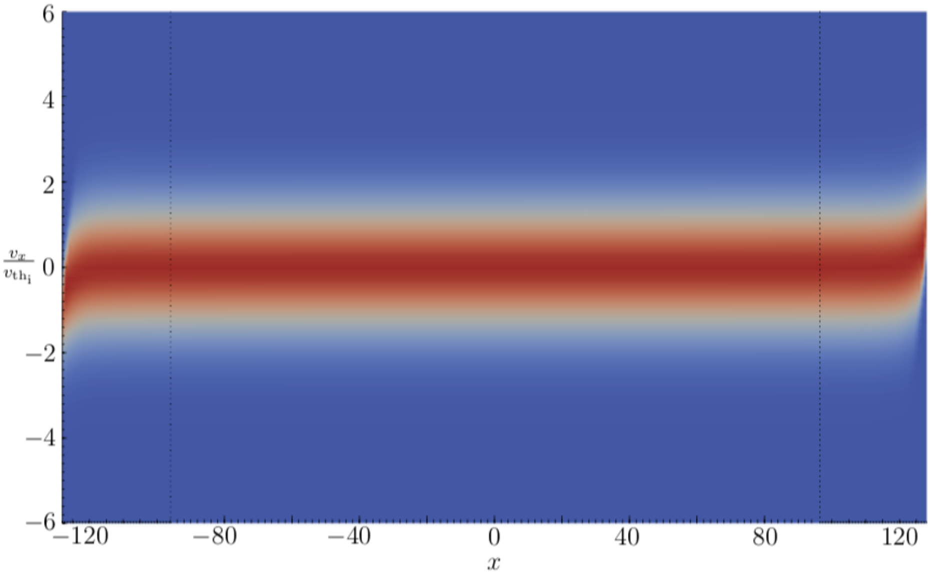
Figure 7.6: The norm $L_2^{\boldsymbol{q}}$ as defined in Eq. (7.2.2) for the double rarefaction wave problem plotted for several velocity-space resolutions using second-order polynomial elements in the velocity direction for $\nu = 1000$. All models use 128 second-order polynomial spatial elements. The values of $L_2^{\boldsymbol{q}}$ reach converged values as the velocity space is accurately resolved when $1/\Delta v_x = 4$, corresponding to the 128×80 resolution case. Mass, momentum, and energy are conserved to a higher degree for the composite distribution function method compared with the direct variable translation method.



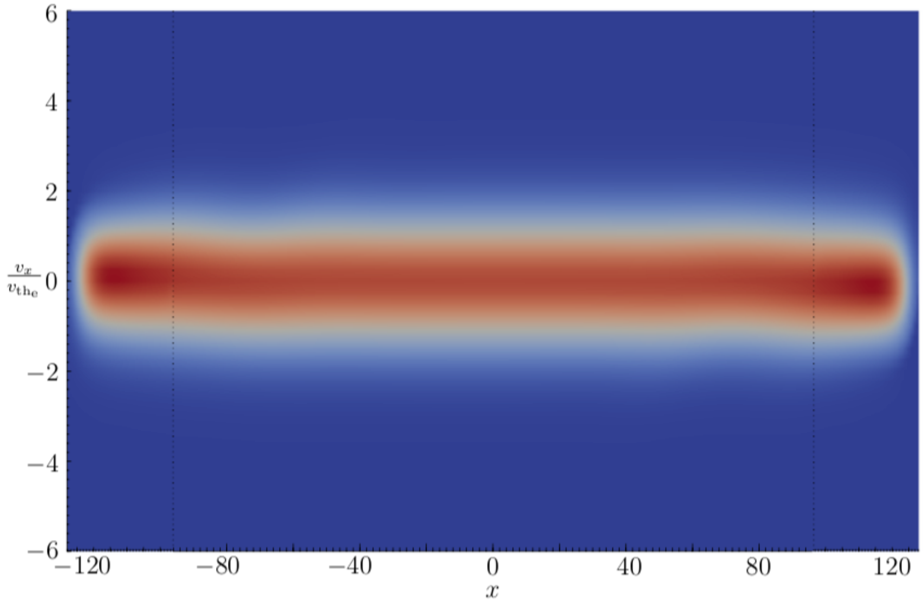
(a) Ions

(b) Electrons

Figure 7.7: Density, velocity, and pressure moments in sheath at $t\omega_{\mathrm{pe}} = 20$ with direct variable translation for ions and electrons. The middle domain is solved using the $5N$-moment two-fluid plasma model while the left and right domains are solved using the continuum kinetic two-species plasma model. Simulation results from applying the continuum kinetic model for the entire domain are plotted in black dashed lines for comparison. Some solution mismatch in the electrons is seen due to departure of the electron distribution function away from a Maxwellian where the model interface assumes validity of the fluid approximation. (The plasma sheath problem is simulated with WARPXM version 1.2.6 / hybrid_kinetics branch using the input file sheath.py for kinetic simulations as well as those using the direct variable translation method while the input file sheath_flux_bc.py is used for simulations involving the composite distribution function method).

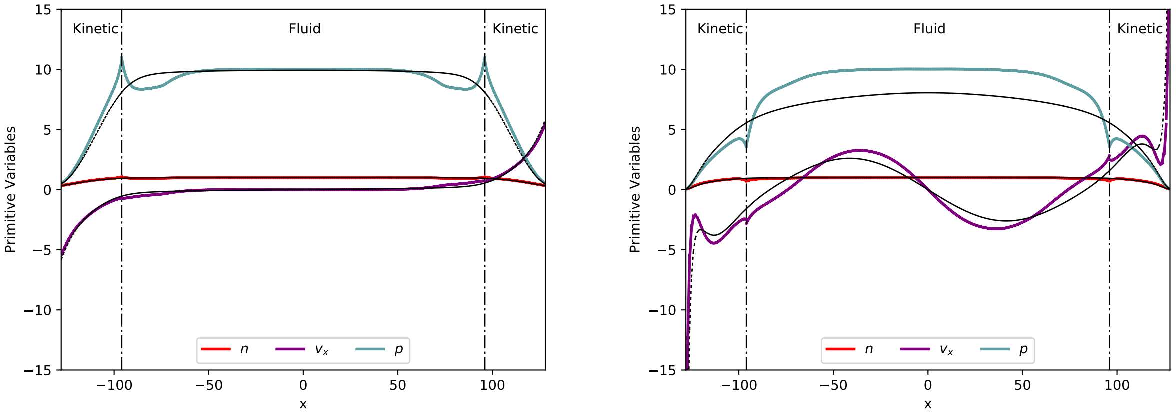


(a) Ions



(b) Electrons

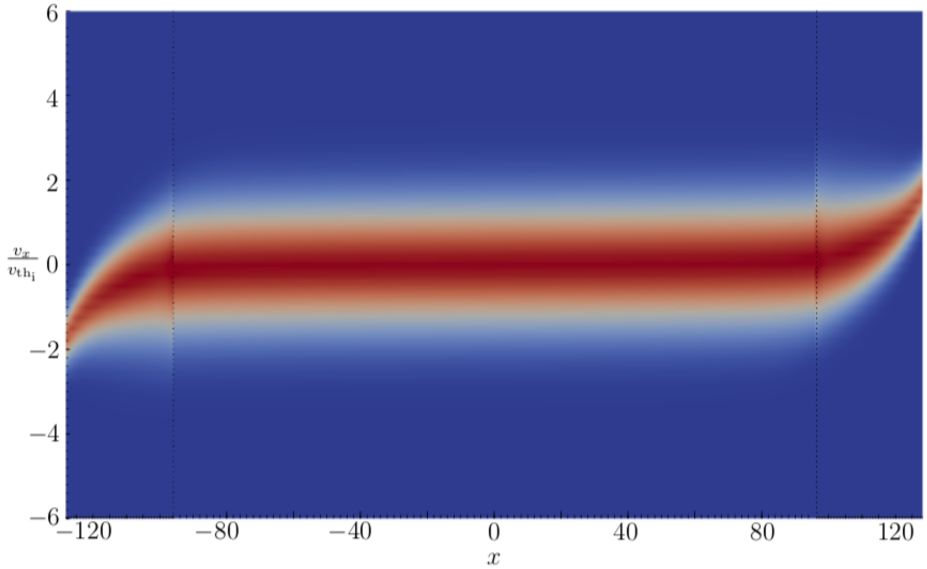
Figure 7.8: Contours of ion and electron distribution functions at $t\omega_{pe} = 20$ solved using the direct variable translation method. The left and right subdomains $x \in [-128, -96]$ and $x \in [96, 128]$ are modeled with the continuum kinetic two-species plasma model, and the middle subdomain x $\in [-96, 96]$ is modeled with the $5N$-moment two-fluid plasma model.



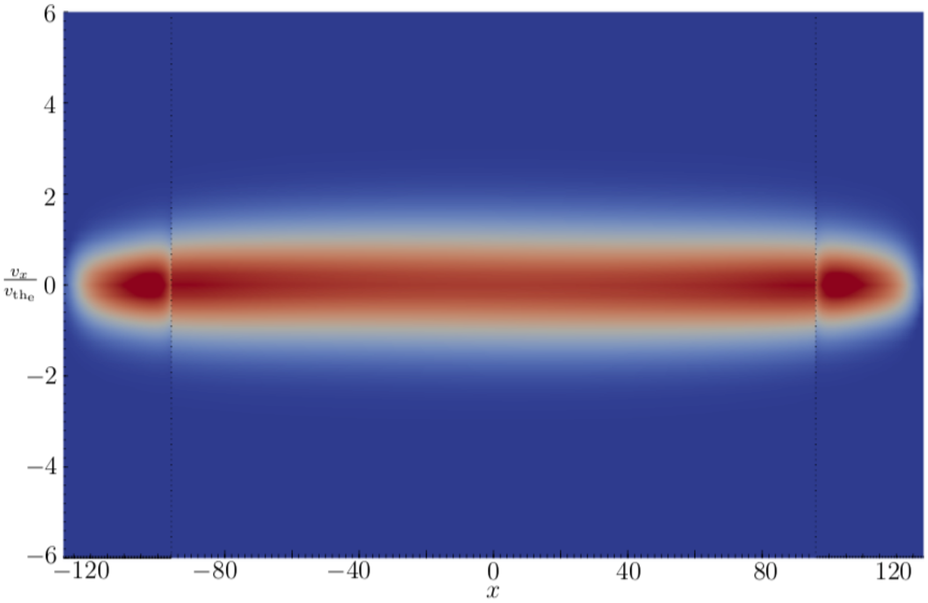
(a) Ions

(b) Electrons

Figure 7.9: Density, velocity, and pressure moments in sheath at $t\omega_{pe} = 200$ with direct variable translation for ions and electrons. The middle domain is solved using the $5N$-moment two-fluid plasma model while the left and right domains are solved using the continuum kinetic two-species plasma model. Simulation results from applying the continuum kinetic model for the entire domain are plotted in black dashed lines for comparison. The mismatch in solution seen at $t\omega_{pe} = 20$ for the electrons continues to evolve at this time, affecting the ion solution.

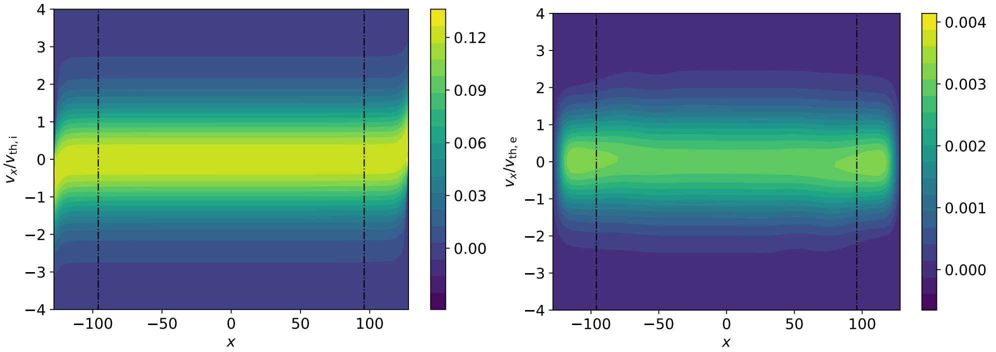


(a) Ions

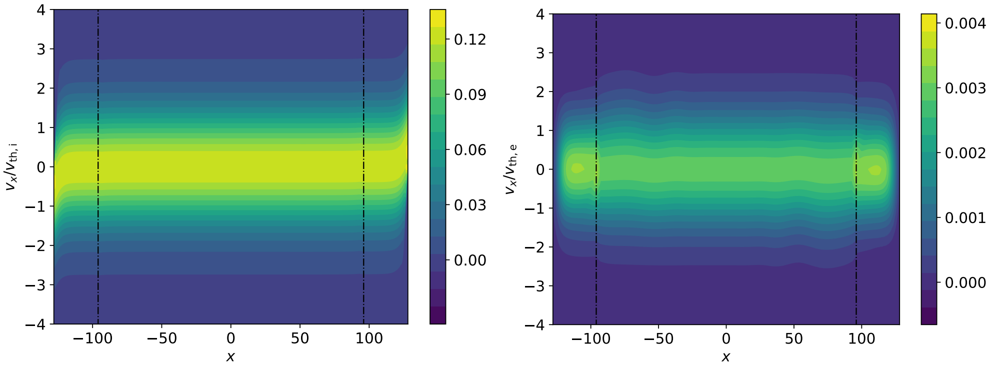


(b) Electrons

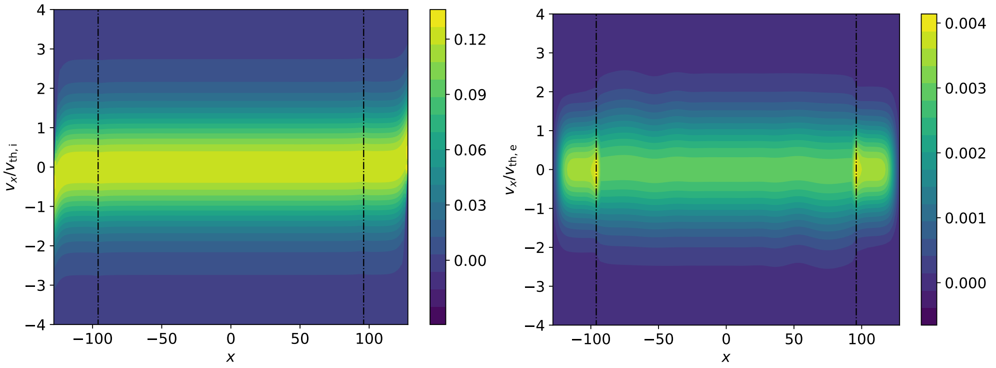
Figure 7.10: Contours of ion and electron distribution functions at $t\omega_{pe} = 200$ solved using the direct variable translation method. The left and right subdomains $x \in [-128, -96]$ and $x \in [96, 128]$ are modeled with the continuum kinetic two-species plasma model, and the middle subdomain x $\in [-96, 96]$ is modeled with the $5N$-moment two-fluid plasma model.



(a) K, Ions (b) K, Electrons

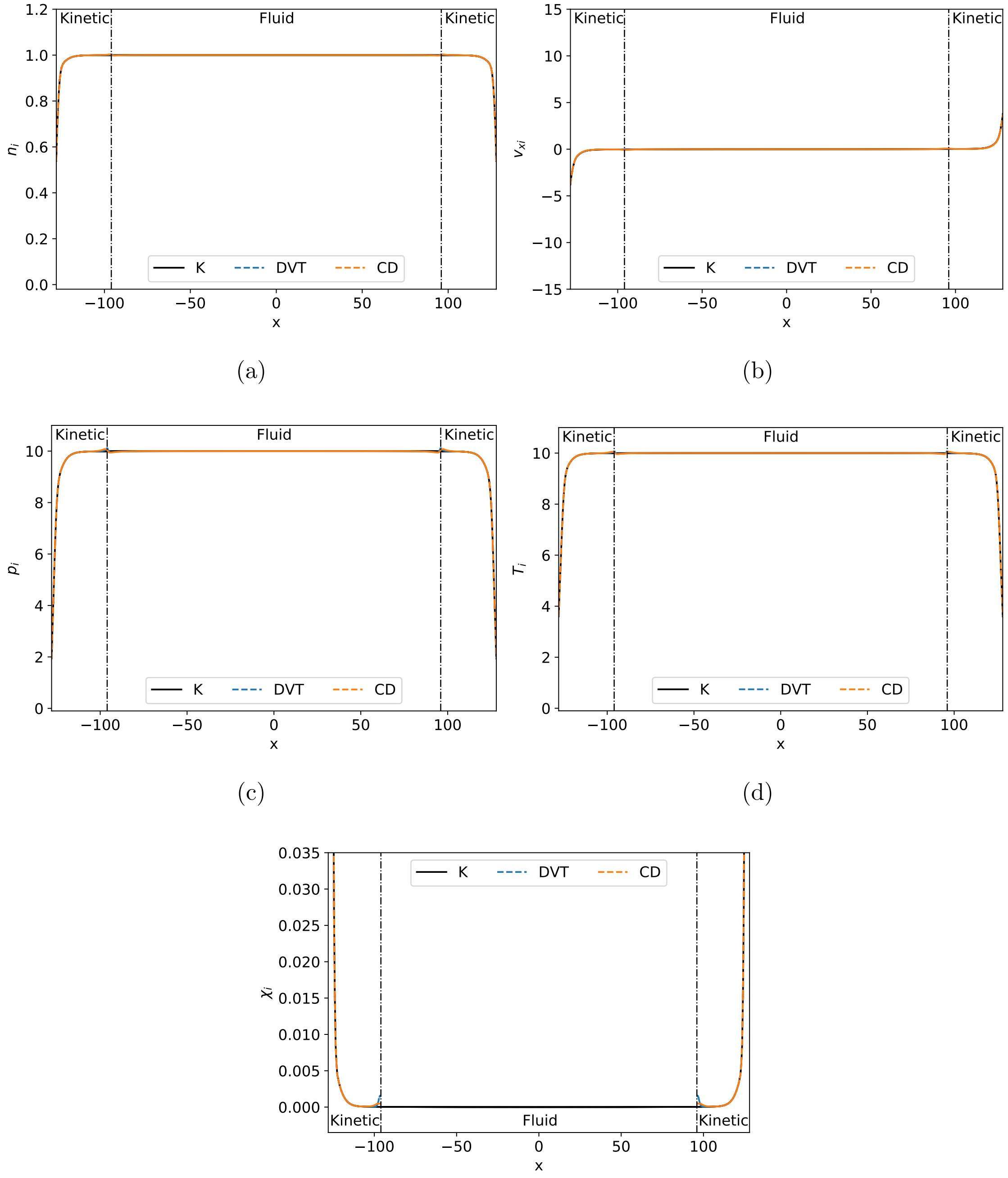


(c) DVT, Ions (d) DVT, Electrons



(e) CD, Ions (f) CD, Electrons

Figure 7.11: Ion and electron distribution functions for a two-species plasma sheath at $t\omega_{pe} = 20$. The kinetic simulation results are shown as well as results for the hybrid simulations using the direct variable translation and composite distribution function methods. For the hybrid simulations, the fluid model is solved on the middle subdomain for $x \in [-96, 96]$ and the kinetic model is solved on the side subdomains for $x \in [-128, -96]$ and $x \in [96, 128]$. Second-order elements are used in physical space with $\Delta x = 1$ and second-order elements are used in velocity space for $v_x \in [-6v_{\mathrm{th}_{\mathrm{i,e}}}, 6v_{\mathrm{th}_{\mathrm{i,e}}}]$ with $\Delta v_{x,\mathrm{i,e}} = v_{\mathrm{th}_{\mathrm{i,e}}}/4$. Maxwell's equations are solved on all subdomains in physical space. Maxwellian distribution functions computed from fluid variables are shown in the middle subdomain where the fluid model is solved in the hybrid simulations. An accumulation of the electron distribution function at the subdomain interfaces at $x_{si} = \pm 96$ is seen in the hybrid simulations, which is more pronounced for the composite distribution function method. The ion distribution functions in the hybrid simulations are indistinguishable from those in the kinetic simulation.



(a)

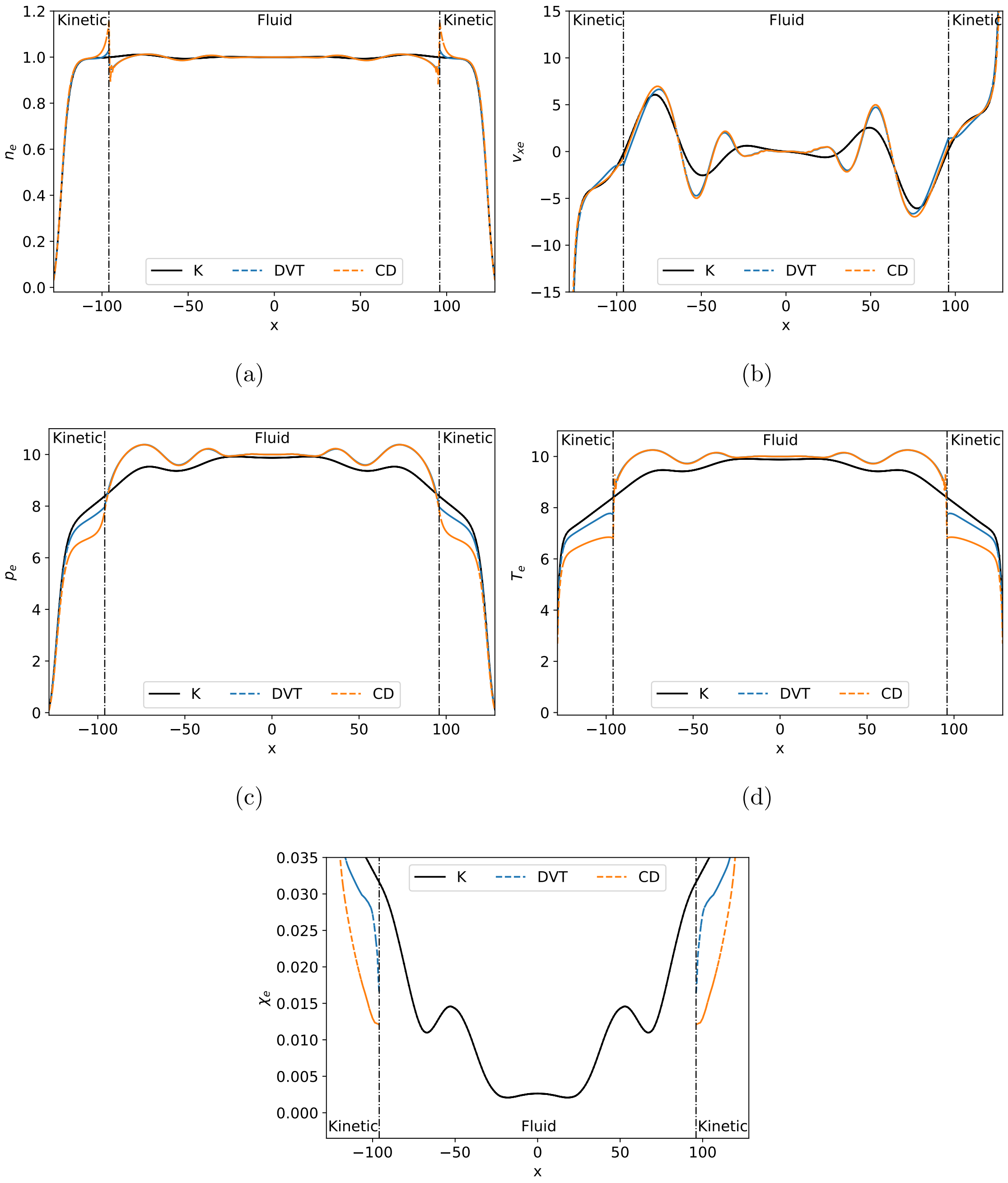
(b)

