Iterative and Analytic Poisson Solving Schemes for the Rosenbluth Potentials of the Fokker-Planck Collision Operator

Atmik Das

A thesis submitted in partial fulfillment of the

requirements for the degree of

Master of Science

University of Washington

2026

Committee:

Bhuvana Srinivasan

Jingwei Hu

Program Authorized to Offer Degree:

Applied Mathematics

University of Washington

Abstract

Iterative and Analytic Poisson Solving Schemes
for the Rosenbluth Potentials of the
Fokker-Planck Collision Operator

Atmik Das

Co-Chairs of the Supervisory Committee:

Bhuvana Srinivasan
Department of Aeronautics and Astronautics

Jingwei Hu
Department of Applied Mathematics

Continuum-kinetic models represent collisional plasma dynamics through distribution functions in phase space, capturing microscopic behavior statistically while avoiding the numerical noise of individual particle-tracking methods. The choice of collision operator plays a vital role here in determining what collisional mechanisms are included in the model. The nonlinear Rosenbluth/Fokker-Planck collision operator (FPO) provides a highly accurate treatment of small-angle Coulomb collisions, including a velocity-dependent effective collision frequency that correctly models the high-energy tails of the distribution. Its numerical implementation requires solving two coupled Poisson equations for the Rosenbluth potentials H and G over a three-dimensional velocity space per timestep and species, resulting in a highly-

dimensional formulation that is both computationally expensive and difficult to scale, making predictive simulation of collisions challenging within reasonable wall-clock time.

In this work, we employ the GKEYLL plasma simulation framework, which applies a discontinuous Galerkin (DG) finite element discretization to the full Vlasov-Maxwell-Fokker-Planck system, and focus specifically on extending the current DG-FPO implementation (MA-DG-FPO): it solves for the Rosenbluth potentials using a Maxwellian approximation, which fails to accurately simulate collisions for non-Maxwellian distributions arising in space, astrophysical, and laboratory plasmas. We study novel numerical and analytical techniques to solve for the Rosenbluth potentials, tested for accuracy, scalability, performance, and integrability with the existing FPO framework. We establish Algebraic Multigrid (AMG), implemented via the HYPRE preconditioner library, as a competitive benchmark, and develop a second-order Richardson iteration scheme which is tested against it, with Richardson as the more suitable numerical engine for our problem. To complement this, an analytic approach is studied using multipole expansions with symmetric trace-free (STF) tensors as a boundary correction to the far-field velocity domain, while Richardson addresses the potential solves in the interior. Together, they form the Extended FPO (X-DG-FPO), integrated into the GKEYLL Vlasov layer to accurately and efficiently simulate collisions across timesteps.

The performance and accuracy of the X-DG-FPO is tested against established collision operators, namely the Bhatnagar-Gross-Krook operator (BGK), the Dougherty-Lenard-Bernstein operator (LBO), and the MA-DG-FPO, using relaxation and thermalization tests on non-Maxwellian distributions. The headline result is that the MA-DG-FPO produces severe spurious oscillations for distributions with heavy tails, such as kappa distributions, while the X-DG-FPO eliminates them completely — demonstrating the importance of accurate handling of the velocity-dependent col-

lision frequency for non-Maxwellian populations. This work motivates continued development towards an efficient, fully conservative, multispecies generalization of the nonlinear Rosenbluth/Fokker-Planck collision operator within the continuum-kinetic framework.

Acknowledgments

This thesis is the product of two years of work that spanned two departments, two primary fields of study, and countless hours of debugging and derivation. None of it would have been possible without the support of many people, and I am grateful for the opportunity to thank them here.

First and foremost, I want to thank my advisor, Prof. Bhuvana Srinivasan. I started this work with a background in applied mathematics and only a surface-level knowledge of a few concepts in plasma physics. It was her constant patience, enthusiasm, and mentorship that made it possible for me to push and strive towards applying my mathematical intuition to complex plasma physics problems. More than the technical knowledge, she has instilled in me the quality of being inquisitive at every step of research, be it in trying to optimize an algorithm or examining an edge-case physics scenario. I am deeply grateful to be continuing my doctoral work under her guidance, and to remain a part of the highly collaborative and positive research environment she has built.

I am also grateful to my co-advisor, Prof. Jingwei Hu. Her guidance in establishing the collaboration with Prof. Srinivasan and in shaping the structure and pacing of this work through regular meetings and presentation opportunities helped ensure that I stayed dedicated and motivated throughout.

This thesis builds directly and substantially on the foundational work of Dr. John Rodman, whose discontinuous Galerkin Fokker-Planck implementation in GKEYLL

is the starting point for everything in this thesis. Beyond the code itself, John was able to help me get started with this research and helped make the ideas and intuition less intimidating. Without his generosity with his time and knowledge, this thesis would not exist in its current form.

I owe a great deal to the GKEYLL folks at Princeton Plasma Physics Laboratory (PPPL). Dr. Ammar Hakim, the chief algorithm alchemist of GKEYLL, played a pivotal role in explaining novel mathematical techniques in plasma physics and his approach to algorithm building is something I keep wanting to emulate. Dr. James Juno also helped me get started on my first few tests and research goals in this project. GKEYLL is a highly efficient and powerful simulation framework and yet it can be very daunting at first, and his guidance early on was invaluable. Dr. Manaure Francisquez provided crucial direction in the final stages of this work, particularly in thinking clearly about research goals and thesis structure. I thank them all for their generosity with their time and expertise.

The algebraic multigrid component of this work would not have been possible without the guidance of Dr. Robert Falgout at Lawrence Livermore National Laboratory (LLNL), who provided essential support in working with the HYPRE library and in thinking through multigrid methods for this problem. I also thank the HYPRE and GKEYLL development teams more broadly for building the open-source infrastructure that this work depends on.

I am grateful to the PLASMAWISE Lab for providing not just a research environment but a genuinely collaborative and supportive community that I look forward to continuing to be a part of. In particular, I want to thank Vignesh Krishna Kumar, Lucio Murillo, Asher Beck, Jeff Contri, Daniel Alex, and Maddie Loebel, whose guidance, support, and advice as senior graduate students and friends helped me navigate the challenges of research and graduate life, and made the lab a very warm and welcoming environment to work in; Jonathan Li, whom I have had the pleasure

of mentoring; and Howard Cheng, whose early engagement pointed me towards Prof. Srinivasan's and Prof. Hu's work and motivated me to join PLASMAWISE.

I am grateful to the Department of Applied Mathematics and the Department of Aeronautics and Astronautics at the University of Washington (UW) for providing the academic home for this interdisciplinary work. The computational resources provided by the HYAK computing cluster at UW were essential for experimentation and validation in this thesis.

Finally, I want to thank my family and friends for everything that does not fit neatly into a research acknowledgment. To my parents — you have been my rock always. All of this is for you.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Plasma — the fourth state of matter — is the most abundant form of visible matter in the observable universe. From the interior of stars and the solar wind to fusion reactors and laboratory discharges, plasma exists across an extraordinary range of scales and conditions. When describing the nature of plasma, the easiest explanation is with respect to the other states of matter. When a gas is heated sufficiently, collisions between constituent particles become energetic enough to strip electrons from their host nuclei, producing a partially or fully ionized medium of free electrons and ions, known as plasma.

However, to distinguish it from a weakly ionized gas, the most accurate definition of plasmas, given by Chen [1]: *"a plasma is a quasineutral gas of charged and neutral particles which exhibits collective behavior." Quasineutrality* means that the free electron and ion number densities (number of particles per unit volume) are approximately equal on macroscopic scales, that is,

$$n_e \approx Z n_i, \tag{1.1}$$

where n_e and n_i are the electron and ion number densities, respectively, and Z is the ion charge state. *Collective behavior* refers to the fact that the long-range Coulomb

interaction couples distant regions of the plasma — the dynamics of any small volume are influenced not only by local conditions but by the state of the plasma far away.

The inherently complex interactions between the enormous number of particles in a plasma — on the order of 10^{20} particles per cubic meter in a tokamak fusion reactor — and the rapidly fluctuating electromagnetic fields they generate make accurate modeling of important physical phenomena, such as collisions, a formidable challenge [2]. However, with carefully developed mathematical tools and physics-informed modeling, it is possible to develop tractable descriptions that capture the essential physics accurately. Such models serve a dual purpose: they advance theoretical understanding of complex plasma behavior, and they directly inform experimental design in applications ranging from space weather prediction to inertial confinement fusion (ICF). Computational plasma physics, in particular, has emerged as an indispensable field of study for exploring regimes that are inaccessible to purely analytical approaches or too costly and dangerous to probe experimentally.

The overarching goal motivating this work is the development of a momentum- and energy-conserving algorithm to solve a continuum-kinetic model of the collisional multispecies nonlinear Rosenbluth/Fokker-Planck equations. This thesis represents a step toward that goal, focusing specifically on extending the existing discontinuous Galerkin (DG) Fokker-Planck implementation in the GKEYLL simulation framework to handle arbitrary, non-Maxwellian distribution functions through novel iterative and analytic Poisson solving schemes for the Rosenbluth potentials [3, 4].