(c)

(d)

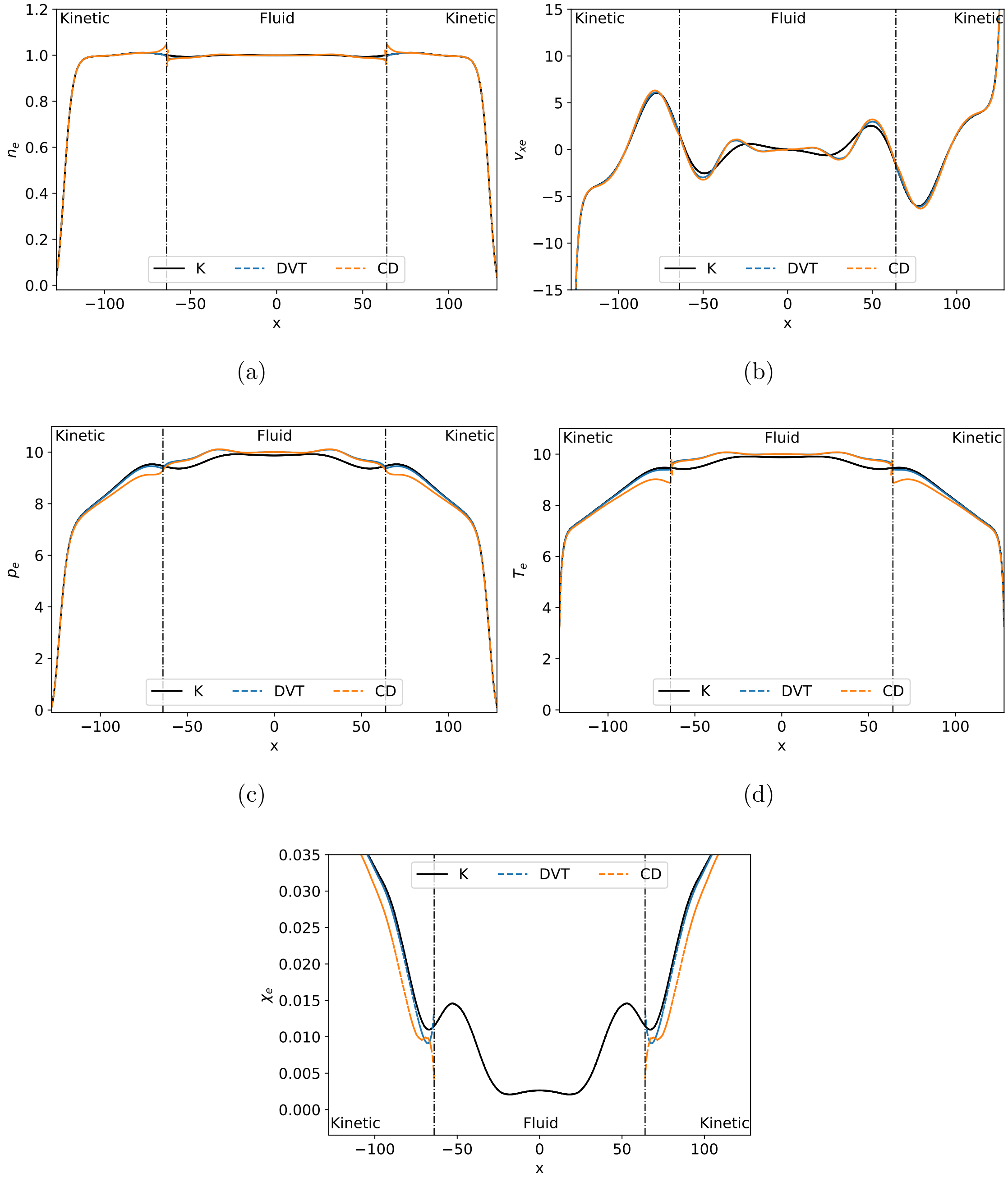
(e)

Figure 7.12: Ion fluid variables n, v_x, p, and T and the χ metric for Maxwellian deviation for the two-species plasma sheath at $t\omega_{pe} = 20$. Simulations involving the kinetic model solved over all subdomains as well as the direct variable translation and composite distribution function methods are plotted. The subdomain interfaces at $x_{si} = \pm 96$ are shown. No large jumps or oscillations of ion fluid variables occur at the subdomain interfaces for the hybrid simulations.



(a) (b) (c) (d) (e)

Figure 7.13: Electron fluid variables n, v_x, p, and T and the χ metric for Maxwellian deviation for the two-species plasma sheath at $t\omega_{pe} = 20$. Simulations involving the kinetic model solved over all subdomains as well as the direct variable translation and composite distribution function methods are plotted. The subdomain interfaces at $x_{si} = \pm 96$ are shown. The simulation using the composite distribution function method exhibits larger jumps in electron density and temperature at the subdomain interfaces than the simulation using the direct variable translation method. Both hybrid methods also produce increased amplitudes of oscillations in the fluid subdomain compared to the kinetic simulation.



(a)

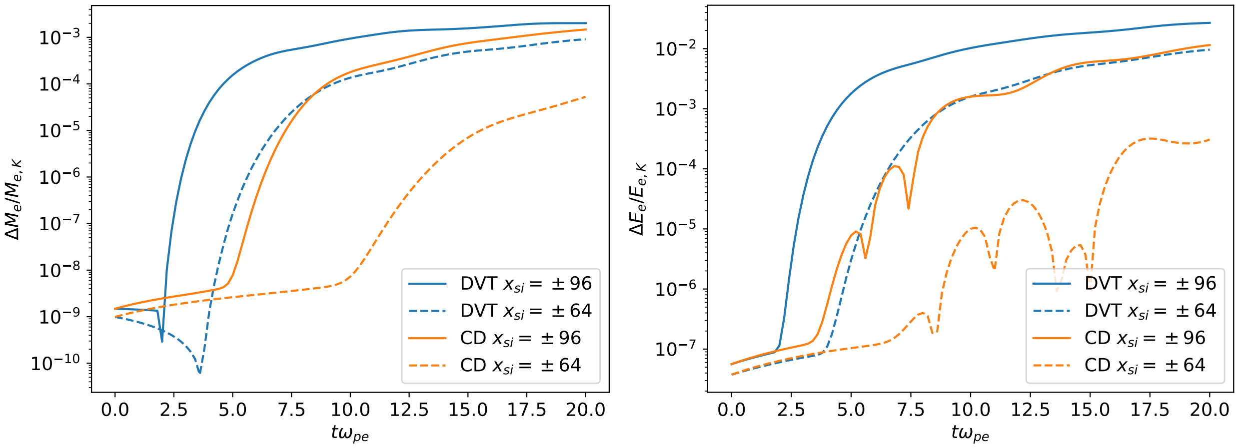
(b)

(c)

(d)

(e)

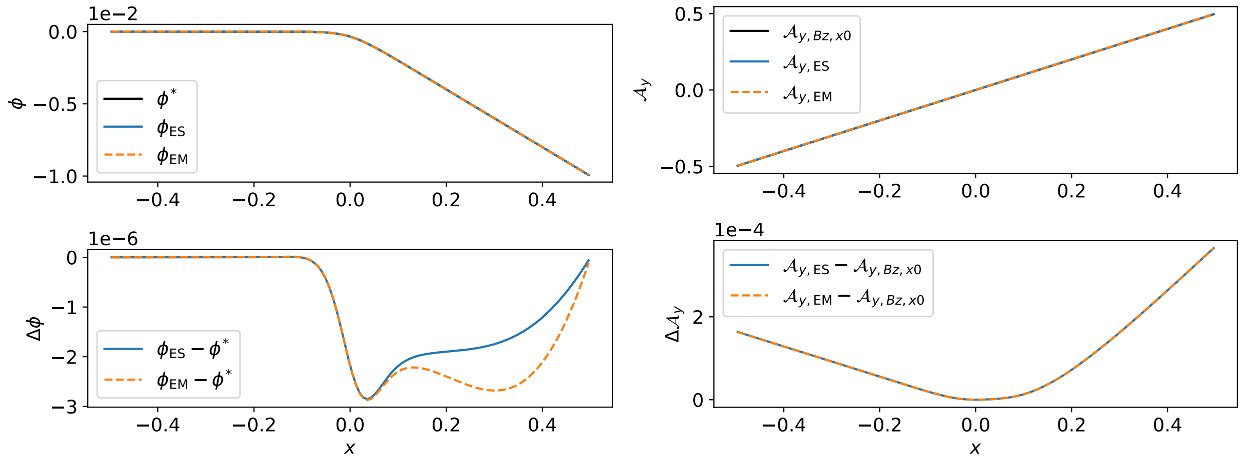
Figure 7.14: Electron fluid variables n, v_x, p, and T and the χ metric for Maxwellian deviation for the two-species plasma sheath at $t\omega_{pe} = 20$, as described in Fig. 7.13, but with larger kinetic subdomains, with the subdomain interfaces at $x_{si} = \pm 64$. Compared with the case of the subdomain interface at $x_{si} = \pm 96$, the jumps in electron density, pressure, and temperature are reduced for the case of the subdomain interface further from the wall at $x_{si} = \pm 64$ where the electron distribution function is closer to Maxwellian. The amplitude of the oscillations in the fluid subdomain is reduced with the expanded kinetic subdomains.



(a) Mass

(b) Energy

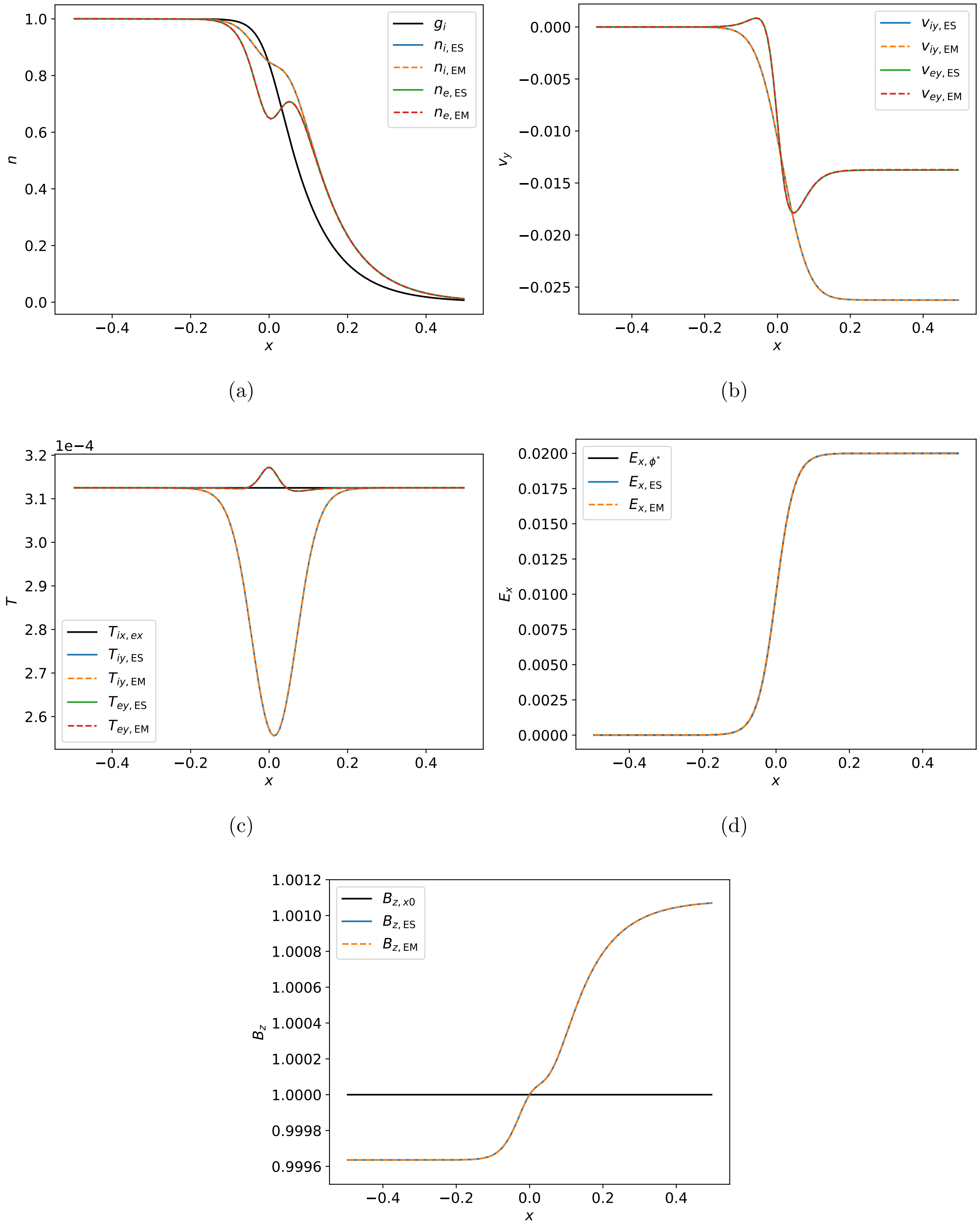
Figure 7.15: Integrated mass and energy for the two-species plasma sheath problem relative to the kinetic simulation for the hybrid simulations using the direct variable translation and composite distribution function methods. The cases with subdomain interfaces at $x_{si} = \pm 96$ and $x_{si} = \pm 64$ are compared. The conservation properties are better for the composite distribution function method than for the variable translation method. However, a larger improvement is achieved by placing the subdomain interface at a location where the fluid model is valid.



(a) ϕ

(b) $\mathcal{A}_y$

Figure 7.16: Profiles of the electrostatic scalar potential and the magnetic vector potential calculated for the initial condition for the magnetized Kelvin-Helmholtz instability. Potentials ϕ^* and $\mathcal{A}_{y,Bz,x0}$ satisfy the fluid equilibrium. Electrostatic quantities (ES) are calculated by deriving the initial condition for the Vlasov-Poisson system. Electromagnetic quantities (EM) are derived by additionally calculating a self-consistent, nonuniform B_z profile according to Eq. (7.6.12), calculating the corresponding magnetic vector potential using Eq. (7.6.4), and recalculating the equilibrium. Since the equilibrium system is for a low-beta plasma, the electrostatic and electromagnetic profiles are similar.



(a)

(b)

(c)

(d)

(e)

Figure 7.17: Profiles of n, v_y, T, E_x, and B_z for the initial condition for the magnetized Kelvin-Helmholtz instability, calculated under electrostatic and electromagnetic assumptions. Figure 7.17e shows the correction to the constant B_z that satisfies Eq. (7.6.12). The electrostatic and electromagnetic profiles are similar, which is expected for the low-beta plasma equilibrium.

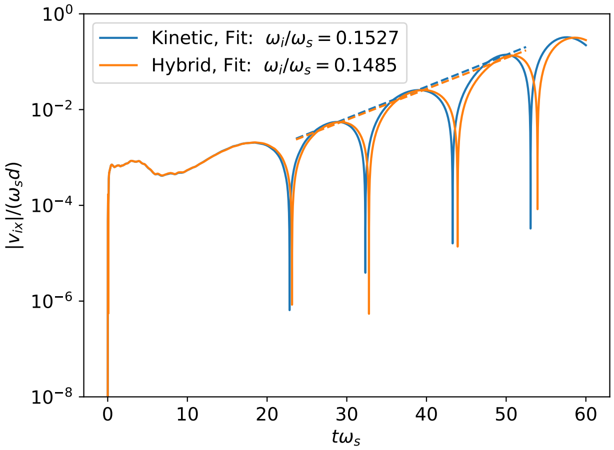
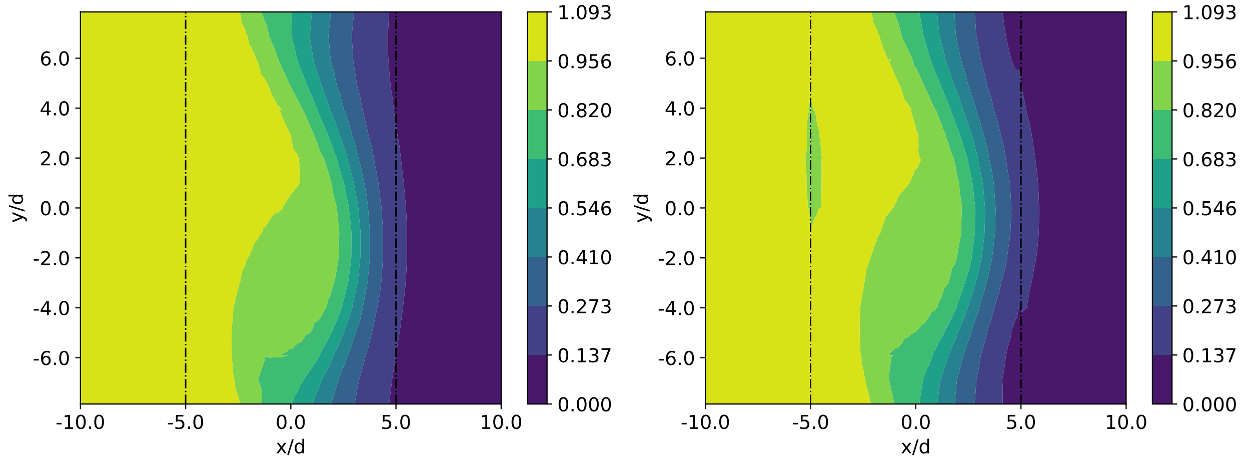


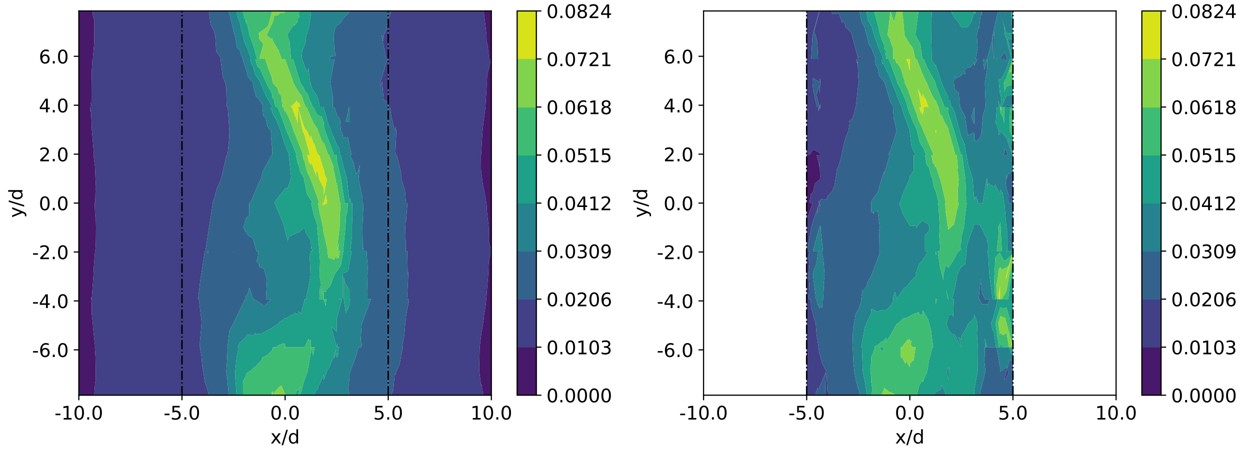
Figure 7.18: Evolution of the ion transverse velocity in the x-direction, $v_{\mathrm{i}x}$, at $(x, y) = (0, 0)$, plotted as $\frac{|v_{\mathrm{i}x}|}{\omega_s d}$ versus $t\omega_s$ for the kinetic and hybrid simulations using $32 \times 16 \times 32 \times 32$ second-order polynomial phase-space elements and third-order explicit Runge-Kutta timestepping. Growth rates of the instability are calculated by fitting a line through the peaks between $t\omega_s = 20$ and $t\omega_s = 50$ before nonlinear saturation occurs. The initial conditions for both simulations are calculated from a kinetic equilibrium consistent with the Vlasov-Maxwell system. In subdomains where the fluid equations are solved, the equilibrium distribution functions are transformed to Maxwellians to initialize the fluid moment variables. The evolution of $v_{\mathrm{i}x}$ in both simulations is similar, indicating the ability of the hybrid method to accurately capture the relevant physical phenomena. (The magnetized Kelvin-Helmholtz instability is simulated with WARPXM version 1.2.5 / hybrid_kinetics branch using input files khiA1_hybrid.py for the kinetic simulation and khiA1_mixed_ion_kinetic_electron.py for the hybrid simulation).



(a) Kinetic

(b) Hybrid

Figure 7.19: Contours of the ion number density, $n_\mathrm{i}(x, y)$, for the magnetized Kelvin-Helmholtz instability at $t\omega_s = 60$ compared for the kinetic and hybrid simulations. Some discontinuities do appear at the subdomain interfaces at $x/d = \pm 5$.



(a) Kinetic

(b) Hybrid

Figure 7.20: Contours of $\chi_{\rm i}(x, y)$ for the magnetized Kelvin-Helmholtz instability at $t\omega_s = 60$ compared for the kinetic and hybrid simulations. The $\chi_{\rm i}$ measurement is performed for $x/d \in [-10, 10]$ in the kinetic simulation and for $x/d \in [-5, 5]$ in the hybrid simulation where the Vlasov-Maxwell equations are solved. The discontinuities in $n_{\rm i}(x, y)$ in Fig. 7.20b at the subdomain interfaces can be explained by the $\chi_{\rm i}$ values at those locations. The Maxwellian deviation is also exacerbated in the hybrid simulation especially at the $x/d = 5$ subdomain interface.

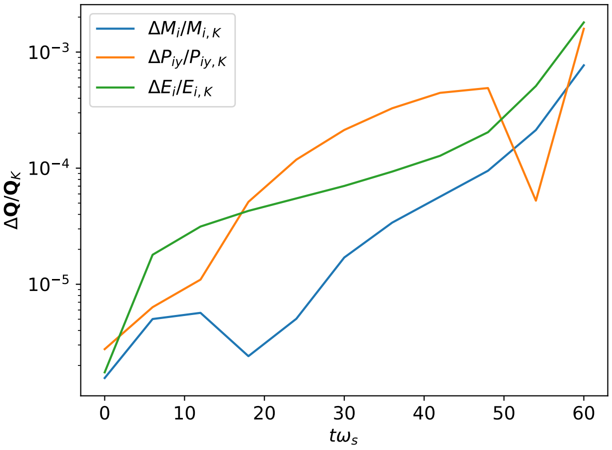


Figure 7.21: Mass, y-momentum, and energy integrated over the physical domain for the hybrid simulation relative to the kinetic simulation, $\Delta\boldsymbol{Q}/\boldsymbol{Q}_K = [\Delta M/M_K, \Delta P_y/P_{y,K}, \Delta E/E_K]$ as described in Eqs. (7.6.16), (7.6.17), and 7.6.18, calculated for ions for the magnetized Kelvin-Helmholtz instability. Calculations are performed at intervals of $t\omega_s = 6$. As the instability develops, the values of χ_i at the subdomain interfaces increase and larger differences between the hybrid and kinetic simulation results are observed, such as the integrated mass, y-momentum, and energy.

Chapter 8

CONCLUSION

The research described in this dissertation continues the work on continuum modeling of plasmas performed by previous graduate students on the WARPX/WARPM/WARPXM group of codes. Namely, model work using the discontinuous Galerkin algorithm is extended onto the unstructured mesh framework of WARPXM, yielding a single code with the MHD model, $5N$-moment fluid model, and continuum kinetic model based on the Vlasov-Maxwell system. This allows for use of any of these models on an arbitrary geometry allowing for realistic simulations of an experiment on unstructured meshes. The unified framework of WARPXM further facilitates development of the domain-decomposed hybridized model discussed in this work. Highlights of these contributions and suggestions for future work are discussed in the rest of this chapter.