This chapter establishes the physical foundation for our research. We begin with a discussion on the various plasma modeling techniques, motivating the continuum-kinetic approach and introducing the Vlasov-Boltzmann equation as our primary

governing system. We then move to survey various collision operators, establishing the physical foundation and advantages behind choosing the Rosenbluth/Fokker-Planck collision operator (FPO) — the crux of our research. Finally, we end this chapter covering the suite of distribution functions that this thesis aims to study, their relevance in various plasma physics regimes, and the mathematical challenges one expects to face while working with them.

Chapter 2 establishes the mathematical infrastructure underlying the implementation. We begin by introducing the DG discretization as implemented in GKEYLL, with emphasis on some preliminary pieces used to evaluate derivatives of the Rosenbluth potentials at cell boundaries. We then frame the coupled Rosenbluth potential Poisson system as a computational problem and motivate two families of solution methods, iterative and analytic, developed in subsequent chapters. Finally, we present the existing DG-FPO algorithm in full and identify precisely where the contributions of this thesis extend it.

Chapter 3 evaluates algebraic multigrid (AMG) as a benchmark solver for the coupled Rosenbluth potential system. We implement an AMG solver using the HYPRE preconditioner library on a finite difference velocity grid, benchmark its convergence and parallel performance against the existing DG-FPO's analytic projection scheme, and identify the structural limitations that make AMG unsuitable for implementation within GKEYLL's DG infrastructure.

Chapter 4 develops a second-order Richardson iteration as a practical Poisson solver for the coupled Rosenbluth potential system within GKEYLL's DG infrastructure. The scheme evolves a damped wave equation in pseudo-time to steady state, from which the Richardson update, stability condition, and optimal damping parameter are derived. A standalone solver is constructed using this framework that is compared to the AMG benchmarks and Rodman's analytic projection results in Chapter 3. In direct performance comparison, the Richardson solver is found to

be a suitable piece to be implemented within the DG-FPO algorithm.

Chapter 5 develops an analytic representation for the Rosenbluth potentials using a multipole expansion in terms of Cartesian velocity moments of f_s. This allows for an exact formalism of the quantities using pre-computed moments in the DG-FPO algorithm. This is followed by a study on the optimal application of these expansions in the velocity domain and is applied to the DG-FPO. The effects of this are studied and validated using relaxation tests and conservation studies across the suite of non-Maxwellian distributions.

Chapter 6 integrates the Richardson iteration and multipole expansion into the existing DG-FPO algorithm in GKEYLL, producing the Extended DG-FPO. Relaxation tests across all four non-Maxwellian distributions and a thermalization test on a bi-Maxwellian initial condition validate the extended algorithm, exposing both its capabilities and limitations.

Chapter 7 summarizes the contributions of this thesis and outlines directions for future work. The chapter traces the development of the Extended DG-FPO from the Rosenbluth potential Poisson solves through the Richardson iteration, multipole expansion, and their integration into GKEYLL, and concludes with short- and long-term directions of research towards a fully conservative multispecies nonlinear Rosenbluth/Fokker-Planck algorithm.

1.1 The Landscape of Plasma Modeling

Accurately modeling a plasma requires choosing a mathematical framework that captures the physical scales and phenomena relevant to the problem at hand. No single model is universally optimal. Each of them occupies a distinct position along the spectrum from simplicity to physical fidelity, and the appropriate choice depends on which effects matter for the regime of interest. The central challenge is

that plasma dynamics span an enormous range of length and time scales, from the microscopic motion of individual particles to large-scale macroscopic flows, and no tractable model can resolve all of them simultaneously. We briefly survey the main approaches here, motivating the continuum-kinetic model that is the framework of this thesis.

Magnetohydrodynamics (MHD). The computationally simplest class of plasma models treats the plasma as one or more continuous conducting fluids, governed by the Navier-Stokes equations coupled to Maxwell's equations via the Lorentz force. In its single-fluid form, ions carry the mass and electrons carry the current [2]. MHD has been successfully applied to large-scale phenomena such as solar wind dynamics, stellar formation, and magnetic reconnection, and remains the main engine of astrophysical plasma modeling. However, its simplicity in terms of viewing the plasma macroscopically as a fluid comes at a cost: microscopic phenomena such as kinetic effects, plasma waves, velocity-space structure, and non-equilibrium behavior are all averaged away, making MHD unsuitable for regimes where these effects are important.

Two-Fluid Models. A natural extension of MHD, this approach treats ions and electrons as two separate interpenetrating fluids, each with its own density, velocity, and pressure evolution equations [2]. This recovers physics inaccessible to single-fluid MHD, such as drift waves and two-stream instabilities, but still assumes near-local thermodynamic equilibrium and loses large-amplitude, non-equilibrium effects. Like MHD, this model also views the plasma's macroscopic properties. This makes it tractable analytically but fundamentally limited in the velocity-space physics they can represent.

Gyrokinetics. In strongly magnetized plasmas, the fast cyclotron motion of particles around magnetic field lines can be averaged over, reducing the six-dimensional kinetic problem to five dimensions [5]. The resulting gyrokinetic model is the

method of choice for studying turbulence and transport in tokamaks, where the separation of gyration and drift timescales is well-justified. However, this averaging introduces non-physical behavior over long timescales and breaks down when resonances and adiabatic invariants are important.

Fully Kinetic Models. At the highest level of physical fidelity (short of tracking every individual particle exactly), sit fully kinetic models, which describe the plasma through the distribution function $f_s(\mathbf{x}, \mathbf{v}, t)$: the number density of species s in the six-dimensional phase space, comprising of three position $\mathbf{x}$ dimensions and three velocity $\mathbf{v}$ dimensions at time t. Kinetic models retain the full velocity-space structure of the plasma, making them capable of capturing runaway electron dynamics, high-energy tail physics, collisional transport, and velocity-space instabilities. The governing mathematical system is the Vlasov-Boltzmann equation, introduced in Section 1.1.2. The price of this fidelity is computational: the six-dimensional phase space makes kinetic simulations substantially more expensive than fluid simulations, requiring fast and scalable numerical methods in order to tackle them. It is precisely this challenge that motivates the work presented in this thesis.

To present a complete picture, at the theoretical extreme sits the Klimontovich description, which is the most complete kinetic model possible. It tracks every individual particle exactly by representing the microscopic phase space density as a sum of delta functions,

$$N_s(\mathbf{x}, \mathbf{v}, t) = \sum_{i=1}^{N_T} \delta[\mathbf{x} - \mathbf{X}_i(t)]\delta[\mathbf{v} - \mathbf{V}_i(t)], \tag{1.2}$$

whose evolution is governed by the Klimontovich equation [6],

$$\frac{\partial N_s}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} N_s + \frac{q_s}{m_s} \left(\mathbf{E}^m + \mathbf{v} \times \mathbf{B}^m\right) \cdot \nabla_{\mathbf{v}} N_s = 0, \tag{1.3}$$

where $\mathbf{E}^m$ and $\mathbf{B}^m$ are the exact microscopic electromagnetic fields. While Eq. (1.3) is

theoretically exact, it is practically intractable: resolving every particle in a system of interest, such as fusion plasmas which contain on the order of 10^{20} particles per cubic meter is simply unfeasible, and the level of detail it provides far exceeds what is needed for most physical questions of interest. We instead seek a model that averages out individual particle information while retaining the collective effects that arise from particle motion — as provided by the Vlasov-Boltzmann equation.

1.1.1 Plasma Kinetic Models

Within the class of fully kinetic models, two broad and popular numerical approaches exist for evolving the distribution function: particle-in-cell (PIC) methods and continuum-kinetic methods.

Particle-In-Cell (PIC). PIC methods represent the distribution function as a collection of macro-particles, each representing many physical particles, which are pushed through phase space under self-consistently computed electromagnetic fields [7]. PIC is conceptually straightforward and has been applied successfully to a wide range of plasma physics problems. However, it suffers from statistical sampling noise that grows with decreasing particle count, requiring large numbers of macro-particles to achieve low noise levels. Accurately resolving collisional physics in PIC requires additional Monte Carlo procedures that introduce further approximation and computational expense.

Continuum-Kinetic Methods. Rather than sampling the distribution function with particles, continuum-kinetic methods discretize f_s directly on a fixed grid in phase space and evolve it as a continuous field. This eliminates statistical sampling noise entirely, enables direct and systematic incorporation of collisions, conservation laws, and boundary conditions, and produces smooth, high-order-accurate solutions. The tradeoff is the memory and computational cost of storing and evolving a high-dimensional field, which scales steeply with resolution and grid sizes. However,

advances in high-order numerical methods on modern GPU architectures have made continuum-kinetic simulations increasingly practical. VLASIATOR [8] and GKEYLL [3, 4] are examples of continuum-kinetic codes that demonstrate this approach can now be applied to physically meaningful problems at previously inaccessible scales.

In this work, we employ the continuum-kinetic approach using the GKEYLL simulation framework, which applies a high-order DG finite element discretization to the Vlasov-Maxwell-Fokker-Planck system. The matrix-free structure of the DG method in GKEYLL enables efficient parallelization on GPU architectures, making it particularly well-suited to the high-dimensional phase space problems of kinetic plasma simulation [3, 4].

1.1.2 Vlasov-Boltzmann Equation

The governing equation of the continuum-kinetic model is the Vlasov-Boltzmann equation [1, 2], which describes the evolution of the distribution function $f_s(\mathbf{x}, \mathbf{v}, t)$ for each plasma species s in phase space. The distribution function evolves due to three contributions: streaming of particles through configuration space, acceleration by electromagnetic forces in velocity space, and collisional interactions between particles. Together these yield,

$$\frac{\partial f_s}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f_s + \frac{q_s}{m_s} (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f_s = C[f_s], \tag{1.4}$$

where q_s and m_s are the charge and mass of species s, $\mathbf{E}$ and $\mathbf{B}$ are the macroscopic electric and magnetic fields, and $C[f_s]$ is the collision operator encoding the effect of inter-particle collisions on f_s. The left-hand side of Eq. (1.4) describes collisionless advection of f_s through phase space under the Lorentz force. When collisional effects are negligible, as is often the case for diffuse high-temperature plasmas, the

right-hand side vanishes and we recover the collisionless Vlasov equation. When collisions are important, as they are in the regimes this thesis addresses, the choice of collision operator $C[f_s]$ becomes critical, and is the subject of Section 1.2.

The electromagnetic fields are self-consistently determined by Maxwell's equations,

$$\nabla_{\mathbf{x}} \cdot \mathbf{E} = \frac{\rho}{\epsilon_0}, \tag{1.5}$$

$$\nabla_{\mathbf{x}} \cdot \mathbf{B} = 0, \tag{1.6}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla_{\mathbf{x}} \times \mathbf{E} = 0, \tag{1.7}$$

$$\nabla_{\mathbf{x}} \times \mathbf{B} - \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} = \mu_0 \mathbf{J}. \tag{1.8}$$

The source terms ρ and $\mathbf{J}$ couple Maxwell's equations back to the distribution function through velocity moments,

$$\rho(\mathbf{x}, t) = \sum_s q_s \int f_s \, d^3v, \tag{1.9}$$

$$\mathbf{J}(\mathbf{x}, t) = \sum_s q_s \int \mathbf{v} f_s \, d^3v, \tag{1.10}$$

Together, Eq. (1.4) and Maxwell's equations, closed by the moment relations, form the Vlasov-Maxwell-Boltzmann system — the governing system for the continuum-kinetic model implemented in the Vlasov layer in GKEYLL.

The macroscopic bulk properties of the plasma are recovered by taking velocity moments of f_s [2]. The zeroth moment gives the number density,

$$n_s(\mathbf{x}, t) = \int f_s(\mathbf{x}, \mathbf{v}, t) \, d^3v, \tag{1.11}$$

the first moment gives the bulk momentum density,

$$n_s(\mathbf{x}, t)\mathbf{u}_s(\mathbf{x}, t) = \int \mathbf{v} f_s(\mathbf{x}, \mathbf{v}, t) \, d^3v, \tag{1.12}$$

and the second moment gives the energy density,

$$\mathcal{E}_s(\mathbf{x}, t) = \frac{1}{2} m_s \int v^2 f_s(\mathbf{x}, \mathbf{v}, t)\, d^3v. \tag{1.13}$$

These moments connect the kinetic description to the macroscopic fluid quantities of simpler plasma models. Crucially, the collision operator $C[f_s]$ must conserve these moments to ensure physical consistency.

1.2 The Choice of Collision Operator

As mentioned previously in Section 1.1.2, the collision operator $C[f_s]$ on the right-hand side of the Vlasov-Boltzmann equation encodes the physical mechanisms by which inter-particle collisions redistribute momentum and energy across the distribution function. Through many small-angle Coulomb interactions, this redistribution gradually smooths out, driving the system toward a state of thermal equilibrium. Any physically consistent collision operator must accurately reflect this behavior: it must conserve particle number density, momentum, and energy, and must drive f_s monotonically toward this equilibrium state. The choice of collision operator determines not only whether these fundamental properties are satisfied, but also how accurately the velocity-dependent nature of Coulomb collisions is captured per timestep. In this section, we survey three such collision operators in increasing physical fidelity: the Bhatnagar-Gross-Krook (BGK) operator, the Dougherty-Lenard-Bernstein (LBO) operator, and the Fokker-Planck operator (FPO).

1.2.1 Bhatnagar-Gross-Krook Operator

The simplest collision operator considered here is the Bhatnagar-Gross-Krook (BGK) operator [9], which bypasses the detailed mechanics of individual collisions and directly relaxes the distribution function toward thermal equilibrium at a rate proportional to the collision frequency,

$$C_{\mathrm{BGK}}[f_s] = \nu_{ss}(f_{M,s} - f_s), \tag{1.14}$$

where ν_{ss} is an average collision frequency and $f_{M,s}$ is the equilibrium distribution constructed from the velocity moments of f_s. The BGK operator conserves particle number density, momentum, and energy and relaxes to thermal equilibrium. This follows directly from the fact that $f_{M,s}$ is constructed using the same moments as f_s, so the first three velocity moments of C_{BGK} vanish identically.

However, the BGK operator has significant physical limitations for plasma applications. It is designed for large-angle binary collisions between neutral particles, rather than the small-angle Coulomb collisions that dominate in a plasma. The use of a single, velocity-independent collision frequency ν_{ss} means that all particles, regardless of their speed, are relaxed at the same rate. In reality, the effective Coulomb collision frequency scales as v^{-3}, meaning fast particles collide far less frequently than slow ones. As a result, the BGK operator substantially overestimates the effect of collisions in the high-energy tails of the distribution function. It is therefore best suited to regimes where departures from equilibrium are small and occur near the bulk of the distribution [9, 10].

1.2.2 Dougherty-Lenard-Bernstein Operator

A more physically motivated collision operator for plasmas is the Dougherty-Lenard-Bernstein operator (LBO) [11, 12], which models collisions through velocity-

space drag and diffusion acting on the distribution function. The LBO takes the form,

$$C_{\mathrm{LBO}}[f_s] = \nu_{ss}\nabla_{\mathbf{v}} \cdot \left[(\mathbf{v} - \mathbf{u}_s) f_s + \frac{T_s}{m_s} \nabla_{\mathbf{v}} f_s \right], \tag{1.15}$$

where ν_{ss} is an average collision frequency, and $\mathbf{u}_s$ and T_s are the bulk velocity and temperature computed from velocity moments of f_s. The drag term pulls the distribution toward the bulk flow velocity, while the diffusion term spreads it toward thermal equilibrium. The LBO conserves particle number density, momentum, and energy.

The key limitation and simplification used in the LBO is the use of a single, velocity-independent collision frequency ν_{ss}. All particles, regardless of speed, experience the same collision rate. As a result, the LBO substantially overestimates the effect of collisions in the high-energy tails of the distribution function. While this is acceptable for near-equilibrium plasma dynamics, it fails for distributions with significant non-equilibrium structure. A physically accurate treatment of these cases requires a velocity-dependent collision operator.

1.2.3 Fokker-Planck Operator

The most physically accurate collision operator for a plasma is the nonlinear Fokker-Planck operator (FPO), which describes the cumulative effect of many small-angle Coulomb collisions on the distribution function through velocity-space drag and diffusion with fully velocity-dependent coefficients. Unlike the BGK and LBO operators, the drag and diffusion coefficients are functions of velocity, computed directly from the distribution function itself, correctly capturing the v^{-3} scaling of the Coulomb collision frequency.

Figure 1.1 illustrates this distinction concretely. For a bump-on-tail distribution evolved to $t = 0.6/\nu$, the LBO drives the high-energy tail to equilibrium, over-

relaxing it relative to the physically accurate FPO result. The FPO correctly preserves the tail structure, demonstrating the critical importance of a velocity-dependent collision frequency for non-equilibrium distributions.

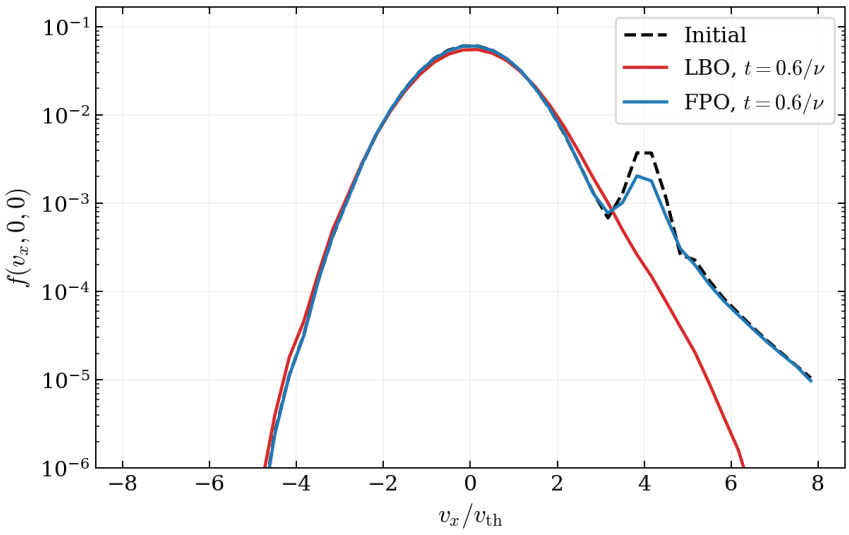

Figure 1.1: Relaxation of a bump-on-tail distribution at $t = 0.6/\nu$ via the FPO and LBO.