8.1 Contributions

In this work, the development of the WARPXM code in C++ is continued. The following represents some of the major contributions.

8.1.1 MHD Model

The MHD model is added to WARPXM to supplement the $5N$-moment model that had previously existed. The MHD model implemented includes Hall, resistive, and viscous terms. To do this, a gradient model has been developed, using the local discontinuous Galerkin and interior penalty methods. For the numerical flux of hyperbolic terms, Rusanov, HLL, HLLD, and Roe methods are also implemented.

8.1.2 5N-moment Model

The gradient module implemented during MHD development also allows for gradient calculations for diffusive terms in the 5N-moment model. Additionally, a few slope-moment limiters are implemented as well as an artificial viscosity limiter, using the gradient module. The number of available approximate Riemann solvers for hyperbolic terms is also expanded for this model, which previously included only the Roe and Rusanov methods, to further include HLLE and HLLEC fluxes.

8.1.3 Flux Boundary Conditions

The ability to perform flux boundary conditions is also implemented. This allows for flux to be specified at a boundary, in addition to the previously-existent method of setting a ghost value and applying a Riemann solver. The flux boundary condition allows for the implementation of the composite distribution function method for the domain-decomposed hybrid method for hybridization between the 5N-moment model and the continuum kinetic model, as discussed in Sec. 3.7.3.

8.1.4 Continuum Kinetic Implementation

The electromagnetic continuum kinetic model is added to WARPXM, coupling the Boltzmann equation with Maxwell's equations. This is performed in 1D1V, 1D2V, and 2D2V using a phase space mesh built upon the unstructured physical space in WARPXM allowing for simulation on unstructured geometries. To solve the Boltzmann equation, the DG algorithm is implemented on a phase space grid, constructed from the unstructured physical space mesh and a Cartesian velocity space that is added. Velocity moments are taken from the velocity space, allowing for moment calculations required for solutions using the continuum kinetic model. In 2D2V, the procedure leads to the usage of a 3-4 duoprism phase space element constructed from the underlying triangular elements in unstructured physical space and line elements in velocity space. Bases of the phase space elements are calculated using tensor products of lower dimensional elements and is currently implemented up to fifth order for the duoprism. A BGK operator is also implemented as a source term.

Validation of implementation

Various test cases are run for standard electrostatic problems such as Landau damping and the two-stream instability, which are compared with theory. Electromagnetic problems involving the Weibel and Dory-Guest-Harris instabilities are also simulated. Particularly interesting results are for the Dory-Guest-Harris instability for which a closed-form integral version of the electromagnetic dispersion relation is derived and compared favorably against simulation. The work extends previous electrostatic treatment of the Dory-Guest-Harris instability to allow for modeling of more general electromagnetic effects.

8.1.5 Domain-Decomposed Hybrid Method

The domain-decomposed hybrid method is described using two separate schemes, one that calculates the flux consistently based on the models, and one that calculates the flux in a conservative manner. These methods are implemented for coupling between the continuum kinetic and $5N$-moment models specifically, with the consistent scheme identified as the direct variable translation method and the conservative scheme identified as the composite distribution function method, in which fluxes are calculated using the same underlying advected distribution function. In the limit of both sides approaching a Maxwellian distribution function, both methods provide consistency and conservation. The direct variable translation method loses conservation properties if the distribution function moves farther from Maxwellian but is found to be more numerically robust than the composite distribution function method. Current implementation involving either method involves a static patch decomposition, in which the subdomains on which each model is simulated remains fixed throughout a simulation.

Test problems

The domain-decomposed hybrid method is used to simulate a 1D1V fluid double rarefaction wave problem to test the direct variable translation and composite distribution function methods using a neutral fluid and with near-Maxwellian distribution functions using a BGK operator. It is then tested on a 1D1V plasma sheath, allowing for deviation from the Maxwellian. It is found that while the composite distribution function method holds conservation properties well even when the kinetic distribution function diverges from Maxwellian, the inconsistency in Riemann problems with the $5N$-moment model leads to inaccurate solutions, manifesting in an accumulation of the distribution function at the subdomain interfaces and leads to significant discontinuities in density and temperature. Similar inaccuracies hold for the direct variable translation method when the distribution functions at the subdomain interfaces are unequal and non-Maxwellian, but the inaccuracies are smaller, leading to a more robust method. If the subdomain interfaces are moved to a location where the fluid model is more accurate, the inaccuracies are found to be reduced. The domain-decomposed hybrid method is then used in a 2D2V simulation of the magnetized Kelvin-Helmholtz instability where the ions are hybridized using the direct variable translation method. Results are promising, showing a similar growth rate of the instability compared to a simulation using the continuum kinetic model throughout all subdomains while reducing simulation time and required computational resources.

8.2 Suggestions for Future Work

Much of the work presented here enhances the capabilities of WARPXM, expanding the models used and modularity of the code to handle them. Moving forward, attention should be placed on speed, making simulation of these models faster while continuing physics development and keeping the code readable for new users to understand and contribute to. The next sections highlight areas that should be focused on in achieving these goals.

8.2.1 Algorithmic Speedup of Physical DG

Current implementation of the DG algorithm splits the calculation of the internal flux (and sources), numerical flux, and the summation of these contributions to the right hand side of the DG equation, which involves iterating through the entire domain 3 times per RK stage of each timestep. This algorithm stems from the legacy of OpenCL, where these operations appeared to be split into parallel onto a GPU. Early work in this research removed OpenCL due to its diminishing support from various operating systems and thus these steps are now performed sequentially. A clear speedup would be to re-write the DG implementation directly as written in Chapter 3, where the right hand side of the DG equation can be calculated directly per element. Additionally, no large arrays holding internal fluxes, numerical fluxes, and sources would be needed, as the right hand side per element node would be updated at the time of its calculation. This element-by-element procedure would also add to the readability of the code and is in fact the method used in the kinetic DG implementation.

8.2.2 Data Parallelism and GPU Offloading

As alluded to in the previous section, OpenCL, which was used to offload calculations onto a GPU was used in the past but has been removed in this work due to diminishing support on various operating systems. Moving forward, adding back data parallelism that OpenCL provided should be made a focus to speed up the DG calculations for the physical and kinetic models. While the paradigm of heterogeneous computing continues to mature, there are still various options that can be used, including OpenMP, OpenACC, and CUDA. Future work should choose a language that can be easily used with the existing code to include data parallelism, whether across CPU, GPU and/or accelerator architecture. This could add much speedup and is critical for simulation especially of the kinetic model in high dimension, since the velocity space is not subdivided into MPI ranks (see next section). The additional maturation of GPU technology will hopefully make use of these devices simpler than in the past.

8.2.3 Load Balancing and Dynamic Domain Decomposition

The code currently uses the Metis graph partitioning library to subdivide the domain into separate patches, each usually assigned to an MPI process to provide task parallelism. This domain decomposition assumes equal weights among patches to divide the domain such that roughly the same computation work is required on each patch. This may be a good method for simulations of a single model where each part of the domain is expected to have the same physics. This is likely not appropriate for hybrid simulations employing different plasma models. This is particularly true for hybrid simulations of fluid and kinetic models, where the kinetic subdomains must consist of a velocity space that is not subdivided into patches on different MPI ranks. Thus, adjustments of the weights should be considered for these simulations. Control of these weights will also be needed for the future aim of dynamically changing subdomains in which each model is simulated, based on physical criteria of the applicability of each model.

8.2.4 Kinetic Algorithm

In writing the kinetic DG method using the unstructured physical space, various lookups are used which may not be needed in a fully structured algorithm, though there may be ways to make these more efficient. The upwinding flux is written also in a single module for all dimensionalities. It may be more efficient to write specific forms for 1D1V, 1D2V, 2D2V, etc. to make this a faster function call.

8.2.5 CSR Method and Union of Kinetic Flux Calculations

In development of the kinetic DG method for the 2D2V duoprism element, it was realized that this element can have faces with a different number of nodes per face. This changes the details of the flux implementation. It was also noticed that the sparsity of arrays at these higher dimensions becomes noticable and thus a CSR format was used for the DG method using these elements, but not for elements at the lower dimensions. Moving forward, a single

DG flux calculation module should be written to handle all types of phase space elements in CSR format for speedup of the algorithm.

8.2.6 Transition Region for the Hybrid Model

A natural next step in the domain-decomposed hybrid method would be to allow for a transition region of elements where the solution smoothly transitions from one model to another. This may yield a more robust method than either the direct variable translation or composite distribution function methods. One could possibly use infrastructure separately developed for Lacuna boundaries using a smoothly-varying parameter from 0 to 1 in the transition region to smoothly transition between the models.

8.2.7 Implementation of other Time-Stepping Schemes

The explicit Runge-Kutta schemes used in this work are beneficial partly due to ease of implementation, but can cause severe timestepping restrictions with high collisionality or resolution. A host of other timestepping schemes including implicit or semi-implicit could be explored to reduce these restrictions for the various models.

8.2.8 Efficient Implementation of Basis Arrays for Higher Order

Currently a Mathematica script is used to calculate basis arrays to high precision analytically. For higher order bases and dimensions, these can take significant time to run and the associated text file size may be large. Migrating these calculations to c++ or python, perhaps using a high order precision library such as the GMP library may be beneficial, both for keeping the code away from a proprietary language and holding the arrays in some format that does not add significantly to the size of the repository of the code.

8.2.9 Poisson Implementation

An implementation of the Poisson equation for integration with the kinetic model would be useful for comparison of continuum kinetics using electromagnetics and electrostatics in the same code. The use of the Petsc library, which is already a dependency of WARPXM could be used to implement this.

8.2.10 Documentation of Code and Software Development

As the WARPXM code continues development, it is important reduce the barrier of entry for new users to add contributions. Proper documentation should be stressed and good software engineering practices should be used. During the course of this work, procedures for code reviews have been developed and adjusted, allowing for feedback from peers on coding practices and reduction in difficulties associated with merging branches between different users. Evolution toward modern software development techniques such as the use of continuous integration and helpful testing suites will aid in the construction of new physics and algorithm advancement moving forward.

BIBLIOGRAPHY

[1] Hesam Abbassi, Farzad Mashayk, and Gustaaf B. Jacobs. Shock capturing with entropy-based artificial viscosity for staggared grid discontinuous spectral element method. *Computers & Fluids*, 98:152–163, Feb 2014.

[2] Douglas N. Arnold, Franco Brezzi, Bernando Cockburn, and L. Donatella Marini. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems. *SIAM Journal of Numerical Analysis*, 39(5):1749–1779, 2002.

[3] Dinshaw S. Balsara, Christoph Altmann, Clas-Dieter Munz, and Michael Dumbser. A sub-cell based indicator for troubled zones in RKDG schemes and a novel class of hybrid RKDG+HWENO schemes. *Journal of Computational Physics*, 226(1):586–620, Sep 2007.

[4] Paul M. Bellan. *Spheromaks: A Practical Application of Magnetohydrodynamic Dynamos.* Imperial College Press, London, UK, 2000.

[5] Ira B. Bernstein. Waves in a Plasma in a Magnetic Field. *Physical Review*, 109(1):10–21, 1958.

[6] P. L. Bhatnagar, E. P. Gross, and M. Krook. A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems. *Physical Review*, 94(3):511–525, 1954.

[7] C. K. Birdsall and A. B. Langdon. *Plasma Physics via Computer Simulation.* IOP Publishing Ltd., Bristol, UK, 1991.

[8] J. Birn. GEM Magnetic Reconnection Challenge. `http://terpconnect.umd.edu/~drake/publications/reconnection/birn01.pdf`. Accessed: 2017-02-16.

[9] 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, and P. L. Pritchett. Geospace Environmental Modeling (GEM) Magnetic Reconnection Challenge. *Journal of Geophysical Research*, 106(A3):3715–3719, 2001.

[10] L. P. Biro, K. Kertesz, Z. Vertesy, G. I. Mark, Zs. Balint, V. Lousee, and J.-P. Vigneron. Living photonic crystals: Buttery scales - Nanostructure and optical properties. *Materials Science and Engineering: C*, 27(5-8):941–946, Sep 2007.

[11] Rupak Biswas, Karen D. Devine, and Joseph E. Flaherty. Parallel, adaptive finite element methods for conservations. *Applied Numerical Mathematics*, 14(1-3):255–283, Apr 1994.

[12] S. I. Braginskii. Transport Processes in Plasma. *Reviews of Plasma Physics*, pages 205–311, 1965.

[13] M. Brio and C. C. Wu. An upwind differencing scheme for the equations of ideal magnetohydrodynamics. *Journal of Computational Physics*, 75(2):400–422, Apr 1988.

[14] Thierry Buffard, Thierry Gallouët, and Jean-Marc Hérard. A sequel to a rough Godunov scheme: application to real gases. *Computers & Fluids*, 29(7):813–847, 2000.

[15] A. Burbeau, P. Sagaut, and Ch.-H. Bruneau. A Problem-Independent Limiter for High-Order Runge-Kutta Discontinuous Galerkin Methods. *Journal of Computational Physics*, 169:111–150, May 2001.

[16] Rommel Bustinza and Gabriel N. Gatica. A Local Discontinuous Galerkin Method for Nonlinear Diffusion Problems with Mixed Boundary Conditions. *SIAM Journal of Scientific Computing*, 26(1):152–177, 2004.

[17] P. Cagas, A. Hakim, J. Juno, and B. Srinivasan. Continuum kinetic and multi-fluid simulations of classical sheaths. *Physics of Plasmas*, 24(022118):1–11, 2017.

[18] P. Cagas, A. Hakim, W. Scales, and B. Srinivasan. Erratum: "Nonlinear saturation of the Weibel intability [Phys. Plasmas 24, 112116 (2017)]". *Physics of Plasmas*, 26(049904):1, 2019.

[19] Petr Cagas. *Continuum Kinetic Simulations of Plasma Sheaths and Instabilities*. PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Va, July 2018.

[20] Xiaofeng Cai, Xiangxiong Zhang, and Jianxian Qiu. Positivity-Preserving High Order Finite Volume HWENO Schemes for Compressible Euler Equations. *Journal of Scientific Computing*, 68:464–483, 2016.

[21] Paul Castillo, Bernardo Cockburn, Ilaria Perugia, and Dominik Schotzau. An A Priori Error Analysis of the Local Discontinuous Galerkin Method for Elliptic Problems. *SIAM Journal of Numerical Analysis*, 38(5):1676–1706, 2000.

[22] Francis F. Chen. *Introduction to Plasma Physics and Controlled Fusion Volume 1: Plasma Physics*. Plenum Press, New York, 2nd edition, 1984.

[23] C. Z. Cheng and Georg Knorr. The Integration of the Vlasov Equation in Configuration Space. *Journal of Computational Physics*, 22(3):330–351, 1976.

[24] Shun Yan Cheung. Working with Sparse Matrices. `http://www.mathcs.emory.edu/~cheung/Courses/561/Syllabus/3-C/sparse.html`. Accessed: 2021-02-23.

[25] Bernardo Cockburn and Clint Dawson. Some Extensions of theLocal Discontinuous Galerkin Method for Convection-Diffusion Equations in Multidimensions. In J. R. Whiteman, editor, *Proceedings of the Conference on the Mathematics of Finite Elements and Applications*, pages 225–238. Elsevier, 2000.

[26] Bernardo Cockburn, Suchung Hou, and Chi-Wang Shu. The Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation Laws IV: The multidimensional case. *Mathematics of Computation*, 54(190):545–581, Apr 1990.

[27] Bernardo Cockburn, Guido Kanshat, Ilaria Perugian, and Dominik Schotzau. Superconvergence of the Local Discontinuous Galerkin method for Elliptic Problems on Cartesian Grids. *SIAM Journal of Numerical Analysis*, 39(1):264–285, 2001.

[28] Bernardo Cockburn, San-Yih Lin, and Chi-Wang Shu. TVB Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation Laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, Sep 1989.

[29] Bernardo Cockburn and Chi-Wang Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation Laws II: General Framework. *Mathematics of Computation*, 52(186):411–435, Apr 1989.

[30] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta local Projection P1-discontinuous Galerkin finite element method for Scalar Conservation Laws. *Modelisation mathematique et analyse numerique*, 25(3):337–361, 1991.

[31] Bernardo Cockburn and Chi-Wang Shu. The Local Discontinuous Galerkin Method for Time-Dependent Convection-Diffusion Systems. *SIAM Journal of Numerical Analysis*, 35(6):2440–2463, Dec 1998.

[32] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation Laws V: Multidimensional Systems. *Journal of Computational Physics*, 141(2):199–224, Apr 1998.

[33] Bernardo Cockburn and Chi-Wang Shu. Runge-Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems. *Journal of Scientific Computing*, 16(3):173–261, September 2001.

[34] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74, December 1928.

[35] H. S. M. Coxeter. *Regular Polytopes*. Methuen & Co. Ltd., London, UK, 1948.

[36] F. W. Crawford. A review of cyclotron harmonic phenomena in plasmas. *Nuclear Fusion*, 5(1):73–84, 1965.

[37] A. Dedner, F. Klemm, D. Kroner, C.-D. Munz, T. Schnitzer, and M Wesenberg. Hyprbolic Divergence Celeaning for the MHD Equations. *Journal of Computational Physics*, 175(2):645–673, January 2002.

[38] Paul J. Dellar. Lattice Kinetic Schemes for Magnetohydrodynamics. *Journal of Computational Physics*, 179:95–126, Jan 2002.

[39] R. A. Dory, G. E. Guest, and E. G. Harris. Unstable electrostatic plasma waves propagating perpendicular to a magnetic field. *Physical Review Letters*, 14(5):131–133, 1965.

[40] W. E. Drummond and M. N. Rosenbluth. Cyclotron Radiation from a Hot Plasma. *The Physics of Fluids*, 2(1):45–51, 1960.

[41] Michael Dumbser, Olindo Zanotti, Raphael Loubere, and Steven Diot. A posteriori subcell limiting of the discontinuous galerkin finite element method for hyperbolic conservation laws. *Journal of Computational Physics*, 278:47–75, Aug 2014.

[42] Paul K. Chu (ed.) and XinPei Lu (ed.). *Low Temperature Plasma Technology: Methods and Applications*. CRC Press, Taylor & Francis Group, Boca Raton, FL, 2014.

[43] A. Erdélyi, W. Magnus, F. Oberhettinger, F.G. Tricomi, and H. Bateman. *Higher Transcendental Functions, California Institute of Technology H. Bateman Manuscript Project*, volume 2. McGraw-Hill, New York, NY, 2003.

[44] Jeffrey Freidberg. *Plasma Physics and Fusion Energy*. Cambridge University Press, Cambridge, UK, 2007.

[45] Burton D. Fried. Mechanism for Instability of Transverse Plasma Waves. *The Physics of Fluids*, 2:337, 1959.

[46] E. Gomez, D. Amutha Rani, C. R. Cheeseman, D. Deegan, M. Wise, and A. R. Boccaccini. Thermal plasma technology for the treatment of wastes: A critical review. *Journal of Hazardous Materials*, 161(2-3):614–626, Jan 2009.

[47] Sigal Gottlieb and Chi-Wang Shu. Total Variation Diminishing Runge-Kutta Schemes. *Mathematics of Computation*, 267(221):73–85, Jan 1998.

[48] Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong Stability-Preserving High-Order Time Discretization Methods. *SIAM Review*, 43(1):89–112, 2001.

[49] Donald A Gurnett and Amitava Bhattacharjee. *Introduction to Plasma Physics With Space, Laboratory and Astrophysical Applications*. Cambridge University Press, New York, New York, 2017.

[50] M. G. Haines. A review of the dense Z-pinch. *Plasma Phys. Control. Fusion*, 53(093001):1–158, Jun 2011.

[51] A. Hakim and U. Shumlak. Two-fluid physics and field-reversed configurations. *Physics of Plasmas*, 14(055911):1–11, 2007.

[52] Ammar H. Hakim. *High Resolution Wave Propagation Schemes for Two-Fluid Plasma Simulations*. PhD thesis, University of Washington, Seattle, WA, 2006.

[53] Shao-Qiang Han, Wen-Ping Song, and Zhong-Hua Han. An improved WENO method based on Gauss-kriging reconstruction with an optimized hyper-parameter. *Journal of Computational Physics*, 422:109742, 2020.

[54] Amiram Harten, Peter D. Lax, and Bram Van Leer. On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws. *SIAM Review*, 25(1):35–61, 1983.

[55] Jan S. Hesthaven and Tim Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*. Springer Science+Business Media, LLC, New York, USA, 2008.

[56] Fred L. Hinton. Collisonal Transport in Plasma. In M. N. Rosenbluth, R. Z. Sagdeev, A. A. Guleev, and R. N. Sudan, editors, *Handbook of Plasma Physics*, volume 1, chapter 1.5, pages 147–197. North-Holand Publishing Company, 1983.

[57] Charles Hirsch. *Numerical Computation of Internal and External Flows*, volume 1. Butterworth-Heinemann, Oxford, United Kingdom, 2 edition, 2007.

[58] J. D. Huba. *NRL Plasma Formulary*. Naval Research Laboratory, Washington DC 20375-5320, nrl/pu/6790–09-523 edition, 2009.

[59] Richard F. Hubbard and Thomas J. Birmingham. Electrostatic Emissions Between Electron Gyroharmonics in the Outer Magnetosphere. *Journal of Geophysical Research*, 83(A10):4837–4850, 1978.

[60] George Karypis. *METIS A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices*. University of Minnesota, Department of Computer Science & Engineering, University of Minnesota Minneapolis, MN 55455, 5.1.0 edition, March 2013.

[61] George Karypis and Vipin Kumar. A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM Journal on Scientific Computing*, 20(1):359—392, 1999.

[62] B. G. Korenev. *Bessel Functions and their Applications*. Taylor & Francis, London, UK, 2002.

[63] Nicholas A. Krall and Alvin W. Trivelpiece. *Principles of Plasma Physics*. McGraw-Hill, New York, 1973.

[64] Lilia Krivodona. Limiters for high-order discontinuous Galerkin methods. *Journal of Computational Physics*, 226:879–896, May 2007.

[65] L. Landau. On the vibrations of the electronic plasma. *Journal of Physics*, 10(1):25–34, 1946.

[66] L. D. Landau and E. M. Lifshitz. *Physical Kinetics*, volume 10 of *Landau and Lifshitz: Course of Theoretical Physics*. Pergamon Press Ltd., Oxford, UK, 1st edition, 1981.

[67] I. Lerche. Transverse Waves in a Relativistic Plasma. *The Physics of Fluids*, 9(6):1073–1080, 1966.

[68] I. Lerche. Plasma Waves in Oscillating Electromagnetic Fields: Construction of the Dispersion Relation. *Plasma Physics*, 16:955–967, 1974.

[69] I. Lerche, R. Schlickeiser, and R. C. Tautz. Comment on “A new derivation of the plasma susceptibility tensor for a hot magnetized plasma without infinite sums of products of Bessel functions” [Phys. Plasmas 14, 092103 (2007)]. *Physics of Plasmas*, 15:024701, 2008.

[70] Randall J. Leveque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, Delhi, India, 2002.

[71] Michael A. Liberman, John S. De Groot, Arthur Toor, and Rick B. Spielman. *Physics of High-Density Z-Pinch Plasmas*. Springer-Verlag, New York, New York, 1999.

[72] John Loverich. *A Discontinuous Galerkin Method for the Two-Fluid Plasma System and Its Application to the Z-Pinch*. PhD thesis, University of Washington, Seattle, WA, 2005.