The FPO was originally formulated by Landau [13] as a nonlinear integro-differential equation [14],

$$C_{\mathrm{FPO}}[f_s, f_{s'}] = -\frac{m_s \Gamma_{ss'}}{2} \nabla_{\mathbf{v}} \cdot \int \mathbf{w} \cdot \left[\frac{f_s(\mathbf{v})}{m_{s'}} \nabla_{\mathbf{v}'} f_{s'}(\mathbf{v}') - \frac{f_{s'}(\mathbf{v}')}{m_s} \nabla_{\mathbf{v}} f_s(\mathbf{v}) \right] d^3 v', \quad (1.16)$$

where $\Gamma_{ss'} = 4\pi \lambda_{ss'} q_s^2 q_{s'}^2 / m_s^2$ is the Coulomb logarithm prefactor with $\lambda_{ss'}$ the Coulomb logarithm, and $\mathbf{w} = (\mathbf{I} - \hat{\mathbf{u}}_{ss'} \hat{\mathbf{u}}_{ss'})/u_{ss'}$ is the deflection tensor with $\mathbf{u}_{ss'} = \mathbf{v} - \mathbf{v}'$ the relative velocity between colliding particles. While physically exact, the Landau form requires evaluating a six-dimensional integral over velocity space at every point and at every timestep — a computational cost that makes direct numerical implementation impractical.

Rosenbluth, MacDonald, and Judd reformulated the Landau operator into a significantly more tractable form by introducing two scalar potentials H_s and G_s, which

satisfy coupled Poisson equations in velocity space [10]. This makes the Rosenbluth form of the FPO amenable to numerical implementation while retaining its full physical accuracy. The Rosenbluth form is the basis for the GKEYLL implementation and is the form adopted exclusively throughout this thesis.

1.3 Multispecies Rosenbluth/Fokker-Planck Equations

The Rosenbluth formulation recasts the Landau operator Eq. (1.16) into an advection-diffusion equation whose coefficients are determined by two scalar potentials satisfying coupled Poisson equations in velocity space. The full derivation can be found in [15]; here we present the working form directly. The multispecies FPO for species s colliding with all field species s' takes the form [10, 16]

$$\left.\frac{\partial f_s}{\partial t}\right|_{\text{coll}} = -\nabla_{\mathbf{v}} \cdot \left(\mathbf{a}_s f_s - \frac{1}{2} \nabla_{\mathbf{v}} \cdot (\mathbf{D}_s f_s) \right), \tag{1.17}$$

where the drag vector $\mathbf{a}_s$ and diffusion tensor $\mathbf{D}_s$ are given by

$$\mathbf{a}_s = \nabla_{\mathbf{v}} h_s, \tag{1.18}$$

$$\mathbf{D}_s = \nabla_{\mathbf{v}} \nabla_{\mathbf{v}} g_s, \tag{1.19}$$

with the composite potentials

$$h_s(\mathbf{v}) = \sum_{s'} \Gamma_{ss'} \left(1 + \frac{m_s}{m_{s'}} \right) H_{s'}(\mathbf{v}), \tag{1.20}$$

$$g_s(\mathbf{v}) = \sum_{s'} \Gamma_{ss'} \, G_{s'}(\mathbf{v}), \tag{1.21}$$

where $\Gamma_{ss'} = 4\pi\lambda_{ss'} q_s^2 q_{s'}^2 / m_s^2$ is the Coulomb logarithm prefactor and $\lambda_{ss'}$ is the Coulomb logarithm [16]. The quantities $H_{s'}(\mathbf{v})$ and $G_{s'}(\mathbf{v})$ are the *Rosenbluth poten-*

tials for field species s', defined by the velocity-space convolution integrals [10],

$$H_{s'}(\mathbf{v}) = \int \frac{f_{s'}(\mathbf{v}')}{|\mathbf{v}-\mathbf{v}'|}\, d^3v', \tag{1.22}$$

$$G_{s'}(\mathbf{v}) = \int f_{s'}(\mathbf{v}')|\mathbf{v}-\mathbf{v}'|\, d^3v'. \tag{1.23}$$

The Rosenbluth potentials are scalar potentials in velocity space that bear a formal similarity to potential theory, with $f_{s'}$ playing the role of a source density [10]. Their gradients yield the drag and diffusion coefficients that drive f_s toward equilibrium. Plots of H_s and G_s and their associated drag and diffusion coefficients for a Maxwellian distribution are shown in Appendix A.

The potentials can also be represented in differential form, where they satisfy Poisson equations in velocity space,

$$\nabla^2_{\mathbf{v}} H_{s'} = -4\pi f_{s'}, \tag{1.24}$$

$$\nabla^2_{\mathbf{v}} G_{s'} = 2H_{s'}, \tag{1.25}$$

which follow from the identity $\nabla^2_{\mathbf{v}}|\mathbf{v}-\mathbf{v}'|^{-1} = -4\pi\delta^3(\mathbf{v}-\mathbf{v}')$ and the relation $\nabla^2_{\mathbf{v}}|\mathbf{v}| = 2/|\mathbf{v}|$ [16]. The primary contribution of this thesis is solving Equations (1.24) and (1.25) using novel numerical and analytic techniques described in Chapters 3, 4, and 5, and then constructing the drag and diffusion coefficients from Eqs. (1.18)–(1.21).

The mass ratio factor $(1 + m_s/m_{s'})$ in h_s encodes the asymmetry of momentum transfer between particles of different masses — an effect absent in both the BGK and LBO operators. Since $\mathbf{a}_s$ depends on $H_{s'}$ and $\mathbf{D}_s$ depends on $G_{s'}$, both of which are functionals of $f_{s'}$, the collision rates emerge directly from the distribution function itself with no free frequency parameter.

Two identities follow from Eqs. (1.24)–(1.25) [16],

$$\mathrm{Tr}(\mathbf{D}_s) = \nabla^2_{\mathbf{v}} g_s = 2\sum_{s'} \Gamma_{ss'} H_{s'}, \tag{1.26}$$

$$\nabla_{\mathbf{v}} \cdot \mathbf{D}_s = \nabla_{\mathbf{v}} \nabla^2_{\mathbf{v}} g_s = 2\sum_{s'} \Gamma_{ss'} \frac{m_s}{m_{s'}} \nabla_{\mathbf{v}} H_{s'}, \tag{1.27}$$

which allow Eq. (1.17) to be rewritten in the equivalent form

$$\left.\frac{\partial f_s}{\partial t}\right|_{\mathrm{coll}} = -\frac{1}{2}\nabla_{\mathbf{v}} \cdot \left(\mathbf{a}'_s f_s - \mathbf{D}_s \cdot \nabla_{\mathbf{v}} f_s\right), \tag{1.28}$$

where $\mathbf{a}'_s = 2\sum_{s'} \Gamma_{ss'}(m_s/m_{s'})\nabla_{\mathbf{v}} H_{s'}$. In this form, the competing roles of drag and diffusion are explicit, and it is this form that is implemented in GKEYLL [3, 4]. While the multispecies form presented here represents the full generality of the FPO, the implementation developed in this thesis operates on a single species. This restriction is deliberate: the focus of this work is the development and validation of numerical Poisson solvers for H_s and G_s, which is the computational bottleneck blocking a production multispecies implementation. In the single-species case, where $s' = s$, the composite potentials reduce to

$$h_s(\mathbf{v}) = 2\Gamma_{ss} H_s(\mathbf{v}), \tag{1.29}$$

$$g_s(\mathbf{v}) = \Gamma_{ss} G_s(\mathbf{v}), \tag{1.30}$$

and the Poisson equations (1.24)–(1.25) are solved for H_s and G_s driven by f_s alone. This is an integral piece of creating a generalised multispecies FPO, as the equations for each field species s' are structurally identical.

1.3.1 Conservation Properties of the FPO

A physically consistent collision operator must conserve the three velocity moments established in Section 1.1.2: particle number density, momentum, and energy. We verify each in turn [15].

Conservation of number density. For the FPO to conserve particles, the velocity integral of the collision operator must vanish,

$$\int C_{ss'}[f_s, f_{s'}]\, d^3v = 0. \tag{1.31}$$

Substituting the Landau form and applying the divergence theorem in velocity space, the integral reduces to a surface term at $|\mathbf{v}| \to \infty$. Since f_s and $\nabla_{\mathbf{v}} f_s$ vanish as $|\mathbf{v}| \to \infty$ for any physical distribution function, this surface term is zero and conservation of number density follows [15].

Conservation of momentum. In collisions between species s and s', the momentum lost by one species must be gained by the other,

$$\mathbf{F}_{ss'} + \mathbf{F}_{s's} = \int m_s \mathbf{v}\, C_{ss'}[f_s]\, d^3v + \int m_{s'} \mathbf{v}\, C_{s's}[f_{s'}]\, d^3v = 0, \tag{1.32}$$

where $\mathbf{F}_{ss'}$ is the momentum exchange rate from species s' to species s. Integrating by parts and exploiting the antisymmetry of the integrand under simultaneous exchange of dummy variables $\mathbf{v} \leftrightarrow \mathbf{v}'$ and species labels $s \leftrightarrow s'$, one finds $\mathbf{F}_{ss'} = -\mathbf{F}_{s's}$, satisfying Eq. (1.32) [15].

Conservation of energy. The total rate of energy exchange between species s and s' must similarly vanish,

$$\mathcal{E}_{ss'} + \mathcal{E}_{s's} = \frac{1}{2} m_s \int v^2\, C_{ss'}[f_s]\, d^3v + \frac{1}{2} m_{s'} \int v^2\, C_{s's}[f_{s'}]\, d^3v = 0, \tag{1.33}$$

where $\mathcal{E}_{ss'}$ is the energy exchange rate from species s' to species s. Applying the same

species-swap argument and using the fact that $\mathbf{u}_{ss'} \cdot \mathbf{w} \cdot \mathbf{u}_{ss'} = 0$ for the deflection tensor $\mathbf{w}$ — since $\mathbf{w}$ projects perpendicular to the relative velocity $\mathbf{u}_{ss'} = \mathbf{v} - \mathbf{v}'$ — one finds $\mathcal{E}_{ss'} = -\mathcal{E}_{s's}$, satisfying Eq. (1.33) [15].

In the single-species case, $s' = s$ and the exchange argument reduces to the requirement that the FPO conserves momentum and energy within the species,

$$\int m_s \mathbf{v}\, C_{ss}[f_s]\, d^3v = 0, \tag{1.34}$$

$$\int \frac{1}{2} m_s v^2\, C_{ss}[f_s]\, d^3v = 0, \tag{1.35}$$

which follow from the same antisymmetry argument with $s' = s$. These are the conservation constraints that the GKEYLL implementation must satisfy discretely, and they remain a target for the final algorithm developed in this thesis.

1.3.2 Boltzmann's H-Theorem for the FPO

Section 1.3.1 proves that the FPO is inherently conservative, even for the single-species case. A separate question is what long-time steady state the FPO drives the distribution toward. This is answered by Boltzmann's H-theorem [17, 18].

Define the entropy functional for species s as,

$$\mathcal{H}_s(\mathbf{x}, t) = \int f_s(\mathbf{x}, \mathbf{v}, t) \ln f_s(\mathbf{x}, \mathbf{v}, t)\, d^3v, \tag{1.36}$$

which can be interpreted as the negative of the velocity-space entropy density of species s. The H-theorem states that entropy production from collisions between species s and s' is always non-negative [17, 18],

$$\frac{\partial \mathcal{H}_{ss'}}{\partial t} + \frac{\partial \mathcal{H}_{s's}}{\partial t} \geq 0, \tag{1.37}$$

where

$$\frac{\partial \mathcal{H}_{ss'}}{\partial t} = \int (1 + \ln f_s) C_{ss'}[f_s] \, d^3v \tag{1.38}$$

is the contribution to the rate of change of $\mathcal{H}_s$ from collisions between s and s'. Integrating by parts and applying the species-swap symmetry, one arrives at

$$\frac{\partial \mathcal{H}_{ss'}}{\partial t} + \frac{\partial \mathcal{H}_{s's}}{\partial t} = -\frac{\gamma_{ss'}}{m_s} \iint \mathbf{a} \cdot \mathbf{w} \cdot \mathbf{a} \, f_s(\mathbf{v}) f_{s'}(\mathbf{v}') \, d^3v \, d^3v', \tag{1.39}$$

where $\gamma_{ss'} = -m_s \Gamma_{ss'}/2 < 0$, $\mathbf{w}$ is the deflection tensor from Eq. (1.16), and

$$\mathbf{a} = \frac{1}{m_s} \nabla_{\mathbf{v}} \ln f_s(\mathbf{v}) - \frac{1}{m_{s'}} \nabla_{\mathbf{v}'} \ln f_{s'}(\mathbf{v}'). \tag{1.40}$$

The integrand satisfies

$$\mathbf{a} \cdot \mathbf{w} \cdot \mathbf{a} = \frac{a^2 u_{ss'}^2 - (\mathbf{a} \cdot \mathbf{u}_{ss'})^2}{u_{ss'}^3} \geq 0, \tag{1.41}$$

by the Cauchy-Schwarz inequality. Since $f_s f_{s'} \geq 0$ and $\gamma_{ss'} < 0$, the right-hand side of Eq. (1.39) is always non-negative, proving Eq. (1.37).

Entropy production is zero if and only if $\mathbf{a}$ and $\mathbf{u}_{ss'}$ are parallel for all $\mathbf{v}$ and $\mathbf{v}'$. Following Rodman [15], this constraint forces the unique steady-state solution to have the form

$$f_{M,s}(\mathbf{v}) = \frac{n_s}{(2\pi v_{\mathrm{th},s}^2)^{3/2}} \exp\left(-\frac{|\mathbf{v} - \mathbf{u}_s|^2}{2 v_{\mathrm{th},s}^2} \right), \tag{1.42}$$

where n_s, $\mathbf{u}_s$, and $v_{\mathrm{th},s} = \sqrt{T_s/m_s}$ are the number density, bulk velocity, and thermal speed recovered from the velocity moments of f_s. This is the *Maxwellian distribution* — the unique equilibrium of the FPO. In the single-species case, f_s relaxes to a Maxwellian parameterized by its own moments. In the multispecies case, two colliding species reach a common Maxwellian only when they share the same bulk velocity and temperature [15].

For the Maxwellian Eq. (1.42), the Rosenbluth potentials can be evaluated analytically [16]. For an isotropic Maxwellian centered at the origin, they take the closed forms

$$H_M(v) = \frac{n_s}{v}\,\mathrm{erf}\left(\frac{v}{\sqrt{2}\,v_{\mathrm{th},s}}\right), \tag{1.43}$$

$$G_M(v) = \sqrt{2}\,n_s v_{\mathrm{th},s}\left[\frac{e^{-v^2/2v_{\mathrm{th},s}^2}}{\sqrt{\pi}} + \mathrm{erf}\left(\frac{v}{\sqrt{2}\,v_{\mathrm{th},s}}\right)\left(\frac{v_{\mathrm{th},s}}{\sqrt{2}\,v} + \frac{v}{\sqrt{2}\,v_{\mathrm{th},s}}\right)\right], \tag{1.44}$$

where $v = |\mathbf{v}|$ and erf is the error function. For a drifting Maxwellian, v is replaced by $|\mathbf{v} - \mathbf{u}_s|$. These closed-form expressions serve as an important validation benchmark for the numerical Poisson solvers developed in this thesis: for a Maxwellian input distribution, the computed H_s and G_s must recover Eqs. (1.43)–(1.44) to within machine precision.

The Maxwellian is therefore the natural reference state of the FPO. The non-Maxwellian distributions studied in this thesis, introduced in Section 1.4, represent departures from this equilibrium that are not easy to resolve, and stress the importance of numerical and analytic Poisson solvers.

1.4 Suite of Non-Maxwellian Distribution Functions

Plots of all distribution functions studied in this thesis, including the Maxwellian equilibrium reference, are shown in Appendix B. The four non-Maxwellian distributions studied in this thesis are introduced below.

1.4.1 Top-Hat Distribution

The top-hat distribution is a uniform distribution over a finite velocity cube [15],

$$f_{\mathrm{TH}}(\mathbf{v}) = \begin{cases} A & |v_x| \le v_0,\ |v_y| \le v_0,\ |v_z| \le v_0 \\ 0 & \text{otherwise,} \end{cases} \tag{1.45}$$

where $A = 0.5$ and $v_0 = v_{\mathrm{th}}$ in the tests presented here. Its sharp discontinuity at the support boundary $|v| = v_0$ is the primary numerical challenge: polynomial basis representations of the discontinuity introduce Gibbs-like oscillations near the boundary. Moreover, the grid needs to be aligned such that the discontinuity falls exactly on a cell edge. When this alignment is satisfied, the top-hat tests the solver's behavior under a maximally non-equilibrium, non-smooth initial condition.

1.4.2 Bump-on-Tail Distribution

The bump-on-tail distribution consists of a bulk Maxwellian background with an additional beam population displaced in velocity space [15],

$$f_{\mathrm{BOT}}(\mathbf{v}) = f_M(\mathbf{v}) + f_b(\mathbf{v}), \tag{1.46}$$

where

$$f_M(\mathbf{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\mathrm{th}}\right)^3} \exp\left(-\frac{|\mathbf{v}|^2}{2v_{\mathrm{th}}^2}\right), \tag{1.47}$$

$$f_b(\mathbf{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\mathrm{th},b}\right)^3} \frac{A_b^2}{(v_x - u_{x,b})^2 + v_y^2 + v_z^2 + S_b^2} \exp\left(-\frac{(v_x - u_{x,b})^2 + v_y^2 + v_z^2}{2v_{\mathrm{th},b}^2}\right), \tag{1.48}$$

with $n = v_{\text{th}} = 1$, $A_b = \sqrt{0.15}$, $u_{x,b} = 4\,v_{\text{th}}$, $S_b = 0.14$, and $v_{\text{th},b} = 3\,v_{\text{th}}$. The positive velocity gradient $\partial f/\partial v_x > 0$ in the beam region drives the bump-on-tail instability of Langmuir waves, observed in the solar wind and planetary magnetospheres [19]. The parameters for this distribution are chosen following Rodman [15]. For this thesis, the BOT distribution is a demanding test of the Poisson solving schemes: the displaced beam generates a strongly asymmetric H_s and G_s that must be resolved accurately far from the bulk of the distribution.