[73] John Loverich, Ammar Hakim, and Uri Shumlak. A Discontinuous Galerkin Method for Ideal Two-Fluid Plasma Equations. *Communications in Computational Physics*, 9(2):240–268, 2011.

[74] Hong Luo, Joseph D. Baum, and Rainald Lohner. A Hermite WENO-based limiter for discontinuous Galerkin method on unstructured grids. *Journal of Computational Physics*, 225(1):686–713, Jul 2007.

[75] Daniel O. Martinez, Shiyi Chen, and William H. Matthaeus. Lattice Boltzmann magnetohydrodynamics. *Physics of Plasmas*, 1(6):1850–1867, June 1994.

[76] E. T. Meier and U. Shumlak. A general nonlinear fluid model for reacting plasma-neutral mixtures. *Physics of Plasmas*, 19(072508):1–11, 2012.

[77] A. B. Mikhailovskii. *Electromagnetic Instabilities in an Inhomogenous Plasma*, volume 1. IOP Publishing Ltd, New York, New York, 1992.

[78] S. T. Miller and U. Shumlak. A multi-species 13-moment model for moderately collisional plasmas. *Physics of Plasmas*, 23(8):082303, aug 2016.

[79] Sean Miller. *Modeling collisional processes in plasmas using discontinuous numerical methods*. PhD thesis, University of Washington, Seattle, WA, 2016.

[80] Takahiro Miyoshi and Kanya Kusano. A multi-state HLL approximate Riemann solver for ideal magnetohydrodynamics. *Journal of Computational Physics*, 208:315–344, Feb 2005.

[81] Scott Moe, James Rossmanith, and David Seal. A Simple and Effective High-Order Shock-Capturing Limiter for Discontinuous Galerkin Methods. *arXiv*, arXiv:1507.03024, 07 2015.

[82] V. A. Nemchinsky and W. S. Severance. What we know and what we do not know about plasma arc cutting. *Journal of Physics D: Applied Physics*, 39:R423–R438, Nov 2006.

[83] Barry S. Newberger. New sum rule for products of Bessel functions with application to plasma physics. *Journal of Mathematical Physics*, 23:1278–1281, 1982.

[84] Barry S. Newberger. Erratum: New sum rule for products of Bessel functions with application to plasma physics [J. Math. Phys. 23, 1278 (1982)]. *Journal of Mathematical Physics*, 24:2250, 1983.

[85] Dwight R. Nicholson. *Introduction to Plasma Theory*. John Wiley & Sons, New York, 1983.

[86] Steven A. Orszag and Cha-Mei Tang. Small-scale structure of two-dimensional magenetohydrodynamic turbulence. *Journal of Fluid Mechanics*, 90(1):129–143, Oct 1977.

[87] Edward Ott, Bertram Hui, and K. R. Chu. Theory of electron cyclotron resonance heating of tokamak plasmas. *The Physics of Fluids*, 23(5):1031–1045, 1980.

[88] J. Peraire and P.-O. Persson. The Compact Discontinous Galerkin (CDG) Method for Elliptic Problems. *SIAM Journal of Scientific Computing*, 30(4):1806–1824, 2008.

[89] Sylvaine Perraut, Alain Roux, Patrick Robert, Roger Gendrin, Jean-André Sauvaud, Jean-Michel Bosqued, Gerhard Kremser, and Axel Korth. A Systematic Study of ULF Waves Above FH+ From GEOS 1 and 2 Measurements and Their Relationships with Protron Ring Distributions. *Journal of Geophysical Research*, 87(A8):6219–6236, 1982.

[90] Per-Olof Persson and Jaime Peraire. Sub-Cell Shock Capturing for Discontinuous Galerkin Methods. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, pages 1–13. AIAA, January 2006.

[91] R. F. Post and M. N. Rosenbluth. Electrostatic Instabilities in Finite Mirror-Confined Plasmas. *Physics of Fluids*, 9(4):730–749, 1966.

[92] Kenneth G. Powell, Philip L. Roe, Timur J. Linde, Tamas I. Gombosi, and Darren L. De Zeeuw. A Solution-Adaptive Upwind Scheme for Ideal Magnetohydrodynamics. *Journal of Computational Physics*, 154(2):284–309, 1999.

[93] Kenneth G. Powell, Philip L. Roe, Rho Shin Myong, Tamas I. Gombosi, and Darren De Zeeuw. An Upwind Scheme for Magnetohydrodynamics. In *12th AIAA Computational Fluid Dynamics Conference: a collection of technical papers*, pages 661–674. AIAA, June 1995.

[94] Hong Qin, Cynthia K. Phillips, and Ronald C. Davidson. A new derivation of the plasma susceptibility tensor for a hot magnetized plasma without infinite sums of products of Bessel functions. *Physics of Plasmas*, 14:092103, 2007.

[95] Hong Qin, Cynthia K. Phillips, and Ronald C. Davidson. Response to “Comment on ‘A new derivation of the plasma susceptibility tensor for a hot magnetized plasma without infinite sums of products of Bessel functions’” [Phys. Plasmas 15, 024701 (2008)]. *Physics of Plasmas*, 15:024702, 2008.

[96] Jianxian Qiu and Chi-Wang Shu. Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galkerkin method: one-dimensional case. *Journal of Computational Physics*, 193(1):115–135, Jan 2004.

[97] Jianxian Qiu and Chi-Wang Shu. Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galkerkin method II : Two-dimensional case. *Computers and Fluids*, 34(6):642–663, Jul 2005.

[98] Jianxian Qiu and Chi-Wang Shu. Runge-Kutta Discontinuous Galerkin Method Using Weno Limiters. *SIAM J. Sci. Comput.*, 26(3):907–929, 2005.

[99] Noah Reddell. *A Kinetic Vlasov Model for Plasma Simulation Using Discontinuous Galerkin Method on Many-Core Architectures.* PhD thesis, University of Washington, Seattle, WA, 2016.

[100] W. H. Reed and T. R. Hill. Triangular Mesh Methods for the Neutron Transport Equation. Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, Los Alamos, NM 87544, 1973.

[101] William J. Rider and Len G. Margolin. Simple Modifications of Monoticity-Preserving Limiter. *Journal of Computational Physics*, 174(1):473–488, Nov 2001.

[102] Beatrice Riviere. *Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation.* SIAM, Philadelphia, 2008.

[103] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.

[104] Osamu Sakai and Kunihide Tachibana. Plasmas as metamaterials: a review. *Plasma Sources Science and Technology*, 21(013001):1–18, Jan 2012.

[105] Chi-Wang Shu and Stanley Osher. Efficient Implementation of Essentially Non-oscillatory Shock-Capturing Schemes. *Journal of Computational Physics*, 77:439–471, Aug 1988.

[106] U. Shumlak and J. Loverich. Approximate Riemann solver for the two-fluid plasma model. *Journal of Computational Physics*, 187(2):620–638, 2003.

[107] U. Shumlak, B. A. Nelson, E. L. Claveau, E. G. Forbes, R. P. Golingo, M. C Hughes, R. J. Oberto, M. P. Ross, and T. R. Weber. Increasing plasma parameters using sheared flow stabilization of a Z-pinch. *Physics of Plasmas*, 24(055702):1–10, 2017.

[108] S. V. Singh, A. P. Kakad, R. V. Reddy, and G. S. Lakhina. Low-frequency instabilities due to energetic oxygen ions. *Journal of Plasma Physics*, 70(5):613–623, 2004.

[109] Eder Marinho Sousa. *A Blended Finite Element Method for Multi-Fluid Plasma Modeling*. PhD thesis, University of Washington, Seattle, WA, 2014.

[110] Bhuvana Srinivasan. *Numerical Methods for 3-dimensional Magnetic Confinement Configurations using Two-Fluid Plasma Equations*. PhD thesis, University of Washington, Seattle, WA, 2010.

[111] Loren C. Steinhauer. Review of field-reversed configurations. *Physics of Plasmas*, 18(070501):1–38, 2011.

[112] Thomas Howerd Stix. *Waves In Plasmas*. API, New York, New York, 1992.

[113] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer Science+Business Media, LLC, New York, NY, 3rd edition, 2002.

[114] A. Suresh and H. T. Huynh. Accurate Monotonicity-Preserving Schemes with Runge-Kutta Time Stepping. *Journal of Computational Physics*, 136:83–99, May 1997.

[115] Gary D Swanson. *Plasma Waves*. Institute of Physics Publishing, Bristol, UK, 2nd edition, 2003.

[116] J. A. Tataronis and F. W. Crawford. Cyclotron harmonic wave propagation and instabilities I. Perpendicular propagation. *Journal of Plasma Physics*, 4(2):231–248, 1970.

[117] Gary Taylor. Recent Developments in Electron Cyclotron Emission Research on Magnetically Confined Plasmas. *Fusion Science and Technology*, 52(2):119–133, 2007.

[118] A. V. Timofeev and V. I. Pistunovich. Cyclotron Instabilities in an Anisotropic Plasma. In M. A. Leontovich, editor, *Reviews of Plasma Physics*, chapter 3, pages 401–445. Springer, Boston, MA, 1970.

[119] Eleuterio F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer-Verlag Berlin Heidelberg, Heidelberg, Germany, 2nd edition, 1999.

[120] M. Torrilhon. Exact Solver and Uniqueness Conditions for Riemann Problems for Ideal Magnetohydrodynamics. Technical Report 2002-06, Eidgenossische Technische Hochschule, CH-8092 Zurich, Switzerland, April 2002.

[121] M. Torrilhon. Uniqueness conditions for Riemann problems of ideal magnetohydrodynamics. *Journal of Plasma Physics*, 69(part 3):253–276, 2003.

[122] Shuangzhang Tu and Shahrouz Aliabadi. A Slope Limiting Procedure in Discontinuous Galerkin Finite Element Method for Gasdynamics Applications. *International Journal of Numerical Analysis and Modeling*, 2(2):163–178, 2005.

[123] National Research Council (U.S.). *Plasma processing of materials: scientific opportunities and technological challenges.* The National Academy Press, Washington, D.C., 1991.

[124] Walter G. Vincenti and Jr. Charles H. Kruger. *Introduction to Physical Gas Dynamics.* Robert E. Krieger Publishing Company, Malabar, Florida, 1983.

[125] G. V. Vogman, P. Colella, and U. Shumlak. Dory-Guest-Harris instability as a benchmark for continuum kinetic Vlasov-Poisson simulations of magnetized plasmas. *Journal of Computational Physics*, 277:101–120, 2014.

[126] G. V. Vogman, J. H. Hammer, and W. A. Farmer. Customizable two-species kinetic equilibria for nonuniform low-beta plasmas. *Physics of Plasmas*, 26:042119, 2019.

[127] G. V. Vogman, J. H. Hammer, U. Shumlak, and W. A. Farmer. Two-fluid and kinetic transport physics of Kelvin–Helmholtz instabilities in nonuniform low-beta plasmas. *Physics of Plasmas*, 27:102109, 2020.

[128] Genia Vogman. *Fourth-order conservative Vlasov-Maxwell solver for Cartesian and cylindrical phase space coordinates.* PhD thesis, University of California, Berkeley, Berekely, CA, 2016.

[129] J. VonNeumann and R. D. Richtmyer. A amethod for the Numerical Calculation of Hydrodynamics Shocks. *Journal of Applied Physics*, 21(232):232–237, Mar 1950.

[130] B. Wang and M. A. Capelli. A plasma photonic crystal bandgap device. *Applied Physics Letters*, 108(161101):1–4, 2016.

[131] M.Y. Wang and G.H. Miley. Particle orbits in field-reversed mirrors. *Nuclear Fusion*, 19(1):39–49, 1979.

[132] Zhi Jian Wang. *Adapative High-Order Methods in Compuatational Fluid Dynamics*, volume 2 of *Advances in Computational Fluid Dyanamics Series*. World Scientific Publishing Co Pte Ltd, Singapore, 2011.

[133] Erich S. Weibel. Spontaneously Growing Transverse Waves In A Plasma Due to An Anisotropic Velocity Distribution. *Physical Review Letters*, 2(3):83–84, 1959.

[134] Jian Yu and Chao Yan. An artificial diffusivity discontinuous Galerkin scheme for discontinuous flows. *Computers & Fluids*, 75:56–71, Feb 2013.

[135] Xinghui Zhong and Chi-Wang Shu. A simple weighted essentially nonoscillatory limiter for Runge-Kutta discontinuous Galerkin methods. *Journal of Computational Physics*, 232:397–415, 2013.

[136] Jun Zhu and Jianxian Qiu. Hermite WENO Schemes and Their Application as Limiters for Runge-Kutta Discontinuous Galerkin Method, III: Unstructered Meshes. *Journal of Scientific Computing*, 39(2):293–321, May 2009.

[137] Jun Zhu and Jianxian Qiu. Runge-Kutta Discontinuous Galerkin Method Using Weno-Type Limiters: Three-Dimensional Unstructured Meshes. *Communications in Computational Physics*, 11(3):985–1005, March 2012.

[138] Jun Zhu and Jianxian Qiu. WENO Schemes and Their Applications as Limiters for RKDG Methods Based on Trigonometric Approximation Spaces. *Journal of Scientific Computing*, 55(3):606–644, Sep 2012.

[139] Jun Zhu, Jianxian Qiu, Chi-Wang Shu, and Michael Dumbser. Runge-Kutta discontinuous Galerkin method using WENO limiters II: Unstructured meshes. *Journal of Computational Physics*, 227(9):4330–4353, Apr 2008.

[140] Jun Zhu, Xinghui Zhong, Chi-Wang Shu, and Jianxian Qiu. Runge-Kutta discontinuous Galerkin method using a new type of WENO limiters on unstructured meshes. *Journal of Computational Physics*, 248:200–220, Sep 2013.

Appendix A

COLLISIONAL NORMALIZATIONS

A.1 Collision Frequency

According to Hinton equation 47 [56], momentum transfer time between species α and β is

$$\nu_{\alpha\beta} = \tau_{\alpha\beta}^{-1} = \frac{4}{3\pi^{\frac{1}{2}}} \frac{n_\beta Z_\beta^2 \frac{4\pi Z_\alpha^2 q_0^4 \ln\Lambda}{m_\alpha^2}\left(1+\frac{m_\alpha}{m_\beta}\right)}{\left(2\frac{T_\alpha}{m_\alpha}+2\frac{T_\beta}{m_\beta}\right)^{\frac{3}{2}}},$$

which is written in Gaussian units and $q_0 = +e$. This can be converted to SI units by multiplying by $\frac{1}{(4\pi\epsilon_0)^2}$ [58], such that it can be written

$$\nu_{\alpha\beta} = \tau_{\alpha\beta}^{-1} = \frac{1}{3\pi^{\frac{3}{2}}\epsilon_0^2} \frac{n_\beta Z_\beta^2 Z_\alpha^2 q_0^4 \ln\Lambda \left(1+\frac{m_\alpha}{m_\beta}\right)}{m_\alpha^2\left(2\frac{T_\alpha}{m_\alpha}+2\frac{T_\beta}{m_\beta}\right)^{\frac{3}{2}}}. \tag{A.1.1}$$

This can then be non-dimensionalized as follows

$$\begin{aligned}
\frac{\tilde{\nu}_{\alpha\beta}}{\tau} &= \frac{1}{3\pi^{\frac{3}{2}}\epsilon_0^2} \frac{n_0 q_0^4}{m_0^2 v_0^3} \frac{\tilde{n}_\beta Z_\beta^2 Z_\alpha^2 \ln\Lambda\left(1+\frac{A_\alpha}{A_\beta}\right)}{2^{\frac{3}{2}} A_\alpha^2 \left(\frac{\tilde{T}_\alpha}{A_\alpha}+\frac{\tilde{T}_\beta}{A_\beta}\right)^{\frac{3}{2}}} \\
&= \frac{1}{12\pi^{\frac{3}{2}} n_0} \frac{n_0^2 q_0^4}{m_0^2 \epsilon_0^2} \frac{\tau^3}{L^3} 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln\Lambda \frac{\left(1+\frac{A_\alpha}{A_\beta}\right)}{A_\alpha^2} \frac{\tilde{n}_\beta}{\left(\frac{\tilde{T}_\alpha}{A_\alpha}+\frac{\tilde{T}_\beta}{A_\beta}\right)^{\frac{3}{2}}} \\
&= \frac{(\omega_p\tau)^4}{12\pi^{\frac{3}{2}} n_0 L^3 \tau} 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln\Lambda \frac{(A_\alpha+A_\beta)}{A_\alpha^2 A_\beta^2} \frac{\tilde{\rho}_\beta}{\left(\frac{\tilde{T}_\alpha}{A_\alpha}+\frac{\tilde{T}_\beta}{A_\beta}\right)^{\frac{3}{2}}} \\
\tilde{\nu}_{\alpha\beta} &= \frac{(\omega_p\tau)^4}{12\pi^{\frac{3}{2}} n_0 L^3 \tau} \tau 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln\Lambda \frac{(A_\alpha+A_\beta)}{A_\alpha^2 A_\beta^2} \frac{\tilde{\rho}_\beta}{\left(\frac{\tilde{T}_\alpha}{A_\alpha}+\frac{\tilde{T}_\beta}{A_\beta}\right)^{\frac{3}{2}}},
\end{aligned}$$

where

$$\omega_p = \left(\frac{q_0^2 n_0}{m_0 \epsilon_0}\right)^{\frac{1}{2}}.$$

Then defining

$$\nu_p = \frac{(\omega_p \tau)^4}{12\pi^{\frac{3}{2}} n_0 L^3 \tau} = \frac{q_0^4}{12\pi^{\frac{3}{2}} \epsilon_0^2} \frac{n_0}{m_0^{\frac{1}{2}} T_0^{\frac{3}{2}}} \tag{A.1.2}$$

since $T_0 = m_0 v_0^2$, leads to

$$\tilde{\nu}_{\alpha\beta} = (\nu_p \tau)\, 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln\Lambda \frac{(A_\alpha + A_\beta)}{A_\alpha^2 A_\beta^2} \frac{\tilde{\rho}_\beta}{\left(\frac{\tilde{T}_\alpha}{A_\alpha} + \frac{\tilde{T}_\beta}{A_\beta}\right)^{\frac{3}{2}}}. \tag{A.1.3}$$

A.2 Resistivity

The resistivity term in Ohm's Law is

$$E_i = \frac{1}{n_\mathrm{e} e} R_{\mathrm{ei}_i} = \frac{1}{n_\mathrm{e} e} m_e n_e \left(v_{\mathrm{i}_i} - v_{\mathrm{e}_i}\right) \nu_\mathrm{ei} = \eta j_i.$$

Upon normalization, this becomes

$$\begin{aligned} \frac{1}{n_0 \tilde{n}_\mathrm{e} e} m_p A_e n_0 \tilde{n}_\mathrm{e} v_0 \left(\tilde{v}_{\mathrm{i}_i} - \tilde{v}_{\mathrm{e}_i}\right) \frac{1}{\tau} \tilde{\nu}_\mathrm{ei} =& \eta_0 \tilde{\eta} j_0 \tilde{j}_i \\ =& \frac{m_p \nu_p}{n_0 e^2} \tilde{\eta} e n_0 v_0 \tilde{j}_i \\ A_e \left(\tilde{v}_{\mathrm{i}_i} - \tilde{v}_{\mathrm{e}_i}\right) \frac{1}{\tau} \tilde{\nu}_\mathrm{ei} =& \nu_p \tilde{\eta} \tilde{j}_i. \end{aligned}$$

Now substitute Eq. (A.1.3)

$$\begin{aligned} A_e \left(\tilde{v}_{\mathrm{i}_i} - \tilde{v}_{\mathrm{e}_i}\right) \frac{1}{\tau} (\nu_p \tau)\, 2^{\frac{1}{2}} Z_\mathrm{i}^2 Z_\mathrm{e}^2 \ln\Lambda \frac{A_\mathrm{e} + A_\mathrm{i}}{A_\mathrm{e}^2 A_\mathrm{i}^2} \frac{\tilde{\rho}_\mathrm{i}}{\left(\frac{\tilde{T}_\mathrm{e}}{A_\mathrm{e}} + \frac{\tilde{T}_\mathrm{i}}{A_\mathrm{i}}\right)^{\frac{3}{2}}} =& \nu_p \tilde{\eta} \tilde{j}_i \\ A_e \left(\tilde{v}_{\mathrm{i}_i} - \tilde{v}_{\mathrm{e}_i}\right) 2^{\frac{1}{2}} Z_\mathrm{i}^2 Z_\mathrm{e}^2 \ln\Lambda \frac{A_\mathrm{e} + A_\mathrm{i}}{A_\mathrm{e}^2 A_\mathrm{i}^2} \frac{A_\mathrm{i} \tilde{n}_\mathrm{i}}{\left(\frac{\tilde{T}_\mathrm{e}}{A_\mathrm{e}} + \frac{\tilde{T}_\mathrm{i}}{A_\mathrm{i}}\right)^{\frac{3}{2}}} =& \tilde{\eta} \tilde{j}_i. \end{aligned}$$

Note by quasineutrality $\tilde{n}_\mathrm{i} = \frac{-Z_\mathrm{e}}{Z_\mathrm{i}} \tilde{n}_\mathrm{e}$ and specifically for two-fluid

$$j_i = q_\mathrm{i} n_\mathrm{i} v_{\mathrm{i}_i} + q_\mathrm{e} n_\mathrm{e} v_{\mathrm{e}_i} = e n_0 v_0 \left(Z_\mathrm{i} \tilde{n}_\mathrm{i} \tilde{v}_{\mathrm{i}_i} + Z_\mathrm{e} \tilde{n}_\mathrm{e} \tilde{v}_{\mathrm{e}_i}\right) = e n_0 v_0 \tilde{n}_\mathrm{e} Z_\mathrm{e} \left(\tilde{v}_{\mathrm{e}_i} - \tilde{v}_{\mathrm{i}_i}\right).$$

Substituting this in

$$\begin{aligned} A_e\left(\tilde{v}_{\mathrm{i}_i}-\tilde{v}_{\mathrm{e}_i}\right)2^{\frac{1}{2}}Z_{\mathrm{i}}^2Z_{\mathrm{e}}^2\ln\Lambda\frac{A_{\mathrm{e}}+A_{\mathrm{i}}}{A_{\mathrm{e}}^2A_{\mathrm{i}}^2}\frac{A_{\mathrm{i}}\frac{-Z_{\mathrm{e}}}{Z_{\mathrm{i}}}\tilde{n}_{\mathrm{e}}}{\left(\frac{\tilde{T}_{\mathrm{e}}}{A_{\mathrm{e}}}+\frac{\tilde{T}_{\mathrm{i}}}{A_{\mathrm{i}}}\right)^{\frac{3}{2}}} =& \tilde{\eta}\tilde{n}_{\mathrm{e}}Z_{\mathrm{e}}\left(\tilde{v}_{\mathrm{e}_i}-\tilde{v}_{\mathrm{i}_i}\right) \\ \tilde{\eta} =& A_{\mathrm{e}}2^{\frac{1}{2}}Z_{\mathrm{i}}Z_{\mathrm{e}}^2\ln\Lambda\frac{A_{\mathrm{e}}+A_{\mathrm{i}}}{A_{\mathrm{e}}^2A_{\mathrm{i}}^2}\frac{A_{\mathrm{i}}}{\left(\frac{\tilde{T}_{\mathrm{e}}}{A_{\mathrm{e}}}+\frac{\tilde{T}_{\mathrm{i}}}{A_{\mathrm{i}}}\right)^{\frac{3}{2}}}. \end{aligned}$$

Now if the assumption of $A_{\mathrm{e}} << A_{\mathrm{i}}$ is made

$$\tilde{\eta} = 2^{\frac{1}{2}}Z_{\mathrm{i}}Z_{\mathrm{e}}^2\ln\Lambda\frac{1}{A_{\mathrm{e}}}\frac{1}{\left(\frac{\tilde{T}_{\mathrm{e}}}{A_{\mathrm{e}}}+\frac{\tilde{T}_{\mathrm{i}}}{A_{\mathrm{i}}}\right)^{\frac{3}{2}}}.$$