1.4.3 Kappa Distribution

The kappa distribution is nearly Maxwellian at low energies and transitions to a power-law tail at high energies [19],

$$f_\kappa(\mathbf{v}) = n_0 \left(\pi\theta^2\kappa\right)^{-3/2} \frac{\Gamma(\kappa+1)}{\Gamma(\kappa-1/2)} \left(1 + \frac{|\mathbf{v}|^2}{\kappa\,\theta^2}\right)^{-(\kappa+1)}, \tag{1.49}$$

where $\theta = [(2\kappa-3)k_B T/(\kappa m_s)]^{1/2}$ is the thermal velocity and Γ is the gamma function. The distribution reduces to the Maxwellian as $\kappa \to \infty$ [19]. Kappa distributions with $2 < \kappa < 6$ fit observations in the solar wind, the terrestrial magnetosphere, and the magnetospheres of Mercury, Jupiter, Saturn, and Uranus [19], and direct fitting of Ulysses electron measurements confirms their presence in the solar wind halo at heliocentric distances between 0.3 and 4 AU [20]. We use $\kappa = 2$, giving tails that decay as v^{-6} — far slower than the Gaussian decay of the Maxwellian. The heavy suprathermal tail means H_s and G_s receive non-trivial contributions from large $|\mathbf{v}|$, requiring the Poisson solver to remain accurate well beyond the bulk of the distribution.

1.4.4 Loss-Cone Distribution

The loss-cone distribution arises in magnetically confined plasmas where particles whose velocity satisfies

$$\frac{|v_\parallel|}{|v_\perp|} < \left(\frac{B_{\max}}{B_{\min}} - 1\right)^{1/2} \tag{1.50}$$

are trapped by the mirror force while those outside this region escape along field lines [2]. It is observed in laboratory mirror machines, the Earth's radiation belts, the magnetospheres of Jupiter and Saturn, and solar coronal loops. We define it as

$$f_{\mathrm{LC}}(\mathbf{v}) = \frac{n_0}{\pi^{3/2} v_{\mathrm{th}}^5}\, v_\perp^2\, \exp\left(-\frac{v_\perp^2 + v_z^2}{v_{\mathrm{th}}^2}\right), \tag{1.51}$$

where $v_\perp^2 = v_x^2 + v_y^2$ and $n_0 = v_{\mathrm{th}} = 1$. The $v_\perp^2$ prefactor produces the characteristic depletion at low perpendicular velocity. The anisotropic structure of the loss-cone means H_s and G_s are themselves anisotropic, posing a stricter test of the Poisson solver than any isotropic distribution.

1.5 Summary of Chapter 1

This chapter establishes the physical foundation for the work presented in this thesis. We begin with the landscape of plasma modeling approaches, from MHD and two-fluid models through gyrokinetics to fully kinetic descriptions, motivating the continuum-kinetic approach implemented in GKEYLL as the framework of this thesis. The governing equation of this framework, the Vlasov-Boltzmann equation Eq. (1.4), evolves the distribution function $f_s(\mathbf{x}, \mathbf{v}, t)$ through configuration-space streaming, electromagnetic acceleration, and collisional interactions, with the velocity moments of f_s recovering the macroscopic bulk properties of the plasma. We then explore the reasoning behind choosing a particular collision operator. The BGK operator relaxes f_s toward a Maxwellian at a velocity-independent rate,

making it computationally simple but physically inaccurate for high-energy tails. The LBO improves on this through a drag-diffusion structure but retains a single collision frequency ν_{ss}, still failing to capture the v^{-3} scaling of the Coulomb collision frequency. The FPO resolves both limitations: its drag and diffusion coefficients are computed directly from f_s through the Rosenbluth potentials H_s and G_s, automatically recovering the correct velocity dependence with no free parameters. Section 1.3 presents the Rosenbluth form of the FPO in full. The central result is that H_s and G_s satisfy coupled Poisson equations in velocity space, Eqs. (1.24)–(1.25). These equations represent the primary computational bottleneck in the FPO algorithm and the central target of this thesis. We show that the FPO conserves particle number density, momentum, and energy, and that Boltzmann's H-theorem guarantees non-negative entropy production, with the Maxwellian distribution as the unique steady-state solution. For the Maxwellian, the Rosenbluth potentials admit closed-form expressions Eqs. (1.43)–(1.44), which serve as a validation benchmark for the numerical solvers. The full multispecies form is presented for completeness. However, the implementation developed in this thesis operates on the single-species case, focusing purely on the coupled Poisson solves.

Finally, Section 1.4 introduces the four non-Maxwellian distributions studied in this thesis: the top-hat, bump-on-tail, kappa, and loss-cone distributions. Each represents a distinct departure from thermal equilibrium and poses a different numerical challenge for the Rosenbluth potential solvers. An ideally performing solver is one which can be applied to any arbitrary f_s and solve for the potentials accurately and efficiently, while obeying conservation laws.

Chapter 2 introduces the mathematical framework underlying the numerical implementation: the DG discretization, the motivation behind various schemes, and the existing DG-FPO algorithm.

Chapter 2

Mathematical Tools

The DG-FPO algorithm in GKEYLL rests on two pillars: the DG discretization of the kinetic update, and a method for computing the Rosenbluth potentials and the corresponding drag and diffusion coefficients from them. This chapter establishes the mathematical infrastructure connecting them, before we dive into the intricacies of building a robust method for computing the potentials in Chapters 3, 4, and 5. We start by introducing the DG infrastructure as implemented in GKEYLL, with emphasis on the basis choice and recovery schemes that matter for the FPO. We then move to describe our system of interest — the coupled Poisson system Eqs. (1.24)–(1.25) as a computational problem and motivate the iterative and analytic approaches developed in subsequent chapters. Finally, we present the existing DG-FPO algorithm in full and identify precisely where our Poisson solver contributions extend the algorithm.

2.1 Discontinuous Galerkin Methods in GKEYLL

2.1.1 Preliminaries

Finite element methods (FEM) approximate solutions to a system of equations by representing them as linear combinations of basis functions defined over subdomains of the computational domain. In a continuous formulation, basis functions are shared across element boundaries, enforcing continuity of the solution globally [15]. The discontinuous Galerkin (DG) method relaxes this requirement: the solution is represented as piecewise polynomials of order p within each cell of the computational grid, with no continuity enforced across cell boundaries [15]. Information between cells is exchanged through numerical flux functions at shared boundaries, keeping all computations local to each cell and its immediate neighbors. This locality makes DG highly parallelizable, and the ability to increase p arbitrarily allows high-order accuracy without refining the grid [3]. A full treatment of DG applied to the Vlasov–Maxwell system is given in [3], and the FPO-specific discretization is developed in depth in [15].

GKEYLL uses the *modal* representation, in which the discrete solution in each cell K_j is written as a linear combination of basis functions,

$$f_h^j(\mathbf{z}, t) = \sum_{n=0}^{N_p-1} \widehat{f}_n^j(t)\, \psi_n(\mathbf{z}), \tag{2.1}$$

where $\widehat{f}_n^j(t)$ are time-dependent expansion coefficients and $\psi_n(\mathbf{z})$ are fixed polynomial basis functions defined on a reference element [15]. The Poisson solver developed in this thesis computes the Rosenbluth potentials H_s and G_s directly in this same modal representation, feeding the resulting expansion coefficients into the existing FPO kernels with no additional conversion. All volume, surface, and recovery integrals in the DG update are pre-computed symbolically using Maxima,

a computer algebra system (CAS), prior to runtime, generating kernels that are exact to machine precision, eliminate aliasing errors that would otherwise corrupt the implicit conservation structure of the weak form, and make the DG-FPO update directly GPU-compatible [15]. All FPO algorithms in this thesis use $p = 2$. Figure 2.1 illustrates the modal DG representation for a simple toy Gaussian at $p = 1$ and $p = 2$.

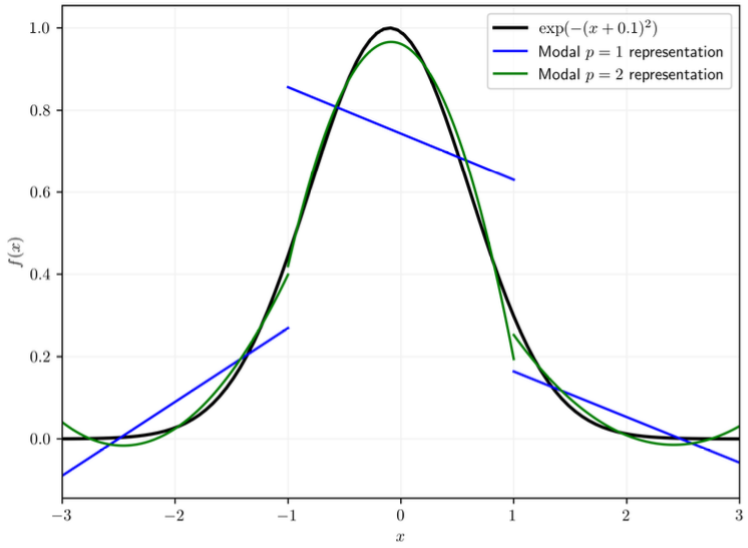

Figure 2.1: Example of a DG discretization using the modal description. The function $\exp(-(x+0.1)^2)$ is discretized using a polynomial order $p = 1$ basis (blue) and $p = 2$ basis (green) over three cells with intervals $[-3,-1]$, $[-1,1]$, and $[1,3]$. Both representations are weakly equal to the original function within their respective cells. Figure reproduced from [15].