If also $T_{\mathrm{i}} \approx T_{\mathrm{e}}$

$$\begin{aligned} \tilde{\eta} =& 2^{\frac{1}{2}}Z_{\mathrm{i}}Z_{\mathrm{e}}^2\ln\Lambda\frac{1}{A_{\mathrm{e}}}\frac{1}{\left(\frac{\tilde{T}_{\mathrm{e}}}{A_{\mathrm{e}}}\right)^{\frac{3}{2}}} \\ \tilde{\eta} =& 2^{\frac{1}{2}}Z_{\mathrm{i}}Z_{\mathrm{e}}^2\ln\Lambda\frac{A_{\mathrm{e}}^{\frac{1}{2}}}{\tilde{T}_{\mathrm{e}}^{\frac{3}{2}}}. \end{aligned}$$

Assuming $Z_{\mathrm{e}} = -1$, then

$$\tilde{\eta} = 2^{\frac{1}{2}}Z_{\mathrm{i}}\ln\Lambda\frac{A_{\mathrm{e}}^{\frac{1}{2}}}{\tilde{T}_{\mathrm{e}}^{\frac{3}{2}}}. \tag{A.2.1}$$

Relating back to the Ohm's Law term

$$\begin{aligned} E_i =& \eta j_i \\ v_0B_0\tilde{E}_i =& \frac{m_p\nu_p}{n_0e^2}\tilde{\eta}en_0v_0\tilde{j}_i \\ \tilde{E}_i =& \frac{m_p}{B_0e}\nu_p\tilde{\eta}\tilde{j}_i \\ \tilde{E}_i =& \frac{1}{\omega_c\tau}\left(\nu_p\tau\right)\tilde{\eta}\tilde{j}_i \\ \tilde{E}_i =& \left(\frac{\delta_p}{L}\right)\left(\nu_p\tau\right)\tilde{\eta}\tilde{j}_i. \end{aligned} \tag{A.2.2}$$

Appendix B

MODEL VARIABLE CALCULATIONS

B.1 Two-Fluid and MHD Component Calculations

Energy: For two-fluid,

$$\begin{aligned} e_{\mathrm{i}} =& \frac{p_{\mathrm{i}}}{\gamma - 1} + \frac{1}{2}\rho_{\mathrm{i}} v_{\mathrm{i}}^2 \\ m_p n_0 v_0^2 \tilde{e}_{\mathrm{i}} =& \frac{m_p n_0 v_0^2 \tilde{p}_{\mathrm{i}}}{\gamma - 1} + \frac{1}{2} m_p n_0 \tilde{\rho}_{\mathrm{i}} v_0^2 \tilde{v}_{\mathrm{i}}^2 \\ \tilde{e}_{\mathrm{i}} =& \frac{\tilde{p}_{\mathrm{i}}}{\gamma - 1} + \frac{1}{2}\tilde{\rho}_{\mathrm{i}} \tilde{v}_{\mathrm{i}}^2. \end{aligned} \tag{B.1.1}$$

Similarly

$$\tilde{e}_{\mathrm{e}} = \frac{\tilde{p}_{\mathrm{e}}}{\gamma - 1} + \frac{1}{2}\tilde{\rho}_{\mathrm{e}} \tilde{v}_{\mathrm{e}}^2. \tag{B.1.2}$$

Note that in the MHD approximation $\tilde{\rho}_{\mathrm{e}} \rightarrow 0$ so

$$\tilde{e}_{\mathrm{e}} = \frac{\tilde{p}_e}{\gamma - 1}. \tag{B.1.3}$$

The total electron energy for MHD is thus

$$\tilde{e}_{t_{\mathrm{e}}} = \tilde{e}_{\mathrm{e}} + \frac{\tilde{B}^2}{2}. \tag{B.1.4}$$

What are n_e and p_e?

- n_e Apply mass density definition, charge conservation, and number density definitions (normalized terms will be explicitely tilded):

$$\begin{aligned} \rho =& m_i n_i + m_e n_e \\ =& A_i m_p n_i + A_e m_p n_e \end{aligned} \tag{B.1.5}$$

$$\begin{aligned} q_i n_i + q_e n_e =& 0 \\ z_i e n_i + z_e e n_e =& 0, \end{aligned} \tag{B.1.6}$$

$$n = n_i + n_e, \tag{B.1.7}$$

where $A_\alpha = \frac{m_\alpha}{m_p}$ and $z_\alpha = \frac{q_\alpha}{e}$. These are 3 equations with 3 unknowns (n, n_e, n_i). Thus from Eq. (B.1.6):

$$n_i = -\frac{z_e}{z_i} n_e. \tag{B.1.8}$$

From Eq. (B.1.5):

$$\begin{aligned} n_e =& \frac{\rho - A_i m_p n_i}{A_e m_p} \\ =& \frac{\rho + A_i m_p \frac{z_e}{z_i} n_e}{A_e m_p} \\ \left(1 - \frac{A_i m_p \frac{z_e}{z_i}}{A_e m_p}\right) n_e =& \frac{\rho}{A_e m_p} \\ n_e =& \frac{\rho / m_p}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \\ n_e =& \frac{\tilde{\rho} n_0}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \\ \tilde{n}_e =& \frac{\tilde{\rho}}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \\ \tilde{n}_e =& \frac{\tilde{\rho}}{\left(A_e - A_i \frac{z_e}{z_i}\right)}. \end{aligned} \tag{B.1.9}$$

$$\begin{aligned} n_i =& -\frac{z_e}{z_i} n_e \\ n_i =& -\frac{z_e}{z_i} \frac{\tilde{\rho} n_0}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \\ \tilde{n}_i =& -\frac{z_e}{z_i} \frac{\tilde{\rho}}{\left(A_e - A_i \frac{z_e}{z_i}\right)}. \end{aligned} \tag{B.1.10}$$

$$\begin{aligned}
n =& n_e + n_i \\
n =& \frac{\tilde{\rho} n_0}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} - \frac{z_e}{z_i} \frac{\tilde{\rho} n_0}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \\
\tilde{n} =& \left[1 - \frac{z_e}{z_i}\right] \frac{\tilde{\rho}}{\left(A_e - A_i \frac{z_e}{z_i}\right)} \\
\tilde{n} =& \left[\frac{z_i - z_e}{z_i}\right] \frac{\tilde{\rho}}{\frac{A_e z_i - A_i z_e}{z_i}} \\
\tilde{n} =& \left(\frac{z_i - z_e}{A_e z_i - A_i z_e}\right) \tilde{\rho}.
\end{aligned} \tag{B.1.11}$$

- p_e

$$\begin{aligned}
p =& p_e + p_i \\
=& n_e T_e + n_i T_i \\
=& \frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)} T_e + \left(-\frac{z_e}{z_i} \frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) T_i \\
=& \left(\frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) \left(T_e - \frac{z_e}{z_i} T_i\right) \\
& \text{let } T_e = \theta T_i \\
=& \left(\frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) \left(\theta - \frac{z_e}{z_i}\right) T_i \\
T_i =& \frac{p}{\left(\frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) \left(\theta - \frac{z_e}{z_i}\right)} \\
\frac{T_i}{m_p v_0^2} =& \frac{\frac{p}{m_p v_0^2}}{\left(\frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) \left(\theta - \frac{z_e}{z_i}\right)} \\
\tilde{T}_i =& \frac{\tilde{p}}{\left(\frac{\tilde{\rho}}{\left(A_e - A_i \frac{z_e}{z_i}\right)}\right) \left(\theta - \frac{z_e}{z_i}\right)},
\end{aligned} \tag{B.1.12}$$

where $\tilde{T}_i = \frac{T_i}{m_p v_0^2}$. So now

$$
\begin{aligned}
p_e =& p - p_i \\
=& p - n_i T_i \\
=& p - \left(- \frac{z_e}{z_i} \frac{\tilde{\rho} n_0}{A_e} \frac{1}{\left(1 - \frac{A_i z_e}{A_e z_i}\right)} \right) \frac{p}{\left(\frac{\tilde{\rho} n_0}{\left(A_e - A_i \frac{z_e}{z_i}\right)} \right) \left(\theta - \frac{z_e}{z_i}\right)} \\
=& p + \left(\frac{z_e}{z_i} \frac{\tilde{\rho} n_0}{1} \frac{1}{\left(A_e - A_i \frac{z_e}{z_i}\right)} \right) \left(\frac{\left(A_e - A_i \frac{z_e}{z_i}\right)}{\tilde{\rho} n_0} \right) \frac{p}{\left(\theta - \frac{z_e}{z_i}\right)} \\
=& p + \left(\frac{z_e}{z_i} \right) \frac{p}{\left(\theta - \frac{z_e}{z_i}\right)} \\
=& \left[1 + \frac{\left(\frac{z_e}{z_i}\right)}{\left(\theta - \frac{z_e}{z_i}\right)} \right] p \\
\tilde{p}_e =& \left[1 + \frac{\left(\frac{z_e}{z_i}\right)}{\left(\theta - \frac{z_e}{z_i}\right)} \right] \tilde{p} \\
\tilde{p}_e =& \left[1 + \frac{1}{\left(\frac{z_i}{z_e}\theta - 1\right)} \right] \tilde{p}.
\end{aligned}
\tag{B.1.13}
$$

B.2 Gradients of p

Assume everything is already normalized.

$$
\begin{aligned}
p =& (\gamma - 1) \left[e - \frac{1}{2} \rho v^2 - \frac{1}{2} B^2 \right] \\
=& (\gamma - 1) \left[q_4 - \frac{1}{2} \frac{q_1^2 + q_2^2 + q_3^2}{q_0} - \frac{1}{2} \left(q_5^2 + q_6^2 + q_7^2 \right) \right],
\end{aligned}
\tag{B.2.1}
$$

$$\begin{aligned}
\nabla p =& \partial_i \left[(\gamma - 1) \left[q_4 - \frac{1}{2} \frac{q_1^2 + q_2^2 + q_3^2}{q_0} - \frac{1}{2} \left(q_5^2 + q_6^2 + q_7^2 \right) \right] \right] \\
=& (\gamma - 1) \left[\partial_i q_4 - \frac{1}{2} \left(\frac{\partial_i \left(q_1^2 + q_2^2 + q_3^2 \right) q_0 - \partial_i \left(q_0 \right) \left(q_1^2 + q_2^2 + q_3^2 \right)}{q_0^2} \right) - \frac{1}{2} \partial_i \left(q_5^2 + q_6^2 + q_7^2 \right) \right] \\
=& (\gamma - 1) \left[\partial_i q_4 - \frac{1}{2} \left(\frac{2 \left(q_1 \partial_i q_1 + q_2 \partial_i q_2 + q_3 \partial_i q_3 \right) q_0 - \partial_i \left(q_0 \right) \left(q_1^2 + q_2^2 + q_3^2 \right)}{q_0^2} \right) - \left(q_5 \partial_i q_5 + q_6 \partial_i q_6 + q_7 \partial_i q_7 \right) \right] \\
=& (\gamma - 1) \left[\partial_i q_4 - \frac{\left(q_1 \partial_i q_1 + q_2 \partial_i q_2 + q_3 \partial_i q_3 \right)}{q_0} + \frac{1}{2} \left(\frac{\partial_i \left(q_0 \right) \left(q_1^2 + q_2^2 + q_3^2 \right)}{q_0^2} \right) - \left(q_5 \partial_i q_5 + q_6 \partial_i q_6 + q_7 \partial_i q_7 \right) \right].
\end{aligned} \tag{B.2.2}$$

Appendix C

HALL MHD HYPERVISCOSITY

To stabilize the Whistler waves the the Hall model, a hyerviscosity is added to Ohm's Law, which appends the Faraday and energy Eqs. (2.3.29) and (2.3.28), respectively. The relevant term of the Ohm's law is given as

$$E_i = -\nu \frac{\partial^2 j_i}{\partial x_j \partial x_j} \tag{C.0.1}$$

for a user-specified typically spatially-constant ν. The construction of j_i is formulated using the local discontinuous Galerkin or interior penalty methods as described in Sec. 3.2. Once this is constructed, this is directly added to Faraday's law as expressed in the general form in Eq. (2.3.4). As for the energy equation, notice the form of the equation given by combining Eqs. (2.2.57) and (2.2.59), normalized, yields a form

$$\frac{\partial}{\partial t}\left(e + \frac{B^2}{2}\right) + \frac{\partial\left(e v_i + P_{ij} v_j + h_i + \left(\frac{\gamma p_{\mathrm{e}}}{\gamma - 1} w_{\mathrm{e}_i} + \Pi_{\mathrm{e}_{ij}} w_{\mathrm{e}_j}\right)\right)}{\partial x_i} = -\frac{\partial}{\partial x_i}\left(\epsilon_{ijk} E_j B_k\right), \tag{C.0.2}$$

where this electric field term can be substituted into the flux in this way.

The construction of this term, however, requires two additional derivatives upon j_i. The gradient method could be progressively applied (Sec. 3.2) on these terms, though this would require many more components as well as boundary condition handling. Instead, the approach as described by Srinivasan [110] is employed here. Namely, this is to couch Eq. (C.0.1) in the weak form and directly calculate the right hand side. To do this, notice this equation can be rewritten as

$$E_i + \frac{\partial F_{ij}}{\partial x_j} = 0, \tag{C.0.3}$$

where

$$F_{ij} = \nu H_{ij} \tag{C.0.4}$$

and

$$H_{ij} = \frac{\partial j_i}{\partial x_j}. \tag{C.0.5}$$

A few observations can be made here. First, Eq. (C.0.3) is not different from the conservation law in Eq. (3.0.1) (for only flux terms) except the time derivative of the conservation variable is replaced by E_i, and that Eq. (C.0.5) is just the gradient definition. In Chapter 3, these equations are converted to the DG form. Here, a similar approach is applied on both equations, except there is no integration by parts to split the integral into volume and surface terms. Instead the whole volume integral is calculated. Starting with the gradient term in Eq. (C.0.5), following the derivation in Chapter 3, it can be written for element λ

$$H^{\lambda}_{ij} = {J^{\lambda}_{kj}}^{-1} \frac{\partial j^{\lambda}_i}{\partial \xi^{\lambda}_k}.$$

This can be turned into the familiar weak form

$$\int_{D^{\lambda}} H^{\lambda}_{ij} \psi_n d\boldsymbol{\xi} = \int_{D^{\lambda}} {J^{\lambda}_{kj}}^{-1} \frac{\partial j^{\lambda}_i}{\partial \xi^{\lambda}_k} \psi_n d\boldsymbol{\xi}.$$

In this case do not apply the product and divergence rules (integration by parts), instead immediately expand on bases (assume ψ)

$$\int_{D^{\lambda}} \hat{H}^{\lambda}_{ijl} \psi_l \psi_n d\boldsymbol{\xi} = \int_{D^{\lambda}} {J^{\lambda}_{kj}}^{-1} \frac{\partial \hat{\jmath}^{\lambda}_{il} \psi_l}{\partial \xi^{\lambda}_k} \psi_n d\boldsymbol{\xi}.$$

Rewriting

$$\hat{H}^{\lambda}_{ijl} \int_{D^{\lambda}} \psi_n \psi_l d\boldsymbol{\xi} = {J^{\lambda}_{kj}}^{-1} \hat{\jmath}^{\lambda}_{il} \int_{D^{\lambda}} \psi_n \frac{\partial \psi_l}{\partial \xi^{\lambda}_k} d\boldsymbol{\xi}$$

or

$$\hat{H}^{\lambda}_{ijl} \mathcal{M}_{nl} = {J^{\lambda}_{kj}}^{-1} \hat{\jmath}^{\lambda}_{il} \mathcal{A}_{nkl}.$$

Multiplying by $\mathcal{M}^{-1}$

$$\begin{aligned} \hat{H}^{\lambda}_{ijl} \mathcal{M}^{-1}_{mn} \mathcal{M}_{nl} &= {J^{\lambda}_{kj}}^{-1} \hat{\jmath}^{\lambda}_{il} \mathcal{M}^{-1}_{mn} \mathcal{A}_{nkl} \\ \hat{H}^{\lambda}_{ijm} &= {J^{\lambda}_{kj}}^{-1} \hat{\jmath}^{\lambda}_{il} \mathcal{D}_{r_{mkl}}. \end{aligned}$$

Swap m and l

$$\hat{H}^{\lambda}_{ijl} = J^{\lambda}_{kj}{}^{-1}\hat{\jmath}^{\lambda}_{im}\mathcal{D}_{r_{lkm}}.$$

Swap l and k

$$\hat{H}^{\lambda}_{ijk} = J^{\lambda}_{lj}{}^{-1}\hat{\jmath}^{\lambda}_{im}\mathcal{D}_{r_{klm}}. \tag{C.0.6}$$

Note this is simply Eq. (3.2.43). For Eq. (C.0.3), a similar procedure to Chapter 3 is applied but again without the integration by parts

$$\begin{aligned}
E^{\lambda}_{i} + J^{\lambda}_{kj}{}^{-1}\frac{\partial \mathcal{F}^{\lambda}_{ij}}{\partial \xi^{\lambda}_{k}} &=0\\
\int_{D^{\lambda}} E^{\lambda}_{i}\psi_m d\boldsymbol{\xi} + \int_{D^{\lambda}} J^{\lambda}_{kj}{}^{-1}\frac{\partial \mathcal{F}^{\lambda}_{ij}}{\partial \xi^{\lambda}_{k}}\psi_m d\boldsymbol{\xi} &=0\\
\int_{D^{\lambda}} \hat{E}^{\lambda}_{in}\psi_n\psi_m d\boldsymbol{\xi} + \int_{D^{\lambda}} J^{\lambda}_{kj}{}^{-1}\frac{\partial \hat{\mathcal{F}}^{\lambda}_{ijn}\psi_n}{\partial \xi^{\lambda}_{k}}\psi_m d\boldsymbol{\xi} &=0\\
\hat{E}^{\lambda}_{in}\int_{D^{\lambda}} \psi_n\psi_m d\boldsymbol{\xi} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\int_{D^{\lambda}} \frac{\partial \psi_n}{\partial \xi^{\lambda}_{k}}\psi_m d\boldsymbol{\xi} &=0\\
\hat{E}^{\lambda}_{in}\int_{D^{\lambda}} \psi_m\psi_n d\boldsymbol{\xi} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\int_{D^{\lambda}} \psi_m\frac{\partial \psi_n}{\partial \xi^{\lambda}_{k}} d\boldsymbol{\xi} &=0\\
\hat{E}^{\lambda}_{in}\mathcal{M}_{mn} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\mathcal{A}_{mkn} &=0\\
\hat{E}^{\lambda}_{in}\mathcal{M}^{-1}_{lm}\mathcal{M}_{mn} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\mathcal{M}^{-1}_{lm}\mathcal{A}_{mkn} &=0\\
\hat{E}^{\lambda}_{il} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\mathcal{M}^{-1}_{lm}\mathcal{A}_{mkn} &=0\\
\hat{E}^{\lambda}_{il} + J^{\lambda}_{kj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\mathcal{D}_{r_{lkn}} &=0.
\end{aligned}$$

Swap l and k

$$\hat{E}^{\lambda}_{ik} + J^{\lambda}_{lj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijn}\mathcal{D}_{r_{kln}} =0.$$

Switch n to m

$$\hat{E}^{\lambda}_{ik} + J^{\lambda}_{lj}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ijm}\mathcal{D}_{r_{klm}} =0.$$

Swap k and j

$$\hat{E}^{\lambda}_{ij} + J^{\lambda}_{lk}{}^{-1}\hat{\mathcal{F}}^{\lambda}_{ikm}\mathcal{D}_{r_{jlm}} =0.$$

Swap m and k

$$\hat{E}_{ij}^{\lambda} + {J_{lm}^{\lambda}}^{-1} \hat{\mathcal{F}}_{imk}^{\lambda} \mathcal{D}_{r_{jlk}} = 0.$$

So finally

$$\hat{E}_{ij}^{\lambda} = - {J_{lm}^{\lambda}}^{-1} \mathcal{D}_{r_{jlk}} \hat{\mathcal{F}}_{imk}^{\lambda}. \tag{C.0.7}$$

Comparing to the volume integral term in Eq. (3.1.37) it can be seen to be similar except for the negative and the use of the $\mathcal{D}_r$ matrix.

Appendix D

1D1V ANALYTIC MAXWELLIAN INTEGRALS

If the subdomain interface is in a region where the kinetic distribution function is Maxwellian, the composite distribution function fluxes could be calculated analytically, allowing for a method of diagnosis of simulation results when using the composite distribution function method described in Sec. 3.7.3. For one velocity dimension, the general Maxwellian as written in Eq. (2.3.17) becomes

$$f_\alpha(v_x) = \frac{n_\alpha}{\left(2\pi v_{\alpha_{\mathrm{th}}}^2\right)^{\frac{1}{2}}} \exp\left[\frac{-\left(v_x - v_{\alpha_x}\right)^2}{2v_{\alpha_{\mathrm{th}}}^2}\right], \tag{D.0.1}$$

where

$$v_{\alpha_{\mathrm{th}}} \equiv \left(\frac{T_\alpha}{A_\alpha}\right)^{\frac{1}{2}}. \tag{D.0.2}$$

Then the 0^{th} moment can be determined analytically as the indefinite integral

$$\int f_\alpha dv_x = \frac{n_\alpha}{2} \mathrm{erf}\left(\frac{v_x - v_{\alpha_x}}{\sqrt{2} v_{\alpha\mathrm{th}}}\right), \tag{D.0.3}$$

where $\mathrm{erf}(\infty) = 1$, $\mathrm{erf}(-\infty) = -1$, $\mathrm{erf}(0) = 0$, and also $\mathrm{erf}(-z) = -\mathrm{erf}(z)$. If this is evaluated as a definite integral over all velocity space, the density is recovered, as expected

$$\int_{-\infty}^{\infty} f_\alpha dv_x = n_\alpha. \tag{D.0.4}$$

But evaluating on the side of negative velocity space achieves

$$\int_{-\infty}^{0} f_\alpha dv_x = \frac{n_\alpha}{2} \left[1 - \mathrm{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2} v_{\alpha\mathrm{th}}}\right)\right] \tag{D.0.5}$$

and for positive velocity space achieves

$$\int\limits_{0}^{\infty} f_\alpha dv_x = \frac{n_\alpha}{2}\left[1+\mathrm{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\mathrm{th}}}\right)\right]. \tag{D.0.6}$$

Of course it is the fluxes that are needed. The 1st moment is the flux of the continuity equation, as shown in Eq. (3.7.3), which can also be calculated in the limit of a Maxwellian. Performing the indefinite integral in Eq. (3.7.3) in 1V for the Maxwellian,

$$\int v_x f_\alpha dv_x = n_\alpha\left[\frac{v_{\alpha_x}}{2}\mathrm{erf}\left(\frac{v_x - v_{\alpha_x}}{\sqrt{2}v_{\alpha\mathrm{th}}}\right) - \frac{v_{\alpha\mathrm{th}}}{\sqrt{2\pi}}\exp\left[\frac{-\left(v_x - v_{\alpha_x}\right)^2}{2v_{\alpha\mathrm{th}}^2}\right]\right]. \tag{D.0.7}$$

Then the definite integral over all velocity space is

$$\int\limits_{-\infty}^{\infty} v_x f_\alpha dv_x = n_\alpha v_{\alpha_x}. \tag{D.0.8}$$

The negative velocity space integral is

$$\int\limits_{-\infty}^{0} v_x f_\alpha dv_x = n_\alpha\left[\frac{v_{\alpha_x}}{2}\left\{1-\mathrm{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\mathrm{th}}}\right)\right\} - \frac{v_{\alpha\mathrm{th}}}{\sqrt{2\pi}}\exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\mathrm{th}}^2}\right]\right] \tag{D.0.9}$$

and the positive velocity space integral is

$$\int\limits_{0}^{\infty} v_x f_\alpha dv_x = n_\alpha\left[\frac{v_{\alpha_x}}{2}\left\{1+\mathrm{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\mathrm{th}}}\right)\right\} + \frac{v_{\alpha\mathrm{th}}}{\sqrt{2\pi}}\exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\mathrm{th}}^2}\right]\right]. \tag{D.0.10}$$

The 2nd dyadic moment yields the flux of the momentum equation as shown in Eq. (3.7.4). In 1V, there is only the v_x dimension so the indefinite integral is

$$\int v_x^2 f_\alpha dv_x = n_\alpha\left[\frac{v_{\alpha\mathrm{th}}^2 + v_{\alpha_x}^2}{2}\mathrm{erf}\left(\frac{v_x - v_{\alpha_x}}{\sqrt{2}v_{\alpha\mathrm{th}}}\right) - \frac{v_{\alpha\mathrm{th}}\left(v_x + v_{\alpha_x}\right)}{\sqrt{2\pi}}\exp\left[\frac{-\left(v_x - v_{\alpha_x}\right)^2}{2v_{\alpha\mathrm{th}^2}}\right]\right]. \tag{D.0.11}$$

This has the definite integrals

$$\int\limits_{-\infty}^{\infty} v_x^2 f_\alpha dv_x = n_\alpha\left(v_{\alpha\mathrm{th}}^2 + v_{\alpha_x}^2\right), \tag{D.0.12}$$

$$\int_{-\infty}^{0} v_x^2 f_\alpha dv_x = n_\alpha \left[\frac{v_{\alpha\text{th}}^2 + v_{\alpha_x}^2}{2}\left\{1 - \text{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\text{th}}}\right)\right\} - \frac{v_{\alpha\text{th}} v_{\alpha_x}}{\sqrt{2\pi}} \exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\text{th}}^2}\right]\right], \tag{D.0.13}$$

$$\int_{0}^{\infty} v_x^2 f_\alpha dv_x = n_\alpha \left[\frac{v_{\alpha\text{th}}^2 + v_{\alpha_x}^2}{2}\left\{1 + \text{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\text{th}}}\right)\right\} + \frac{v_{\alpha\text{th}} v_{\alpha_x}}{\sqrt{2\pi}} \exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\text{th}}^2}\right]\right]. \tag{D.0.14}$$

The moment term for the energy equation is given by Eq. (3.7.5). For 1V where v_x is the only velocity space dimension, the indefinite integral is

$$\int v_x^3 f_\alpha dv_x = n_\alpha \left[\frac{v_{\alpha_x}\left(3v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right)}{2}\text{erf}\left(\frac{v_x - v_{\alpha_x}}{\sqrt{2}v_{\alpha\text{th}}}\right) - \frac{v_{\alpha\text{th}}\left(2v_{\alpha\text{th}}^2 + v_{\alpha_x}^2 + v_{\alpha_x}v_x + v_x^2\right)}{\sqrt{2\pi}} \exp\left[\frac{-\left(v_x - v_{\alpha_x}\right)^2}{2v_{\alpha\text{th}}^2}\right]\right]. \tag{D.0.15}$$

This has the definite integrals

$$\int_{-\infty}^{\infty} v_x^3 f_\alpha dv_x = n_\alpha v_{\alpha_x}\left(3v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right), \tag{D.0.16}$$

$$\int_{-\infty}^{0} v_x^3 f_\alpha dv_x = n_\alpha \left[\frac{v_{\alpha_x}\left(3v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right)}{2}\left\{1 - \text{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\text{th}}}\right)\right\} - \frac{v_{\alpha\text{th}}\left(2v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right)}{\sqrt{2\pi}} \exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\text{th}}^2}\right]\right], \tag{D.0.17}$$

$$\int_{0}^{\infty} v_x^3 f_\alpha dv_x = n_\alpha \left[\frac{v_{\alpha_x}\left(3v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right)}{2}\left\{1 + \text{erf}\left(\frac{v_{\alpha_x}}{\sqrt{2}v_{\alpha\text{th}}}\right)\right\} + \frac{v_{\alpha\text{th}}\left(2v_{\alpha\text{th}}^2 + v_{\alpha_x}^2\right)}{\sqrt{2\pi}} \exp\left[\frac{-v_{\alpha_x}^2}{2v_{\alpha\text{th}}^2}\right]\right]. \tag{D.0.18}$$

Appendix E

METIS LIBRARY USAGE EXAMPLE

Here the usage of the Metis library [61] is shown for a very simple mesh. Imagine at 2D 2×2 mesh of triangles as shown in Fig. E.1. This grid consists of 9 nodes, and 8 triangle elements, each with 3 nodes per element. A typical meshfile will give the coordinates of the 9 nodes, a set of connectivities comprising of the nodes per element, and some sets of nodes comprising each boundary. For the DG algorithm, we add nodes internal to each triangle, the number and location determined by the order of the element. But only the mesh information as shown in this figure is needed for partitioning. This being small enough to observe by inspection, the neighborhoods and connectivities can be determined, assuming a counterclockwise numbering from the bottom as an example. Table E.1 shows these, where

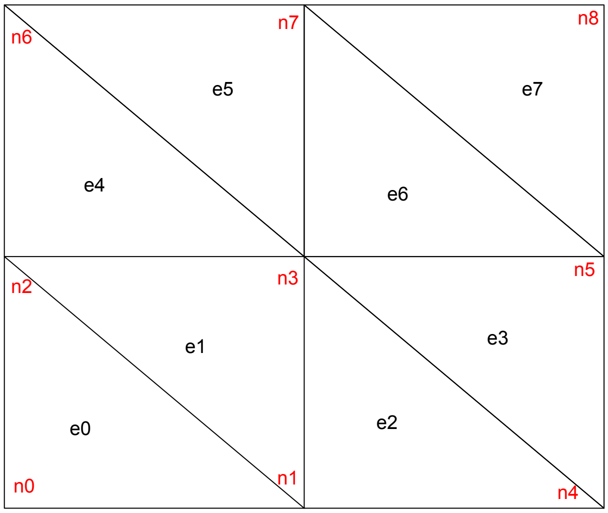


Figure E.1: 2D 2×2 gird of triangles

Element	Connectivity Nodes	Neighborhood Elements
0	$\{0, 1, 2\}$	$\{-1, 1, -1\}$
1	$\{1, 3, 2\}$	$\{0, 2, 4\}$
2	$\{1, 4, 3\}$	$\{-1, 3, 1\}$
3	$\{4, 5, 3\}$	$\{2, -1, 6\}$
4	$\{2, 3, 6\}$	$\{1, 5, -1\}$
5	$\{3, 7, 6\}$	$\{4, 6, -1\}$
6	$\{3, 5, 7\}$	$\{3, 7, 5\}$
7	$\{5, 8, 7\}$	$\{6, -1, -1\}$

Table E.1: Connectivities and Neighborhoods of a 2D 2×2 Triangular mesh

for neighborhoods, -1 is used to denote a boundary face. The Metis library can be used to apply a partitioning algorithm based on the multilevel partitioning paradigm, partitioning a graph into k parts using a recursive bisection or k-way method [60]. To use a function such as `METIS_PartGraphRecursive()` which uses the recursive bisection method, the elements and neighborhood are needed in a CSR format. The relationship between these are shown in Table E.1 in a matrix form as given

$$\text{Neighborhood} = \left[\begin{array}{c|cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 0 & & 1 & & & & & & \\ 1 & 1 & & 1 & & 1 & & & \\ 2 & & 1 & & 1 & & & & \\ 3 & & & 1 & & & & 1 & \\ 4 & & 1 & & & & 1 & & \\ 5 & & & & & 1 & & 1 & \\ 6 & & & & 1 & & 1 & & 1 \\ 7 & & & & & & & 1 & \end{array}\right]. \qquad \text{(E.0.1)}$$

This is stored in CSR format, with the column array

$$\text{adjncy} = [1, 0, 2, 4, 3, 1, 2, 6, 1, 5, 4, 6, 3, 7, 5, 6, 0, 0, 0, 0, 0, 0, 0, 0] \tag{E.0.2}$$

and row pointer array

$$\text{xadj} = [0, 1, 4, 6, 8, 10, 12, 15, 16]\,. \tag{E.0.3}$$

Now for this example the aim is to partition this mesh into 2 patches, each with an equivalent weight of 0.5. Calling `METIS_PartGraphRecursive()` yields a partitioning where elements 0,1,2,3 are on patch 0 and elements 4,5,6,7 are on patch 1, that is the lower and upper elements are on different patches. Setting weights and the partitioning algorithms differently can change the elements and number of elements on each patch.

The use of the function `METIS_PartMeshDual()` is also demonstrated, which can directly partition a mesh where the element connectivities are given. The number of nodes common on element edges must be specified, and the library will determine the neighborhood on its own. The correct number of common nodes must be chosen. For 1D line elements this is 1, for 2D triangles this is 2, and for 3D tetrahedron elements, this is 3. The element × node matrix representation of the connectivities is shown in Table E.1

$$\text{Connectivity} = \left[\begin{array}{c|ccccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \hline 0 & 1 & 1 & 1 & & & & & & \\ 1 & & 1 & 1 & 1 & & & & & \\ 2 & & 1 & & 1 & 1 & & & & \\ 3 & & & & 1 & 1 & 1 & & & \\ 4 & & & 1 & 1 & & & 1 & & \\ 5 & & & & 1 & & & 1 & 1 & \\ 6 & & & & 1 & & 1 & & 1 & \\ 7 & & & & & & 1 & & 1 & 1 \end{array}\right]. \tag{E.0.4}$$

This is stored in CSR format, with the column array

$$\text{eind} = [0, 1, 2, 1, 3, 2, 1, 4, 3, 4, 5, 3, 2, 3, 6, 3, 7, 6, 3, 5, 7, 5, 8, 7] \tag{E.0.5}$$

and row pointer array

$$\text{eptr} = [0, 3, 6, 9, 12, 15, 18, 21, 24]\,. \tag{E.0.6}$$

Calling `METIS_PartMeshDual()` with these arrays among 2 partitions, again with equal weights to each partition and directing it to use the bisection recursion method, the elements put into partition 0 are 0, 1, 2, 3 and the elements put into partition 1 are 4, 5, 6, 7, same as with the previous method.

There are various other parameters that can be put into these functions, including partition weights and even element weights. However, this shows the usage of Metis at this time in WARPXM. Specifically, the `METIS_PartGraphRecursive()` function is used with equal weightings among partitions. Better load balancing can be achieved when using the domain-decomposed hybrid method by adjusting weights of partitions and elements.

The code to generate this partition, both using `METIS_PartGraphRecursive()` and `METIS_PartMeshDual()` is given in the following code.

```cpp
// clang++ -std=c++11 main.cc -lmetis
#include <iostream>
#include <vector>
#include <string>
#include <metis.h>
// Get enum information in form we can read
// https://stackoverflow.com/questions/3342726/c-print-out-enum-value-as-text
// https://svn.alcf.anl.gov/repos/libs/METIS/include/metis.h
// error status
std::ostream& operator<<(std::ostream& out, const rstatus_et value){
    const char* s = 0;
#define PROCESS_VAL(p) case(p): s = #p; break;
    switch(value){PROCESS_VAL(METIS_OK); PROCESS_VAL(METIS_ERROR_INPUT); PROCESS_VAL(METIS_ERROR_MEMORY);
        PROCESS_VAL(METIS_ERROR);}
#undef PROCESS_VAL
    return out << s;
}
// object type status
std::ostream& operator<<(std::ostream& out, const mobjtype_et value){
    const char* s = 0;
#define PROCESS_VAL(p) case(p): s = #p; break;
    switch(value){PROCESS_VAL(METIS_OBJTYPE_CUT); PROCESS_VAL(METIS_OBJTYPE_VOL); PROCESS_VAL(
        METIS_OBJTYPE_NODE);}
#undef PROCESS_VAL
    return out << s;
}
// print function
void print(const std::vector<idx_t>& csr_rowptr, const std::vector<idx_t>& csr_col, const std::string&
    csr_rowptr_str, const std::string& csr_col_str){
```

```cpp
    // print rowptr
    std::cout << csr_rowptr_str << " = [";
    for (int i = 0; i < csr_rowptr.size(); i++){
        std::cout << csr_rowptr[i];
        if (i != csr_rowptr.size() - 1)
            std::cout << ", ";
    }
    std::cout << "]" << std::endl;
    // print col
    std::cout << csr_col_str << " = [";
    for (int i = 0; i < csr_col.size(); i++){
        std::cout << csr_col[i];
        if (i != csr_col.size() - 1)
            std::cout << ", ";
    }
    std::cout << "]" << std::endl;
    // print the col for each element
    for (int i = 0; i < csr_rowptr.size() - 1; i++){
        std::cout << "Element " << i << " : [";
        for (int j = csr_rowptr[i]; j < csr_rowptr[i + 1]; j++){
            std::cout << csr_col[j];
            if (j != csr_rowptr[i + 1] - 1)
                std::cout << ", ";
        }
        std::cout << "]" << std::endl;
    }
}
// graph partition function
int partition_graph(const std::vector<std::vector<int>>& neighborhood, const std::vector<real_t>& tpwgts, int num_partitions, std::vector<idx_t>& xadj, std::vector<idx_t>& adjncy, idx_t& objval, std::vector<idx_t>& part){
    int num_elems = neighborhood.size();
    int num_faces = neighborhood[0].size();
    // Fill xadj and adjncy
    idx_t adjncyindex = 0, xadjindex = 0;
    // xadj starts with 0
    xadj[xadjindex++] = 0;
    for (int i = 0; i < num_elems; i++){
        // Run through element neighbors and add them to adjncy
        for (int j = 0; j < num_faces; j++){
            if (neighborhood[i][j] > -1)
                adjncy[adjncyindex++] = neighborhood[i][j];
        }
        xadj[xadjindex++] = adjncyindex;
    }
    // partition
    idx_t ncon = 1;                  // number of balancing constraints
    idx_t options[METIS_NOPTIONS]; // set options
    METIS_SetDefaultOptions(options);
    options[METIS_OPTION_NUMBERING] = 0; // C-style for adjency structure
    options[METIS_OPTION_NITER] = 100; // # uncoarsening iterations
    options[METIS_OPTION_SEED] = 42; // seed for random number generator
    // recursive bisection
    return METIS_PartGraphRecursive(&num_elems, &ncon, xadj.data(), adjncy.data(), nullptr, nullptr, nullptr, &num_partitions, const_cast<real_t*>(tpwgts.data()), nullptr, options, &objval, part.data());
    // // kway
```

```
    // return METIS_PartGraphKway(&num_elems, &ncon, xadj.data(), adjncy.data(), nullptr, nullptr, nullptr, &num_partitions, const_cast<real_t*>(tpwgts.data()), nullptr, options, &objval, part.data());
}
// mesh partition function
int partition_mesh(const std::vector<std::vector<int>>& connectivity, const std::vector<real_t>& tpwgts, int num_nodes, int num_partitions, std::vector<idx_t>& eptr, std::vector<idx_t>& eind, idx_t& objval, std::vector<idx_t>& epart, std::vector<idx_t>& npart){
    int num_elems = connectivity.size();
    int num_nodes_per_element = connectivity[0].size(); // assume same elems
    // Fill eptr and eind
    idx_t eptrindex = 0, eindindex = 0;
    eptr[eptrindex++] = 0;
    for (int i = 0; i < num_elems; i++){
        // Run through element connecivity and add them to eind
        for (int j = 0; j < num_nodes_per_element; j++){
            eind[eindindex++] = connectivity[i][j];
        }
        eptr[eptrindex++] = eindindex;
    }
    // partition
    // n_common is # of nodes along a common face.
    // lines: n_common = 1, triangles: n_common = 2, tets: n_common = 3
    idx_t n_common = 2;            // we have triangles
    idx_t options[METIS_NOPTIONS]; // set options
    METIS_SetDefaultOptions(options);
    options[METIS_OPTION_NUMBERING] = 0; // C-style for adjency structure
    options[METIS_OPTION_PTYPE] = METIS_PTYPE_RB; // use recursive bisection
    options[METIS_OPTION_NITER] = 100; // # uncoarsening iterations
    options[METIS_OPTION_SEED] = 42; // seed for random number generator
    // dual partitioner makes elements as the "nodes"
    return METIS_PartMeshDual(&num_elems, &num_nodes, eptr.data(), eind.data(), nullptr, nullptr, &n_common, &num_partitions, const_cast<real_t*>(tpwgts.data()), options, &objval, epart.data(), npart.data());
    // // nodal partitioner makes nodes as the "nodes"
    // return METIS_PartMeshNodal(&num_elems, &num_nodes, eptr.data(), eind.data(), nullptr, nullptr, &num_partitions, const_cast<real_t*>(tpwgts.data()), options, &objval, epart.data(), npart.data());
}
int main(int argc, char** argv){
    // Imagine a 2x2 square grid broken into 8 triangles.
    const int num_elems = 8, num_faces = 3, num_nodes_per_element = 3, num_nodes = 9;
    // number of partitions
    const int num_partitions = 2;
    // Weights must be sum to 1
    // assume both same weight
    std::vector<real_t> tpwgts(num_partitions, 1.0 / num_partitions);
    // Graph partition needs elements as "nodes" and neighborhood as "edges".
    std::cout << "Graph Partition" << std::endl;
    std::vector<int> face_v_tmp(num_faces, 0);
    std::vector<std::vector<int>> neighborhood(num_elems, face_v_tmp);
    neighborhood[0][0] = -1; neighborhood[0][1] = 1; neighborhood[0][2] = -1;
    neighborhood[1][0] = 0; neighborhood[1][1] = 2; neighborhood[1][2] = 4;
    neighborhood[2][0] = -1; neighborhood[2][1] = 3; neighborhood[2][2] = 1;
    neighborhood[3][0] = 2; neighborhood[3][1] = -1; neighborhood[3][2] = 6;
    neighborhood[4][0] = 1; neighborhood[4][1] = 5; neighborhood[4][2] = -1;
    neighborhood[5][0] = 4; neighborhood[5][1] = 6; neighborhood[5][2] = -1;
    neighborhood[6][0] = 3; neighborhood[6][1] = 7; neighborhood[6][2] = 5;
    neighborhood[7][0] = 6; neighborhood[7][1] = -1; neighborhood[7][2] = -1;
```

```
    // The node and edge element information is given in CSR format.
    // val=1 (or true) at rows (elements) and columns (neighborhood elements).
    // Therefore don't need to pass in val array.
    // xadj=rowptr, points to col index, giving elements of the neighborhood.
    // adjncy=col, the neighborhood elements.
    std::vector<idx_t> xadj(num_elems + 1), adjncy(num_elems * num_faces), part(num_elems);
    idx_t objval_graph;
    int error_graph = partition_graph(neighborhood, tpwgts, num_partitions, xadj, adjncy, objval_graph, part);
    print(xadj, adjncy, "xadj", "adjncy");
    std::cout << "error_graph = " << rstatus_et(error_graph) << " objval_graph = " << mobjtype_et(objval_graph) << std::endl;
    for (int i = 0; i < num_elems; i++)
        std::cout << "elem " << i << ", partition " << part[i] << std::endl;
    //  //  //  //  //  //  //  //  //  //  //  //  //  //  //  //  //  //
    // Often the meshfile won't have neighborhoods but will have connectivity.
    // With mesh partition, you can give the elements and the connectivity.
    std::cout << "\nMesh Partition" << std::endl;
    std::vector<int> node_temp_v(num_nodes_per_element, 0);
    std::vector<std::vector<int>> connectivity(num_elems, node_temp_v);
    connectivity[0][0] = 0; connectivity[0][1] = 1; connectivity[0][2] = 2;
    connectivity[1][0] = 1; connectivity[1][1] = 3; connectivity[1][2] = 2;
    connectivity[2][0] = 1; connectivity[2][1] = 4; connectivity[2][2] = 3;
    connectivity[3][0] = 4; connectivity[3][1] = 5; connectivity[3][2] = 3;
    connectivity[4][0] = 2; connectivity[4][1] = 3; connectivity[4][2] = 6;
    connectivity[5][0] = 3; connectivity[5][1] = 7; connectivity[5][2] = 6;
    connectivity[6][0] = 3; connectivity[6][1] = 5; connectivity[6][2] = 7;
    connectivity[7][0] = 5; connectivity[7][1] = 8; connectivity[7][2] = 7;
    // The mesh element and connectivity information is given in CSR format.
    // val=1 (or true) at rows (elements) and columns (connectivity nodes).
    // Therefore don't need to pass in val array.
    // eptr=rowptr, points to col index, giving nodes of the element.
    // eind=col, the connectivity nodes.
    std::vector<idx_t> eptr(num_elems + 1), eind(num_elems * num_nodes_per_element), epart(num_elems), npart(num_nodes);
    idx_t objval_mesh;
    int error_mesh = partition_mesh(connectivity, tpwgts, num_nodes, num_partitions, eptr, eind, objval_mesh, epart, npart);
    print(eptr, eind, "eptr", "eind");
    std::cout << "error_mesh = " << rstatus_et(error_mesh) << " objval_mesh = " << mobjtype_et(objval_mesh) << std::endl;
    for (int i = 0; i < num_elems; i++)
        std::cout << "elem " << i << ", epartition " << epart[i] << std::endl;
    for (int i = 0; i < num_nodes; i++)
        std::cout << "node " << i << ", npartition " << npart[i] << std::endl;
    return 0;
}
```

Listing E.1: Example c++ file showing use of Metis Graph Partitioning

Appendix F

HARTMANN FLOW EQUILIBRIUM

The visco-resistive MHD equations for this problem, normalized should be

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0, \tag{F.0.1}$$

$$\frac{\partial (\rho \boldsymbol{u})}{\partial t} + \nabla \cdot \left[\rho \boldsymbol{u}\boldsymbol{u} + p\overline{\overline{I}} - \left(\boldsymbol{B}\boldsymbol{B} - \frac{1}{2}B^2\overline{\overline{I}}\right) + \frac{1}{(\nu_p \tau)}\overline{\overline{\Pi}}\right] = 0, \tag{F.0.2}$$

$$\frac{\partial e_t}{\partial t} + \nabla \cdot \left[\left(e_t + p + \frac{1}{2}B^2\right)\boldsymbol{u} - (\boldsymbol{u} \cdot \boldsymbol{B})\,\boldsymbol{B} - \left(\frac{\delta_p}{L}\right)(\nu_p \tau)\,\boldsymbol{B} \times \eta \boldsymbol{J} + \frac{1}{(\nu_p \tau)}\overline{\overline{\Pi}} \cdot \boldsymbol{v} + \frac{1}{(\nu_p \tau)}\boldsymbol{h}\right] = 0, \tag{F.0.3}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \times \left(-\boldsymbol{u} \times \boldsymbol{B} + \left(\frac{\delta_p}{L}\right)(\nu_p \tau)\,\eta \boldsymbol{J}\right) = 0, \tag{F.0.4}$$

with Newtonian viscosity relations

$$\overline{\overline{\Pi}} = -\mu\left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T - \frac{2}{3}\nabla \cdot \boldsymbol{u}\overline{\overline{I}}\right), \tag{F.0.5}$$

$$\boldsymbol{h} = -K\nabla T. \tag{F.0.6}$$

Dynamic viscosity for some species α is

$$\mu_\alpha = \frac{p_\alpha}{\nu_{\alpha\alpha}}, \tag{F.0.7}$$

where

$$\nu_{\alpha\beta} = 2^{\frac{1}{2}} Z_\beta^2 Z_\alpha^2 \ln \Lambda \frac{(A_\alpha + A_\beta)}{A_\alpha^2 A_\beta^2} \frac{\rho_\beta}{(\theta_\alpha + \theta_\beta)^{\frac{3}{2}}}.$$

Thermal conductivity is

$$K = \frac{3p_\alpha}{\nu_{\alpha\alpha} A_\alpha}.$$

Some important parameters appear in this problem. Fluid viscosity, μ_v

$$\mu_v = \frac{p_0}{\nu_p}. \tag{F.0.8}$$

The Hartmann number:

$$\mathrm{Ha} = \frac{B_0 L}{\sqrt{\eta_0 \mu_{v0}}} = B_0 L \sqrt{\frac{1}{\frac{m_0 \nu_p}{e^2 n_0} \frac{m_0 n_0 v_0^2}{\nu_p}}} = B_0 L \sqrt{\frac{e^2}{m_0^2 v_0^2}} = B_0 L \frac{e}{m_0 v_0} = \frac{L}{v_0} \frac{e B_0}{m_0} = \omega_c \tau = \left(\frac{L}{\delta_p}\right). \tag{F.0.9}$$

Similarly for Reynolds Number:

$$\mathrm{Re} = \frac{\rho_0 v_0 L}{\mu_{v0}} = \frac{m_0 n_0 v_0^2 \tau}{\frac{m_0 n_0 v_0^2}{\nu_p}} = \nu_p \tau. \tag{F.0.10}$$

F.1 Analytic Solution

F.1.1 Conditions on v_y and B_y

Assume steady state, incompressible flow with Newtonian viscosity. Also energy is not evolved. The aim will be to derive expressions for v_y and B_y. The solution here mimics the procedure given by Dellar [38] and Martinez [75], though for the normalization present in this work. Before applying the assumptions, it is helpful to convert continuity and momentum to primitive formulations.