Weak Equality

Rather than requiring equations to hold pointwise, DG enforces them in the *weak* sense: the residual of the equation must be orthogonal to every basis function in the cell. Two functions f and g are said to be *weakly equal* over a cell K_j, written $f \doteq g$, if [15]

$$\int_{K_j} (f - g)\, \psi_i \, d\mathbf{z} = 0, \quad \forall\, i \in [0, \ldots, N_p - 1]. \tag{2.2}$$

Projecting a quantity onto the local basis set is a fundamental operation throughout the DG-FPO. The Poisson solver uses weak projection to form the right-hand side of the discrete Poisson system from the distribution function, f_s, and produces H_s and G_s as modal expansion coefficients in the same DG basis used by the FPO — so the two systems interface directly with no additional projection at the handoff.

Choice of basis

Our implementation in GKEYLL uses the *Serendipity* polynomial space, constructed by removing from the full polynomial expansion all monomials whose super-linear degree, the total degree counting only powers strictly greater than one, exceeds p [15]. In variables (x, v_x, v_y, v_z) at $p = 2$, for example, the monomial $x^2 v_x^2 v_y v_z$ has super-linear degree $2 + 2 = 4$ and is excluded [15]. This reduced basis retains the formal convergence order of the full polynomial expansion while growing far more slowly with dimension, making it tractable for simulations in up to six-dimensional phase space [3, 15].

The basis is orthonormalized via the Gram–Schmidt procedure, which eliminates the need to invert any matrix when solving for the expansion coefficients at each timestep and improves the linear independence of the basis compared to a monomial expansion [15].

2.1.2 Recovery

The FPO diffusion term involves second-order velocity-space derivatives of the Rosenbluth potentials. When the DG weak form is applied to such terms and integrated by parts twice, surface integrals appear that require derivatives of the potentials at cell boundaries [15]. Because the DG solution is discontinuous at boundaries, those derivatives are not available directly.

The Recovery-based Discontinuous Galerkin (RDG) scheme resolves this by using

information from neighboring cells to construct a single higher-order polynomial $\widetilde{q}$ of degree $2p+1$ that is continuous across the boundary and weakly equal to the DG approximation q_h in each neighboring cell [15, 21, 22]. For two adjacent cells I_L and I_R with approximate solutions $q_{h,L}$ and $q_{h,R}$, the recovered polynomial satisfies

$$\int_{I_L} \left(\widetilde{q} - q_{h,L}\right) \psi_{Li}\, dx = 0, \tag{2.3}$$

$$\int_{I_R} \left(\widetilde{q} - q_{h,R}\right) \psi_{Ri}\, dx = 0, \tag{2.4}$$

providing $2N_p$ equations for the $2p+2$ coefficients of $\widetilde{q}$ [15]. Because $\widetilde{q}$ has degree $2p+1 > p$, its derivative at the interface can be evaluated without reducing the formal order of accuracy of the scheme. This operation between two cells sharing an interface is called *2-cell recovery*. Figure 2.2 illustrates the recovered polynomial for a simple one-dimensional example.

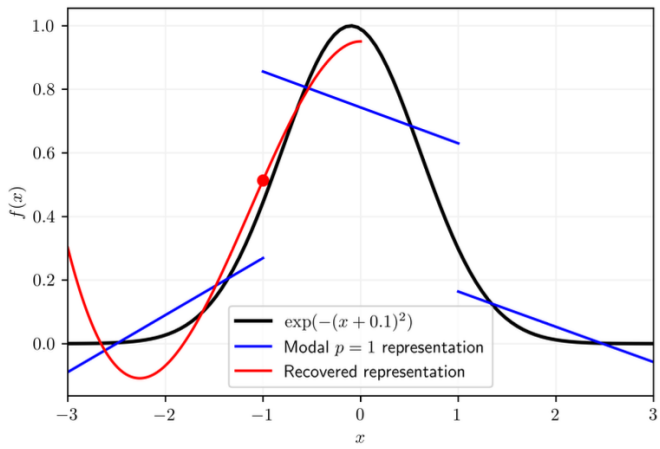

Figure 2.2: Example of 2-cell recovery, where a third-order polynomial (red) is constructed using information from cells defined over $[-3,-1]$ and $[-1,1]$. The recovered polynomial is a better approximation of the original function than the $p=1$ linear representations (blue), and its value at the cell interface (red dot) can be used to evaluate derivatives accurately. Figure reproduced from [15].

The FPO requires two further extensions of this base scheme [15]:

- *1-cell recovery* constructs a polynomial that is continuous at both boundaries

of a single cell, enabling higher-order derivatives to be taken within the cell volume. It proceeds in two steps: 2-cell recovery is performed at each boundary of the central cell, the recovered polynomials are evaluated at the respective interfaces, and these interface values serve as boundary conditions for a final recovery within the cell itself [15].

- *6-cell recovery* constructs a polynomial continuous both across a cell interface and along it, enabling transverse derivatives to be evaluated at the interface. It uses a six-cell stencil: 2-cell recovery is performed in the interface-normal direction for three adjacent cell pairs, the results are projected onto a surface basis of one fewer dimension, and 1-cell recovery is then applied along the interface [15]. Figure 2.3 shows the stencil in two dimensions.

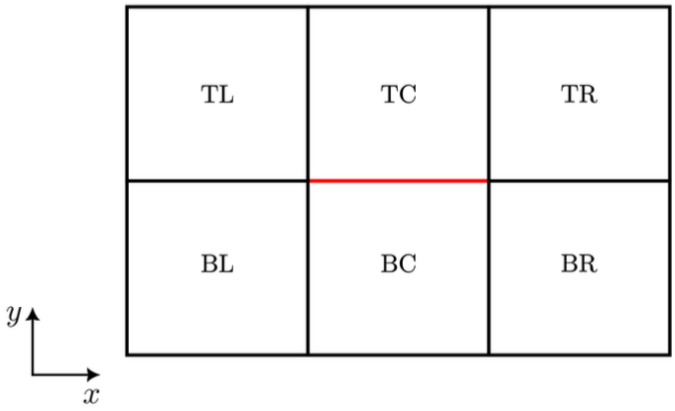

Figure 2.3: Six-cell stencil used for interface-tangential recovery in two dimensions. 2-cell recovery is first performed in the y-direction for each of the three cell pairs (BL-TL, BC-TC, BR-TR). The results are projected onto a surface basis, and 1-cell recovery is applied along the interface (red) in the x-direction. Figure reproduced from [15].

In the DG-FPO, q_h represents the Rosenbluth potentials H_s and G_s. All three recovery variants are used in distinct roles: 1-cell recovery computes the drag coefficient and diagonal diffusion terms in the interior of the velocity domain; 2-cell recovery handles derivatives across cell interfaces for the off-diagonal diffusion

terms; and 6-cell recovery handles transverse derivatives along cell interfaces [15]. At domain boundaries in velocity space, where neighboring cells are unavailable, analytic derivatives of H_s and G_s are supplied in place of recovery [15].

However, when applied to the off-diagonal diffusion term D_{xy}, 6-cell recovery fails to accurately resolve the singularity structure near the origin of velocity space, while 2-cell recovery does not exhibit this failure. Figure 2.4 shows this directly — the spatial structure of D_{xy} computed with 6-cell recovery is qualitatively incorrect, while 2-cell recovery captures it accurately and converges at the expected rate. All implementation runs in this thesis therefore use 2-cell recovery for the off-diagonal diffusion terms. The expected spatial structure of H_s, G_s, and the resulting drag and diffusion coefficients for a Maxwellian distribution is shown in Appendix A for reference.

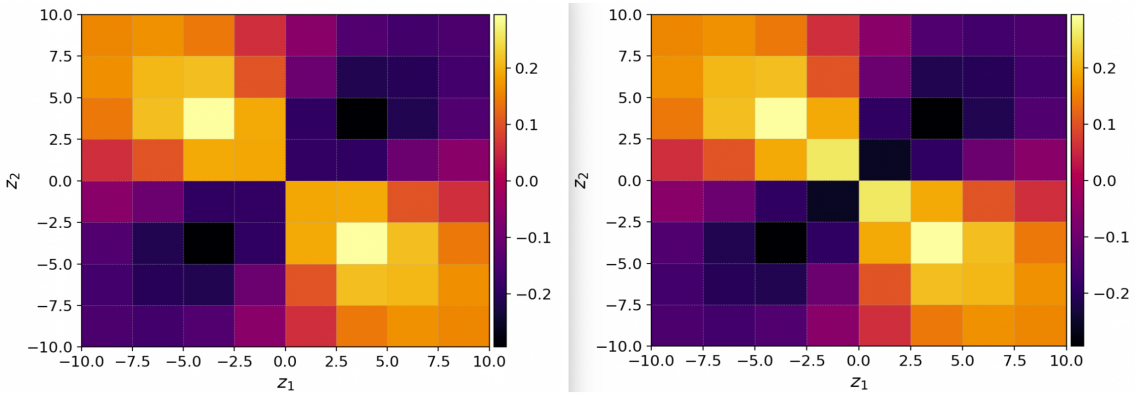

Figure 2.4: Comparison of 6-cell (left) and 2-cell (right) recovery for the off-diagonal diffusion coefficient D_{xy}.