For continuity:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) &= 0 \\
\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_j)}{\partial x_j} &= 0 \\
\frac{\partial \rho}{\partial t} + \rho \frac{\partial u_j}{\partial x_j} + u_j \frac{\partial \rho}{\partial x_j} &= 0 \\
\frac{\partial \rho}{\partial t} + \rho \nabla \cdot \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \rho &= 0.
\end{aligned} \tag{F.1.1}$$

For Momentum:

$$\begin{aligned}
\frac{\partial\left(\rho\boldsymbol{u}\right)}{\partial t}+\nabla\cdot\left[\rho\boldsymbol{u}\boldsymbol{u}+p\overline{\overline{I}}-\left(\boldsymbol{B}\boldsymbol{B}-\frac{1}{2}B^2\overline{\overline{I}}\right)-\frac{\mu}{\left(\nu_p\tau\right)}\left(\nabla\boldsymbol{u}+\nabla\boldsymbol{u}^T-\frac{2}{3}\nabla\cdot\boldsymbol{u}\overline{\overline{I}}\right)\right]=&0\\
\frac{\partial\left(\rho u_i\right)}{\partial t}+\frac{\partial}{\partial x_j}\left[\rho u_i u_j+p\delta_{ij}-\left(B_iB_j-\frac{1}{2}B^2\delta_{ij}\right)-\frac{\mu}{\left(\nu_p\tau\right)}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right]=&0\\
\rho\frac{\partial u_i}{\partial t}+u_i\frac{\partial\rho}{\partial t}+\rho\frac{\partial u_i}{\partial x_j}u_j+u_i\frac{\partial\rho}{\partial x_j}u_j+\rho u_i\frac{\partial u_j}{\partial x_j}+\frac{\partial}{\partial x_j}\left[p\delta_{ij}-\left(B_iB_j-\frac{1}{2}B^2\delta_{ij}\right)-\frac{\mu}{\left(\nu_p\tau\right)}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right]=&0\\
\rho\frac{\partial u_i}{\partial t}+\rho\frac{\partial u_i}{\partial x_j}u_j+\frac{\partial}{\partial x_j}\left[p\delta_{ij}-\left(B_iB_j-\frac{1}{2}B^2\delta_{ij}\right)-\frac{\mu}{\left(\nu_p\tau\right)}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}-\frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right]=&0\\
\rho\frac{\partial u_i}{\partial t}+\rho\frac{\partial u_i}{\partial x_j}u_j+\frac{\partial}{\partial x_i}\left[p+\frac{B^2}{2}\right]-B_i\frac{\partial B_j}{\partial x_j}-B_j\frac{\partial B_i}{\partial x_j}-\left[\frac{\mu}{\left(\nu_p\tau\right)}\left(\frac{\partial^2 u_i}{\partial x_j\partial x_j}+\frac{\partial^2 u_j}{\partial x_i\partial x_j}-\frac{2}{3}\frac{\partial}{\partial x_j}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right]=&0\\
\rho\frac{\partial u_i}{\partial t}+\rho\frac{\partial u_i}{\partial x_j}u_j+\frac{\partial}{\partial x_i}\left[p+\frac{B^2}{2}\right]-B_j\frac{\partial B_i}{\partial x_j}-\left[\frac{\mu}{\left(\nu_p\tau\right)}\left(\frac{\partial^2 u_i}{\partial x_j\partial x_j}+\frac{\partial^2 u_j}{\partial x_i\partial x_j}-\frac{2}{3}\frac{\partial}{\partial x_i}\frac{\partial u_k}{\partial x_k}\right)\right]=&0\\
\rho\left[\frac{\partial\boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right]+\nabla\left[p+\frac{B^2}{2}\right]-\boldsymbol{B}\cdot\nabla\boldsymbol{B}-\left[\frac{\mu}{\left(\nu_p\tau\right)}\left(\Delta\boldsymbol{u}+\frac{1}{3}\nabla\nabla\cdot\boldsymbol{u}\right)\right]=&0.
\end{aligned}\tag{F.1.2}$$

Now the incompressible assumption ($\frac{\partial\rho}{\partial t}+\boldsymbol{u}\cdot\nabla\rho=0$) implies the continuity equation becomes

$$\nabla\cdot\boldsymbol{u}=0.\tag{F.1.3}$$

This is applied to the momentum equation along with the steady state assumption ($\frac{\partial}{\partial t}=0$), yielding

$$\rho\left[\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right]+\nabla\left[p+\frac{B^2}{2}\right]-\boldsymbol{B}\cdot\nabla\boldsymbol{B}=\frac{\mu}{\left(\nu_p\tau\right)}\Delta\boldsymbol{u}.\tag{F.1.4}$$

The induction Eq. (F.0.4) can be turned into another form noting formulary vector identities 10 and 14 [58] along with

$$\boldsymbol{j}=\left(\frac{\delta_p}{L}\right)\nabla\times\boldsymbol{B}\tag{F.1.5}$$

and incompressibility of both $\boldsymbol{B}$ and $\boldsymbol{u}$ and the steady state assumption, leading to

$$\left(\boldsymbol{u}\cdot\nabla\right)\boldsymbol{B}-\left(\boldsymbol{B}\cdot\nabla\right)\boldsymbol{u}=\left(\frac{\delta_p}{L}\right)^2\left(\nu_p\tau\right)\eta\Delta\boldsymbol{B}.\tag{F.1.6}$$

Now the vector quantities are defined as

$$\boldsymbol{u}=v(x)\hat{y},\tag{F.1.7}$$

$$\boldsymbol{B} = B_x\hat{x} + B_y(x)\hat{y}, \tag{F.1.8}$$

where $B_x = \text{const}$. The momentum equation for v from Eq. (F.1.4) is

$$\begin{aligned} \rho\left[u_j\frac{\partial v}{\partial x_j}\right] + \frac{\partial}{\partial y}\left[p + \frac{B^2}{2}\right] - B_j\frac{\partial B_y}{\partial x_j} &= \frac{\mu}{(\nu_p\tau)}\frac{\partial^2 v}{\partial x_j \partial x_j} \\ \frac{\partial p}{\partial y} - B_x\frac{\partial B_y}{\partial x} &= \frac{\mu}{(\nu_p\tau)}\frac{\partial^2 v}{\partial x^2}. \end{aligned} \tag{F.1.9}$$

Similarly with the induction Eq. (F.1.6) for B_y

$$\begin{aligned} \left(u_j\frac{\partial}{\partial x_j}\right)B_y - \left(B_j\frac{\partial}{\partial x_j}\right)u_y &= \left(\frac{\delta_p}{L}\right)^2(\nu_p\tau)\,\eta\frac{\partial^2 B_y}{\partial x_j\partial x_j} \\ -B_x\frac{\partial v}{\partial x} &= \left(\frac{\delta_p}{L}\right)^2(\nu_p\tau)\,\eta\frac{\partial^2 B_y}{\partial x^2}. \end{aligned} \tag{F.1.10}$$

Equation (F.1.9) can be rearranged

$$\frac{\partial^2 v}{\partial x^2} = -\frac{(\nu_p\tau)}{\mu}B_x\frac{\partial B_y}{\partial x} + \frac{(\nu_p\tau)}{\mu}\frac{\partial p}{\partial y} \tag{F.1.11}$$

and Eq. (F.1.10) can be rearranged

$$\frac{\partial^2 B_y}{\partial x^2} = -\frac{\left(\frac{L}{\delta_p}\right)^2}{\eta\,(\nu_p\tau)}B_x\frac{\partial v}{\partial x}. \tag{F.1.12}$$

In terms of the other normalizations these equations are

$$\frac{\partial^2 v}{\partial x^2} = -\frac{\text{Re}}{\mu}B_x\frac{\partial B_y}{\partial x} + \frac{\text{Re}}{\mu}\frac{\partial p}{\partial y}, \tag{F.1.13}$$

$$\frac{\partial^2 B_y}{\partial x^2} = -\frac{\text{Ha}^2}{\text{Re}}\frac{B_x}{\eta}\frac{\partial v}{\partial x}. \tag{F.1.14}$$

This system is solved analytically using Mathematica on a domain $x \in [-L, L]$ with rigid conducting walls, such that $v(x = -L) = -V_{\text{wall}}$, $v(x = L) = +V_{\text{wall}}$ (Couette conditions), $B_y(x = -L) = 0$, $B_y(x = +L) = 0$. The solution from Mathematica, after some algebraic manipulation becomes

$$B_y(x) = \frac{\left(-\frac{\partial p}{\partial y}\right)L}{B_x}\left(\frac{\sinh\left(\text{Ha}\tilde{\text{Ha}}\frac{x}{L}\right)}{\sinh\left(\text{Ha}\tilde{\text{Ha}}\right)} - \frac{x}{L}\right) + v_w\sqrt{\frac{\mu}{\eta}}\frac{\text{Ha}}{\text{Re}}\coth\left(\text{Ha}\tilde{\text{Ha}}\right)\left[1 - \frac{\cosh\left(\text{Ha}\tilde{\text{Ha}}\frac{x}{L}\right)}{\cosh\left(\text{Ha}\tilde{\text{Ha}}\right)}\right], \tag{F.1.15}$$

$$v(x) = \frac{\left(-\frac{\partial p}{\partial y}\right) L}{B_x} \sqrt{\frac{\eta}{\mu}} \frac{\mathrm{Re}}{\mathrm{Ha}} \coth\left(\mathrm{Ha}\tilde{H}a\right) \left[1 - \frac{\cosh\left(\mathrm{Ha}\tilde{H}a\frac{x}{L}\right)}{\cosh\left(\mathrm{Ha}\tilde{H}a\right)}\right] + v_w \frac{\sinh\left(\mathrm{Ha}\tilde{\mathrm{Ha}}\frac{x}{L}\right)}{\sinh\left(\mathrm{Ha}\tilde{\mathrm{Ha}}\right)}. \tag{F.1.16}$$

A few examples are plotted. A Couette case with only a V$_{wall}$ and magnetization is shown in Fig. F.1. A Poiseuille case with only pressure gradient and magnetization is shown in Fig. F.2. Combination cases are shown in Figs. F.3 and F.4.

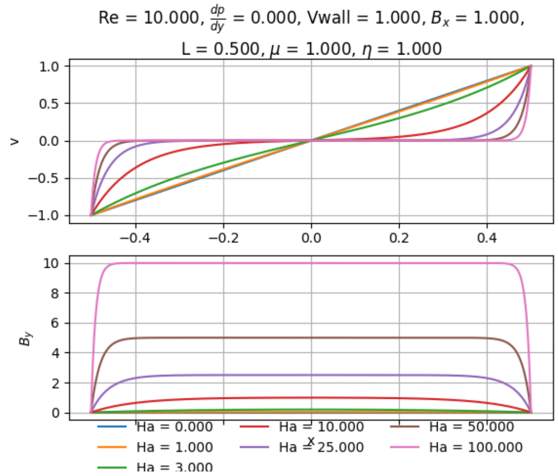


Figure F.1: Analytic Hartmann Flow Equilibrium: Couette Case with Magnetic Field

F.1.2 Condition on the other variables

The equilibrium on the other MHD variables can determined by substituting the equilibrium v and B_y back into the equations.

- Continuity

$$\frac{\partial\left(\rho u_j\right)}{\partial x_j} = 0$$

$$\frac{\partial}{\partial x}\left(\rho u\right) + \frac{\partial}{\partial y}\left(\rho v\right) = 0. \tag{F.1.17}$$

This is satisfied trivially due to constancy of u and $v = v(x)$.

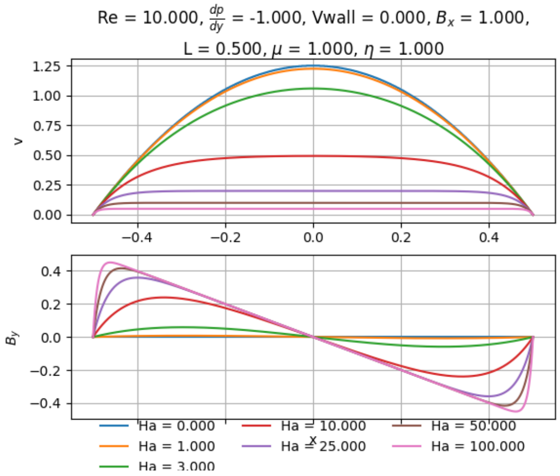


Figure F.2: Analytic Hartmann Flow Equilibrium: Poiseuille Case with Magnetic Field

- Momentum

$$
\begin{aligned}
\frac{\partial}{\partial x_j}\left[\rho u_i u_j + p\delta_{ij} - \left(B_iB_j - \frac{1}{2}B^2\delta_{ij}\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right] =& 0\\
\frac{\partial}{\partial x}\left[\rho u_i u + p\delta_{i1} - \left(B_iB_x - \frac{1}{2}\left(B_x^2 + B_y^2 + B_z^2\right)\delta_{i1}\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u_i}{\partial x} + \frac{\partial u}{\partial x_i} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\delta_{i1}\right)\right]&\\
+\frac{\partial}{\partial y}\left[\rho u_i v + p\delta_{i2} - \left(B_iB_y - \frac{1}{2}\left(B_x^2 + B_y^2 + B_z^2\right)\delta_{i2}\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u_i}{\partial y} + \frac{\partial v}{\partial x_i} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\delta_{i2}\right)\right]&\\
+\frac{\partial}{\partial z}\left[\rho u_i w + p\delta_{i3} - \left(B_iB_z - \frac{1}{2}\left(B_x^2 + B_y^2 + B_z^2\right)\delta_{i3}\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u_i}{\partial z} + \frac{\partial w}{\partial x_i} - \frac{2}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\right)\delta_{i3}\right)\right] =& 0\\
\frac{\partial}{\partial x}\left[\rho u_i u + p\delta_{i1} - \left(B_iB_x - \frac{1}{2}\left(B_y^2\right)\delta_{i1}\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u_i}{\partial x}\right)\right] + \frac{\partial}{\partial y}\left[p\delta_{i2}\right] + \frac{\partial}{\partial z}\left[0\right] =& 0\\
\frac{\partial}{\partial x}\begin{pmatrix} \rho uu + p - \left(B_xB_x - \frac{1}{2}\left(B_y^2\right)\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial u}{\partial x}\right)\\ \rho vu - \left(B_yB_x\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial v}{\partial x}\right)\\ \rho wu - \left(B_zB_x\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial w}{\partial x}\right)\end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix}0\\ p\\ 0\end{pmatrix} =& 0\\
\frac{\partial}{\partial x}\begin{pmatrix} p + \frac{1}{2}B_y^2\\ -\left(B_yB_x\right) - \frac{\mu}{(\nu_p\tau)}\left(\frac{\partial v}{\partial x}\right)\\ 0\end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix}0\\ p\\ 0\end{pmatrix} =& 0.
\end{aligned}
\tag{F.1.18}
$$

The y component does match Eq. (F.1.9). The x momentum specifies pressure, once

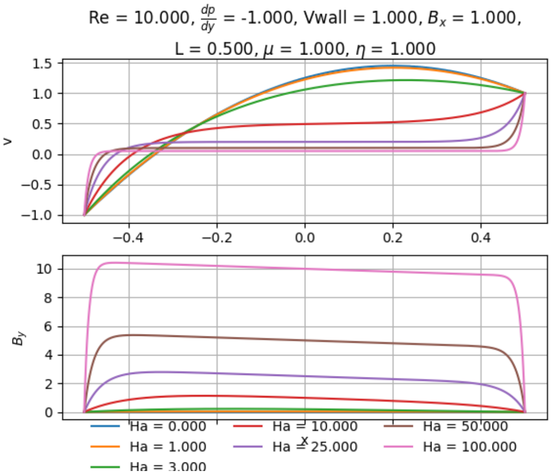


Figure F.3: Analytic Hartmann Flow Equilibrium: Combination Case 1

B_y is solved for. It becomes

$$\begin{aligned}\frac{\partial}{\partial x}\left(p+\frac{1}{2}B_y^2\right) =&0\\ p =&-\frac{1}{2}B_y^2+C.\end{aligned}\tag{F.1.19}$$

- Induction

$$\begin{aligned}\nabla\times\left(-\boldsymbol{u}\times\boldsymbol{B}+\left(\frac{\delta_p}{L}\right)(\nu_p\tau)\,\eta\boldsymbol{J}\right) =&0\\ \nabla\times\left(-\boldsymbol{u}\times\boldsymbol{B}+\left(\frac{\delta_p}{L}\right)(\nu_p\tau)\,\eta\left(\frac{\delta_p}{L}\right)\nabla\times\boldsymbol{B}\right) =&0\\ \nabla\cdot(\boldsymbol{uB}-\boldsymbol{Bu})-\left(\frac{\delta_p}{L}\right)^2(\nu_p\tau)\,\eta\nabla^2\boldsymbol{B} =&0\\ \frac{\partial}{\partial x}\begin{pmatrix}0\\ uB_y-B_xv\\ uB_z-B_xw\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}vB_x-B_yu\\ 0\\ vB_z-B_yw\end{pmatrix}+\frac{\partial}{\partial z}\begin{pmatrix}wB_x-B_zu\\ wB_y-B_zv\\ 0\end{pmatrix}-\left(\frac{\delta_p}{L}\right)^2(\nu_p\tau)\,\eta\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2}\right)\begin{pmatrix}B_x\\ B_y\\ B_z\end{pmatrix} =&0\\ \frac{\partial}{\partial x}\begin{pmatrix}0\\ -B_xv\\ 0\end{pmatrix}-\left(\frac{\delta_p}{L}\right)^2(\nu_p\tau)\,\eta\left(\frac{\partial^2}{\partial x^2}\right)\begin{pmatrix}0\\ B_y\\ 0\end{pmatrix} =&0.\end{aligned}\tag{F.1.20}$$

This does match Eq. (F.1.10).

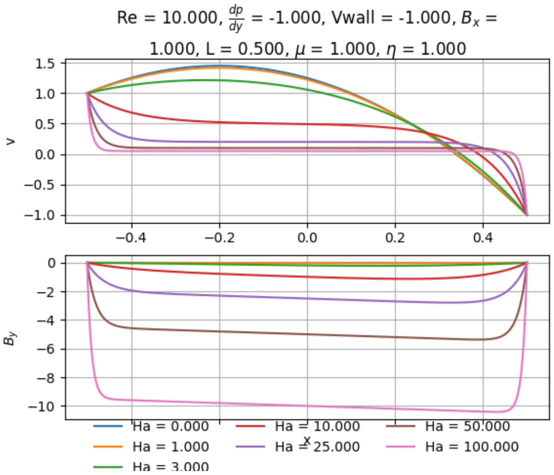


Figure F.4: Analytic Hartmann Flow Equilibrium: Combination Case 2 (Case 1 with wall velocities flipped)

Density can also be set by forcing temperature to be constant so as to not invoke the energy equation, implying $\rho = p$ $(T = p/n)$. Thus, the equilbrium, in the absence of an energy evolution is given by,

$$\rho = -0.5B_y^2 + C \tag{F.1.21}$$

$$u = 0 \tag{F.1.22}$$

$$v = v(x) \tag{F.1.23}$$

$$w = 0 \tag{F.1.24}$$

$$p = -0.5B_y^2 + C \tag{F.1.25}$$

$$B_x = 1 \tag{F.1.26}$$

$$B_y = By(x) \tag{F.1.27}$$

$$B_z = 0. \tag{F.1.28}$$

Appendix G

GEM CHALLENGE EQUILIBRIUM SETUP

G.1 Initial Conditions

From Birn [8], the initial conditions are:

$$B_x(z) = B_0 \tanh\left(\frac{z}{\lambda}\right),$$

$$n(z) = n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty.$$

Electron and ion temperatures T_i and T_e are taken to be uniform in the initial state. The initial magnetic flux perturbation is given by:

$$\psi(x,z) = \psi_0 \cos\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right),$$

where the magnetic field perturbation is given by:

$$\delta\mathbf{B} = \hat{y} \times \nabla\psi.$$

So then

$$\begin{aligned} d\mathbf{B} =& \hat{y} \times \left[\partial_x \psi_0 \cos\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right)\hat{x} + \partial_y \psi_0 \cos\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right)\hat{y} + \partial_z \psi_0 \cos\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right)\hat{z}\right] \\ =& \psi_0 \hat{y} \times \left[-\frac{2\pi}{L_x}\sin\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right)\hat{x} - \frac{\pi}{L_z}\cos\left(\frac{2\pi x}{L_x}\right)\sin\left(\frac{\pi z}{L_z}\right)\hat{z}\right] \\ =& \psi_0 \left[\frac{2\pi}{L_x}\sin\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi z}{L_z}\right)\hat{z} - \frac{\pi}{L_z}\cos\left(\frac{2\pi x}{L_x}\right)\sin\left(\frac{\pi z}{L_z}\right)\hat{x}\right]. \end{aligned}$$

Birn says pressure balance gives $n_0(T_e + T_i) = B_0^2/8\pi$. This can be found by taking MHD equilibrium (with $n_i = n_e = n$):

$$\begin{aligned}
\nabla p =& \mathbf{J} \times \mathbf{B} \\
\nabla \left(n_i T_i + n_e T_e\right) =& \frac{c}{4\pi} \left(\nabla \times \mathbf{B}\right) \times \mathbf{B} \\
\left(T_i + T_e\right) \partial_z n \hat{z} =& \frac{c}{4\pi} \left[\left(\partial_x \hat{x} + \partial_y \hat{y} + \partial_z \hat{z}\right) \times \left(B_x \hat{x}\right)\right] \times B_x \hat{x} \\
\left(T_i + T_e\right) \partial_z n \hat{z} =& \frac{c}{4\pi} \left[\partial_y B_x \left(-\hat{z}\right) + \partial_z B_x \hat{y}\right] \times B_x \hat{x} \\
\left(T_i + T_e\right) \partial_z n \hat{z} =& \frac{c}{4\pi} \left[\partial_y \left(B_x\right) B_x \left(-\hat{y}\right) + \partial_z \left(B_x\right) B_x \left(-\hat{z}\right)\right] \\
\left(T_i + T_e\right) \partial_z n \hat{z} =& -\frac{c}{4\pi} \partial_z \left(B_x\right) B_x \hat{z} \\
\left(T_i + T_e\right) \frac{n_0}{\lambda} \left(-2 \tanh\left(\frac{z}{\lambda}\right) \operatorname{sech}^2\left(\frac{z}{\lambda}\right)\right) =& -\frac{c}{4\pi} \frac{B_0}{\lambda} \operatorname{sech}^2\left(\frac{z}{\lambda}\right) B_0 \tanh\left(\frac{z}{\lambda}\right) \\
n_0 \left(T_i + T_e\right) =& \frac{c}{4\pi} \frac{B_0^2}{2}.
\end{aligned}$$

In terms of two-fluid, momentum equilibrium should give us:

$$\nabla \cdot \left(\rho_\alpha \mathbf{u}_\alpha \mathbf{u}_\alpha\right) + \nabla p_\alpha = \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha \rho_\alpha}{A_\alpha} \left(\mathbf{E} + \mathbf{u}_\alpha \times \mathbf{B}\right).$$

Assume initially $\mathbf{B} = 0$ and the divergence is defined $\nabla \cdot \overline{\overline{A}} = \partial_j A_{ij}$. Then:

$$\begin{aligned}
\nabla \cdot \left(\rho_\alpha \mathbf{u}_\alpha \mathbf{u}_\alpha\right) + \nabla p_\alpha &= \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha \rho_\alpha}{A_\alpha} \left(\mathbf{E} + \mathbf{u}_\alpha \times \mathbf{B}\right) \\
\nabla \cdot \begin{pmatrix} \rho_\alpha u_x^\alpha u_x^\alpha & \rho_\alpha u_x^\alpha u_y^\alpha & \rho_\alpha u_x^\alpha u_z^\alpha \\ \rho_\alpha u_y^\alpha u_x^\alpha & \rho_\alpha u_y^\alpha u_y^\alpha & \rho_\alpha u_y^\alpha u_z^\alpha \\ \rho_\alpha u_z^\alpha u_x^\alpha & \rho_\alpha u_z^\alpha u_y^\alpha & \rho_\alpha u_z^\alpha u_z^\alpha \end{pmatrix} + \nabla \left(n_\alpha T_\alpha\right) &= \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha \rho_\alpha}{A_\alpha} \left(\mathbf{u}_\alpha \times \mathbf{B}\right) \\
\begin{pmatrix} \partial_x \left(\rho_\alpha u_x^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_x^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_x^\alpha u_z^\alpha\right) \\ \partial_x \left(\rho_\alpha u_y^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_y^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_y^\alpha u_z^\alpha\right) \\ \partial_x \left(\rho_\alpha u_z^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_z^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_z^\alpha u_z^\alpha\right) \end{pmatrix} + \begin{pmatrix} \partial_x \left(n_\alpha T_\alpha\right) \\ \partial_y \left(n_\alpha T_\alpha\right) \\ \partial_z \left(n_\alpha T_\alpha\right) \end{pmatrix} &= \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha \rho_\alpha}{A_\alpha} \begin{pmatrix} u_y^\alpha B_z - u_z^\alpha B_y \\ u_z^\alpha B_x - u_x^\alpha B_z \\ u_x^\alpha B_y - u_y^\alpha B_x \end{pmatrix} \\
\begin{pmatrix} \partial_x \left(\rho_\alpha u_x^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_x^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_x^\alpha u_z^\alpha\right) \\ \partial_x \left(\rho_\alpha u_y^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_y^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_y^\alpha u_z^\alpha\right) \\ \partial_x \left(\rho_\alpha u_z^\alpha u_x^\alpha\right) + \partial_y \left(\rho_\alpha u_z^\alpha u_y^\alpha\right) + \partial_z \left(\rho_\alpha u_z^\alpha u_z^\alpha\right) \end{pmatrix} + \begin{pmatrix} \partial_x \left(n_\alpha T_\alpha\right) \\ \partial_y \left(n_\alpha T_\alpha\right) \\ \partial_z \left(n_\alpha T_\alpha\right) \end{pmatrix} &= \left(\frac{L}{\delta_p}\right) \frac{Z_\alpha \rho_\alpha}{A_\alpha} \begin{pmatrix} 0 \\ u_z^\alpha B_x \\ -u_y^\alpha B_x \end{pmatrix}.
\end{aligned}$$

If $u_x^\alpha = u_z^\alpha = 0$ is assumed then the y equation is eliminated and only a z equation remains:

$$\partial_z \left(n_\alpha T_\alpha\right) = -\left(\frac{L}{\delta_p}\right)\frac{Z_\alpha \rho_\alpha}{A_\alpha} u_y^\alpha B_x.$$

This yields a constraint for each species:

$$\left(\frac{L}{\delta_p}\right) B_x = -\frac{\partial_z \left(n_\alpha T_\alpha\right)}{Z_\alpha n_\alpha u_y^\alpha} = -\frac{\partial_z \left(n_\beta T_\beta\right)}{Z_\beta n_\beta u_y^\beta}$$

with

$$n_\alpha = \frac{\rho_\alpha}{A_\alpha}.$$

Given the same density for both species initially as the profile for n suggests, with spatially constant temperature profiles this simplifies to

$$\frac{T_\alpha}{Z_\alpha u_y^\alpha} = \frac{T_\beta}{Z_\beta u_y^\beta}.$$

But this is not needed to find u_y^α:

$$\begin{aligned}
\partial_z \left(n_\alpha T_\alpha\right) &= -\left(\frac{L}{\delta_p}\right)\frac{Z_\alpha \rho_\alpha}{A_\alpha} u_y^\alpha B_x \\
\partial_z \left(n_\alpha T_\alpha\right) &= -\left(\frac{L}{\delta_p}\right) Z_\alpha n_\alpha u_y^\alpha B_x \\
u_y^\alpha &= -\,\partial_z \left(n_\alpha T_\alpha\right)\left(\frac{\delta_p}{L}\right)\frac{1}{Z_\alpha n_\alpha B_x} \\
&= -\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{B_x}\frac{\partial_z \left(n_\alpha\right)}{n_\alpha} \\
&= -\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{B_0 \tanh\left(\frac{z}{\lambda}\right)}\frac{\frac{-2}{\lambda} n_0 \tanh\left(\frac{z}{\lambda}\right)\operatorname{sech}^2\left(\frac{z}{\lambda}\right)}{\left(n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty\right)} \\
&= 2\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{B_0 \lambda}\frac{n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right)}{\left(n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty\right)} \\
&= \frac{2}{B_0 \lambda}\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right)}{\left(n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty\right)} \\
&= \frac{2}{B_0 \lambda}\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{\left(1 + \frac{n_\infty}{n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right)}\right)}
\end{aligned}$$

$$\boxed{u_y^\alpha = \frac{2}{B_0 \lambda}\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{\left(1 + \frac{n_\infty}{n_0}\cosh^2\left(\frac{z}{\lambda}\right)\right)}}.$$

The current can now be calculated and substituted into Ampere's law to set a condition for temperatures:

$$\begin{aligned}
\mathbf{J} =& \sum_{\alpha} Z_\alpha n_\alpha \mathbf{u}_\alpha \\
=& \sum_{\alpha} Z_\alpha n_\alpha u_y^\alpha \hat{y} \\
=& \sum_{\alpha} Z_\alpha \left(n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty \right) \left(\frac{2}{B_0 \lambda} \left(\frac{\delta_p}{L}\right) \frac{T_\alpha}{Z_\alpha} \frac{1}{\left(1 + \frac{n_\infty}{n_0} \cosh^2\left(\frac{z}{\lambda}\right)\right)} \right) \hat{y} \\
=& \frac{2}{B_0 \lambda} \left(\frac{\delta_p}{L}\right) \frac{\left(n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) + n_\infty\right)}{\left(1 + \frac{n_\infty}{n_0} \cosh^2\left(\frac{z}{\lambda}\right)\right)} \sum_{\alpha} T_\alpha \hat{y} \\
=& \frac{2}{B_0 \lambda} n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) \left(\frac{\delta_p}{L}\right) \frac{\left(1 + \frac{n_\infty}{n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right)}\right)}{\left(1 + \frac{n_\infty}{n_0} \cosh^2\left(\frac{z}{\lambda}\right)\right)} \sum_{\alpha} T_\alpha \hat{y} \\
=& \frac{2}{B_0 \lambda} n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) \left(\frac{\delta_p}{L}\right) \sum_{\alpha} T_\alpha \hat{y}.
\end{aligned}$$

And by Ampere's Law with no electric field, this yields:

$$\begin{aligned}
\nabla \times \mathbf{B} =& \left(\frac{L}{\delta_p}\right) \mathbf{J} \\
\partial_z B_x =& \left(\frac{L}{\delta_p}\right) \frac{2}{B_0 \lambda} n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) \left(\frac{\delta_p}{L}\right) \sum_{\alpha} T_\alpha \\
\frac{B_0}{\lambda} \operatorname{sech}^2\left(\frac{z}{\lambda}\right) =& \left(\frac{L}{\delta_p}\right) \frac{2}{B_0 \lambda} n_0 \operatorname{sech}^2\left(\frac{z}{\lambda}\right) \left(\frac{\delta_p}{L}\right) \sum_{\alpha} T_\alpha \\
B_0 =& \frac{2}{B_0} n_0 \sum_{\alpha} T_\alpha \\
\frac{B_0^2}{2 n_0} =& \sum_{\alpha} T_\alpha.
\end{aligned}$$

Specifically for two-fluid, this means

$$\frac{B_0^2}{2 n_0} = T_i + T_e.$$

Let $\theta = \frac{T_i}{T_e}$. Then,

$$\begin{aligned} \frac{B_0^2}{2n_0} =& \theta T_e + T_e \\ =& T_e\left(\theta + 1\right) \\ T_e =& \frac{B_0^2}{2n_0\left(\theta+1\right)}. \end{aligned} \tag{G.1.1}$$

G.2 WARPXM Geometry

Note that the Gem challenge geometry as given by Birn [8] is different from WARPXM. Compared to WARPXM:

Birn	WARPXM
X	X
Y	-Z
Z	Y

WARPXM	Birn
X	X
Y	Z
Z	-Y

This means for the initial setup:

$$n_{\mathrm{e}} = n_{\mathrm{i}} = n(y) = n_0 \operatorname{sech}^2\left(\frac{y}{\lambda}\right) + n_\infty, \tag{G.2.1}$$

$$\mathbf{u}_\alpha = u_z^\alpha \hat{z} = -\frac{2}{B_0\lambda}\left(\frac{\delta_p}{L}\right)\frac{T_\alpha}{Z_\alpha}\frac{1}{\left(1 + \frac{n_\infty}{n_0}\cosh^2\left(\frac{y}{\lambda}\right)\right)}\hat{z}, \tag{G.2.2}$$

where the temperature ratio (θ) is chosen and electron temperature is given from Eq. (G.1.1). Also

$$\boldsymbol{B} = B_x(y)\hat{x} + \frac{1}{10}\delta\boldsymbol{B}, \tag{G.2.3}$$

where

$$B_x(y) = B_0 \tanh\left(\frac{y}{\lambda}\right), \tag{G.2.4}$$

$$\delta\mathbf{B} = \psi_0\left[\frac{2\pi}{L_x}\sin\left(\frac{2\pi x}{L_x}\right)\cos\left(\frac{\pi y}{L_y}\right)\hat{y} - \frac{\pi}{L_y}\cos\left(\frac{2\pi x}{L_x}\right)\sin\left(\frac{\pi y}{L_y}\right)\hat{x}\right]. \tag{G.2.5}$$

Appendix H

RECASTING PRODUCTS OF BESSEL FUNCTIONS OF COMPLEX ORDER IN TERMS OF INTEGRALS OF BESSEL FUNCTIONS OF REAL ORDER

The electromagnetic susceptibility in Eq. (6.4.23) of the DGH instability resulted in an expression containing products of complex order Bessel functions which are difficult to treat numerically. However, by applying Neumann's formula [62, 43]

$$\begin{aligned} J_\nu\left(z\right)J_\mu\left(z\right) =& \frac{2}{\pi}\int\limits_0^{\frac{\pi}{2}} J_{\nu+\mu}\left(2z\cos\theta\right)\cos\left[\left(\mu-\nu\right)\theta\right]d\theta \\ \operatorname{Re}\left(\nu+\mu\right) >& -1, \end{aligned} \tag{H.0.1}$$

these products can be recast in terms of integrals of real order Bessel functions. Application of Eq. (H.0.1) on the product $J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right)$ leads to

$$J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right) = \frac{2}{\pi}\int\limits_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right)d\theta. \tag{H.0.2}$$

Differentiation of Eq. (H.0.2) yields

$$\begin{aligned} \frac{\partial}{\partial\beta_s}\left[J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right)\right] =& \frac{2}{\pi}\frac{\partial}{\partial\beta_s}\int\limits_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right)d\theta \\ =& \frac{2}{\pi}\int\limits_0^{\frac{\pi}{2}} \frac{\partial J_0\left(2\beta_s\cos\theta\right)}{\partial\left(2\beta_s\cos\theta\right)}\frac{\partial\left(2\beta_s\cos\theta\right)}{\partial\beta_s}\cos\left(2\alpha_s\theta\right)d\theta \\ =& \frac{2}{\pi}\int\limits_0^{\frac{\pi}{2}} \frac{\partial J_0\left(2\beta_s\cos\theta\right)}{\partial\left(2\beta_s\cos\theta\right)}2\cos\theta\cos\left(2\alpha_s\theta\right)d\theta. \end{aligned} \tag{H.0.3}$$

Application of the recurrence relation

$$J_{\nu-1}(z) - J_{\nu+1}(z) = 2\frac{\partial}{\partial z}\left[J_{\nu}(z)\right], \tag{H.0.4}$$

and identity for integer n [43]

$$J_{-n}(z) = (-1)^n J_n(z), \tag{H.0.5}$$

on the derivative term yields

$$\begin{aligned}\frac{\partial J_0\left(2\beta_s \cos\theta\right)}{\partial\left(2\beta_s \cos\theta\right)} &= \frac{1}{2}\left[J_{-1}\left(2\beta_s \cos\theta\right) - J_1\left(2\beta_s \cos\theta\right)\right] \\ &= \frac{1}{2}\left[-J_1\left(2\beta_s \cos\theta\right) - J_1\left(2\beta_s \cos\theta\right)\right] \\ &= -J_1\left(2\beta_s \cos\theta\right).\end{aligned} \tag{H.0.6}$$

Substitution of Eq. (H.0.6) into Eq. (H.0.3) results in

$$\frac{\partial}{\partial\beta_s}\left[J_{-\alpha_s}\left(\beta_s\right) J_{\alpha_s}\left(\beta_s\right)\right] = -\frac{4}{\pi}\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s \cos\theta\right)\cos\theta\cos\left(2\alpha_s\theta\right) d\theta. \tag{H.0.7}$$

$J'_{-\alpha_s}\left(\beta_s\right) J'_{\alpha_s}\left(\beta_s\right)$ can also be simplified, first by substitution of Eq. (H.0.4), yielding

$$\begin{aligned}J'_{-\alpha_s}\left(\beta_s\right) J'_{\alpha_s}\left(\beta_s\right) &= \frac{1}{2}\left[J_{-\alpha_s-1}\left(\beta_s\right) - J_{-\alpha_s+1}\left(\beta_s\right)\right]\frac{1}{2}\left[J_{\alpha_s-1}\left(\beta_s\right) - J_{\alpha_s+1}\left(\beta_s\right)\right] \\ &= \frac{1}{4}\left[\underbrace{J_{-\alpha_s-1}\left(\beta_s\right) J_{\alpha_s-1}\left(\beta_s\right)}_{A} - \underbrace{J_{-\alpha_s-1}\left(\beta_s\right) J_{\alpha_s+1}\left(\beta_s\right)}_{B}\right. \\ &\quad \left. - \underbrace{J_{-\alpha_s+1}\left(\beta_s\right) J_{\alpha_s-1}\left(\beta_s\right)}_{C} + \underbrace{J_{-\alpha_s+1}\left(\beta_s\right) J_{\alpha_s+1}\left(\beta_s\right)}_{D}\right],\end{aligned} \tag{H.0.8}$$

where the terms A-D have been defined for convenience and will be treated separately using Eq. (H.0.1). However for the term $A = J_{-\alpha_s-1}\left(\beta_s\right) J_{\alpha_s-1}\left(\beta_s\right)$, Eq. (H.0.1) cannot be used directly because $\operatorname{Re}\left(\nu+\mu\right) = -2 < -1$. This term can be rewritten using the recurrence relation for complex ν [43]

$$J_{\nu+1}\left(z\right) + J_{\nu-1}\left(z\right) = \frac{2\nu}{z} J_{\nu}\left(z\right), \tag{H.0.9}$$

yielding

$$\begin{aligned}A =&J_{-\alpha_s-1}\left(\beta_s\right)J_{\alpha_s-1}\left(\beta_s\right) = \left[\frac{-2\alpha_s}{\beta_s}J_{-\alpha_s}\left(\beta_s\right) - J_{-\alpha_s+1}\left(\beta_s\right)\right]\left[\frac{2\alpha_s}{\beta_s}J_{\alpha_s}\left(\beta_s\right) - J_{\alpha_s+1}\left(\beta_s\right)\right]\\ =&\frac{-4\alpha_s^2}{\beta_s^2}\underbrace{J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right)}_{E} + \frac{2\alpha_s}{\beta_s}\underbrace{J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s+1}\left(\beta_s\right)}_{F}\\ &-\frac{2\alpha_s}{\beta_s}\underbrace{J_{-\alpha_s+1}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right)}_{G} + \underbrace{J_{-\alpha_s+1}\left(\beta_s\right)J_{\alpha_s+1}\left(\beta_s\right)}_{D},\end{aligned} \tag{H.0.10}$$

where terms E-G have also been defined for convenience. Term E is given by Eq. (H.0.2). Application of Eq. (H.0.1) on the remaining terms yields

$$B =J_{-\alpha_s-1}\left(\beta_s\right)J_{\alpha_s+1}\left(\beta_s\right) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left[\left(2\alpha_s+2\right)\theta\right]d\theta, \tag{H.0.11}$$

$$C =J_{-\alpha_s+1}\left(\beta_s\right)J_{\alpha_s-1}\left(\beta_s\right) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left[\left(2\alpha_s-2\right)\theta\right]d\theta, \tag{H.0.12}$$

$$D =J_{-\alpha_s+1}\left(\beta_s\right)J_{\alpha_s+1}\left(\beta_s\right) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} J_2\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right)d\theta, \tag{H.0.13}$$

$$F =J_{-\alpha_s}\left(\beta_s\right)J_{\alpha_s+1}\left(\beta_s\right) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s\cos\theta\right)\cos\left[\left(2\alpha_s+1\right)\theta\right]d\theta, \tag{H.0.14}$$

$$G =J_{-\alpha_s+1}\left(\beta_s\right)J_{\alpha_s}\left(\beta_s\right) = \frac{2}{\pi}\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s\cos\theta\right)\cos\left[\left(2\alpha_s-1\right)\theta\right]d\theta. \tag{H.0.15}$$

Substitution of Eqs. (H.0.2) and (H.0.10)-(H.0.15) into Eq. (H.0.8) produces

$$\begin{aligned}
J'_{-\alpha_s}(\beta_s)\, J'_{\alpha_s}(\beta_s) =& \frac{1}{4}\left(A - B - C + D\right)\\
=& \frac{1}{4}\left[\frac{-4\alpha_s^2}{\beta_s^2}E + \frac{2\alpha_s}{\beta_s}F - \frac{2\alpha_s}{\beta_s}G + D - B - C + D\right]\\
=& \frac{1}{4}\left[\frac{-4\alpha_s^2}{\beta_s^2}E - B - C + \frac{2\alpha_s}{\beta_s}\left(F - G\right) + 2D\right]\\
=& \frac{1}{2\pi}\left[-\int_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\left[\frac{4\alpha_s^2}{\beta_s^2}\cos\left(2\alpha_s\theta\right) + \cos\left[\left(2\alpha_s+2\right)\theta\right] + \cos\left[\left(2\alpha_s-2\right)\theta\right]\right] d\theta\right.\\
&+ \frac{2\alpha_s}{\beta_s}\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s\cos\theta\right)\left\{\cos\left[\left(2\alpha_s+1\right)\theta\right] - \cos\left[\left(2\alpha_s-1\right)\theta\right]\right\} d\theta\\
&\left.+ 2\int_0^{\frac{\pi}{2}} J_2\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right) d\theta\right]. \qquad \text{(H.0.16)}
\end{aligned}$$

The integrals involving J_0 and J_1 can be further simplified by using the cosine sum trigonometric identity

$$\cos\left(a+b\right) = \cos a\cos b - \sin a\sin b, \qquad \text{(H.0.17)}$$

yielding

$$\begin{aligned}
J'_{-\alpha_s}(\beta_s)\, J'_{\alpha_s}(\beta_s) =& \frac{1}{\pi}\left[-\int_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right)\left[\frac{2\alpha_s^2}{\beta_s^2} + \cos\left(2\theta\right)\right] d\theta\right.\\
&- \frac{2\alpha_s}{\beta_s}\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s\cos\theta\right)\sin\theta\sin\left(2\alpha_s\theta\right) d\theta\\
&\left.+ \int_0^{\frac{\pi}{2}} J_2\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right) d\theta\right]. \qquad \text{(H.0.18)}
\end{aligned}$$

The integral involving J_1 can be removed by relating it to an integral involving J_0 by integrating Eq. (H.0.2) by parts and employing Eq. (H.0.6), yielding

$$\int_0^{\frac{\pi}{2}} J_1\left(2\beta_s\cos\theta\right)\sin\theta\sin\left(2\alpha_s\theta\right)d\theta = \frac{\sin\left(\pi\alpha_s\right)}{2\beta_s} - \frac{\alpha_s}{\beta_s}\int_0^{\frac{\pi}{2}} J_0\left(2\beta_s\cos\theta\right)\cos\left(2\alpha_s\theta\right)d\theta. \quad \text{(H.0.19)}$$

Substitution of Eq. (H.0.19) into Eq. (H.0.18) yields

$$J'_{-\alpha_s}\left(\beta_s\right)J'_{\alpha_s}\left(\beta_s\right) = \frac{1}{\pi}\left[\int_0^{\frac{\pi}{2}}\cos\left(2\alpha_s\theta\right)\left[J_2\left(2\beta_s\cos\theta\right) - J_0\left(2\beta_s\cos\theta\right)\cos\left(2\theta\right)\right]d\theta - \frac{\alpha_s}{\beta_s^2}\sin\left(\pi\alpha_s\right)\right]. \quad \text{(H.0.20)}$$

Equations (H.0.2), (H.0.7), and (H.0.20) recast the products of complex-order Bessel functions in the susceptibility tensor in Eq. (6.4.23) as integrals of real integer-order Bessel functions which can be evaluated using standard scientific computing packages. This allows for numerical evaluation of the electromagnetic dispersion relation for the DGH instability given in Eq. (6.4.21).

VITA

Iman Datta was born in India but grew up in Newfoundland and Texas. He earned a Bachelor's Degree in mechanical engineering from the University of Toronto. After graduation, his passion for aerospace led him to an internship at Nasa Ames Research Center, where he developed an interest in research before coming to the University of Washington. There, he earned his Master's and Doctorate degrees in Aeronautics & Astronautics, specializing in computational plasma physics.