2.1.3 DG Discretization of the Rosenbluth/Fokker–Planck Operator

With the important DG infrastructure foundations in place, we can now describe how it is applied to the FPO. Recall from Chapter 1 the advection-diffusion form of

the FPO for a single species Eq. (1.17),

$$\frac{\partial f_s}{\partial t} = -\frac{1}{2}\nabla_{\mathbf{v}} \cdot \left(\mathbf{a}_h f_s - \mathbf{D}_h \cdot \nabla_{\mathbf{v}} f_s\right), \tag{2.5}$$

where the drag coefficient $\mathbf{a}_h$ and diffusion tensor $\mathbf{D}_h$ are computed from the Rosenbluth potentials via [15, 16]

$$\mathbf{a}_h \doteq 2\Gamma \nabla_{\mathbf{v}} H_h, \tag{2.6}$$

$$\mathbf{D}_h \doteq \Gamma \nabla_{\mathbf{v}} \otimes \nabla_{\mathbf{v}} G_h. \tag{2.7}$$

Here H_h and G_h are the modal DG representations of the Rosenbluth potentials H_s and G_s. The velocity-space derivatives in Eqs. (2.6) and (2.7) are evaluated using the recovery schemes described in Section 2.1.2: 1-cell recovery for the drag coefficient and diagonal diffusion terms in the interior, and 2-cell and 6-cell recovery for the off-diagonal diffusion terms and their transverse derivatives [15].

Following the standard DG discretization process, we multiply Eq. (2.5) by a basis function ψ_i and integrate over each cell K_j. Integrating by parts twice to move derivatives off f_h and onto the test functions yields the semi-discrete FPO [15],

$$\begin{aligned}\int_{K_j} \frac{\partial f_h}{\partial t}\psi_i \, d\mathbf{z} = &-\frac{1}{2}\oint_{\partial K_j} \psi_i^- \, \hat{\mathbf{n}} \cdot \hat{\mathbf{F}} \, dS - \frac{1}{2}\oint_{\partial K_j} \nabla_{\mathbf{v}}\psi^- \cdot \widetilde{\mathbf{D}}_h \widetilde{f}_h \cdot \hat{\mathbf{n}} \, dS \\ &+ \frac{1}{2}\int_{K_j} \nabla_{\mathbf{v}}\psi_i \cdot \mathbf{a}_h f_h \, d\mathbf{z} + \frac{1}{2}\int_{K_j} f_h \nabla_{\mathbf{v}} \cdot \left(\mathbf{D}_h \cdot \nabla_{\mathbf{v}}\psi_i\right) d\mathbf{z},\end{aligned} \tag{2.8}$$

where tildes denote recovered quantities evaluated at cell boundaries, and ψ_i^- indicates the basis function evaluated just inside the cell boundary. The combined numerical flux $\hat{\mathbf{F}}$ contains contributions from both the drag and diffusion terms [15],

$$\hat{\mathbf{F}} = \frac{1}{2}\left(\widetilde{\mathbf{a}}_h f_h^- - \widetilde{\mathbf{a}}_h f^+\right) - \frac{|\max(\widetilde{\mathbf{a}}_h)|}{2}(f_h^+ - f_h^-) - \widetilde{\mathbf{D}}_h \cdot \nabla_{\mathbf{v}} \widetilde{f}_h. \tag{2.9}$$

The drag contribution uses a Lax-Friedrichs flux, which penalizes jumps in f_h across cell interfaces and reduces to a simple upwind flux when the sign of $\widetilde{\mathbf{a}}_h$ is constant along a cell boundary [15]. The diffusion contribution uses the recovered distribution function $\widetilde{f}_h$ to evaluate the gradient across the boundary. The advection velocity $\mathbf{a}_h$ must be continuous across cell boundaries for stability, which is precisely what the recovery scheme enforces [15].

The semi-discrete FPO in Eq. (2.8) does not automatically conserve momentum and energy at the discrete level. This is because the finite velocity domain and the discontinuous cell representation introduce boundary surface integrals and volume moments that do not cancel exactly when summed over all velocity-space cells [15]. Following the same approach used for the LBO in GKEYLL, discrete conservation is enforced by solving for scalar corrections $\delta\mathbf{a}_h(\mathbf{x})$ and $\delta D_h(\mathbf{x})$ — functions of configuration space only — that are added to the drag and diffusion coefficients [15],

$$\mathbf{a}_h \rightarrow \widetilde{\mathbf{a}}_h(\mathbf{x},\mathbf{v}) + \delta\mathbf{a}_h(\mathbf{x}), \tag{2.10}$$

$$\mathbf{D}_h \rightarrow \widetilde{\mathbf{D}}_h(\mathbf{x},\mathbf{v}) + \delta D_h(\mathbf{x})\mathbf{I}. \tag{2.11}$$

The corrections are determined by requiring the discrete analogues of the momentum and energy conservation constraints from Chapter 1 to hold exactly, producing a small linear system for $\delta\mathbf{a}_h$ and δD_h at each configuration space point that is solved as part of the FPO update at every timestep [15]. The corrected coefficients ensure the scheme is fully conservative in number density, momentum, and energy.

2.2 Rosenbluth Potentials and corresponding Poisson Equations

Chapter 1 establishes that the Rosenbluth potentials H_s and G_s satisfy the coupled Poisson system

$$\nabla_{\mathbf{v}}^2 H_s = -4\pi f_s, \tag{2.12}$$

$$\nabla_{\mathbf{v}}^2 G_s = 2H_s, \tag{2.13}$$

and that the drag and diffusion coefficients entering the FPO are computed from these potentials. This section frames Eqs. (2.12)–(2.13) as a computational problem and motivates two trains of thought with respect to solving such systems, which will be used to develop the schemes in Chapters 3, 4, and 5.

2.2.1 Motivating iterative Poisson solving schemes

Discretizing Eqs. (2.12)–(2.13) on a DG or even finite difference (FD) velocity grid converts the Poisson system into a large sparse linear system of the form

$$A\mathbf{u} = \mathbf{f}, \tag{2.14}$$

where $A \in \mathbb{R}^{n\times n}$ is the discrete Laplacian matrix, $\mathbf{u}$ is the vector of expansion coefficients for H_s or G_s, and $\mathbf{f}$ is the corresponding right-hand side. At velocity space resolutions of NV $\in \{16, 32, 64\}$ cells per dimension with $p = 2$ Serendipity basis in GKEYLL, the number of degrees of freedom per solve ranges from $\mathcal{O}(10^4)$ to $\mathcal{O}(10^6)$ at a single configuration-space point. This system must be solved twice per timestep per species — once for H_s and once for G_s. The implementation of any DG-FPO algorithm will require extension to multiple configuration space

cells over multiple dimensions, compounding the computational cost and making solver efficiency and GPU-parallelizability hard requirements for any practical implementation. Figure 2.5 contrasts two representative thermal velocity cases: at $v_{th} = 1.0$, which is the standard test configuration used throughout this thesis, convergence is robust across all three resolutions. At $v_{th} = 0.01$, representing lower thermal velocity, convergence is only achieved at NV $= 32$ and NV $= 64$. Building a Poisson solver capable of handling NV $\in \{16, 32, 64\}$ therefore provides a scheme that is already well-positioned for such demanding resolutions.

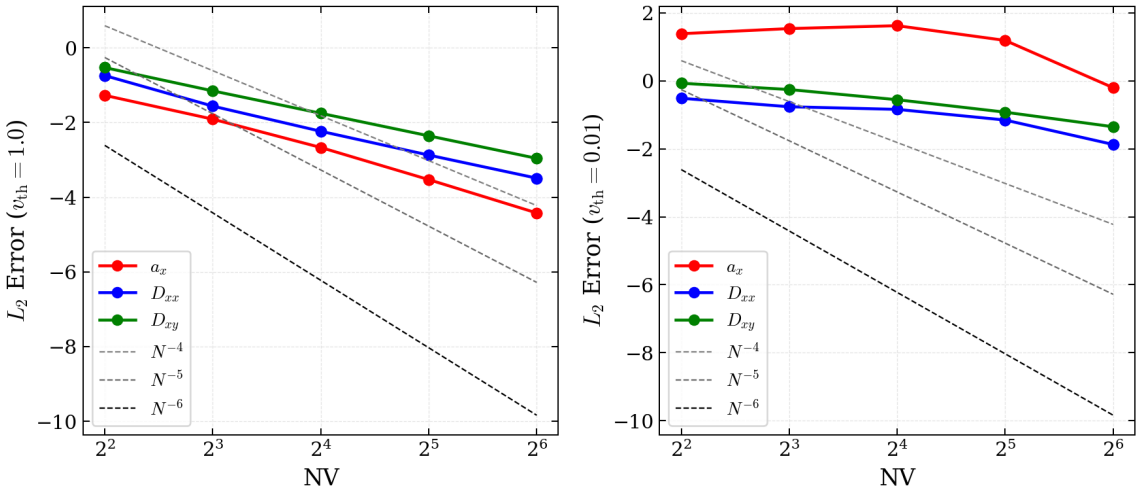

Figure 2.5: L_2 error convergence for a_x, D_{xx}, and D_{xy} as a function of grid resolution NV for $v_{th} = 1.0$ (left) and $v_{th} = 0.01$ (right). At $v_{th} = 1.0$, convergence is robust across NV $\in \{16, 32, 64\}$. At smaller thermal velocities convergence requires NV $= 32$ or higher.

An iterative method starts from an initial guess $\mathbf{u}^{(0)}$ and constructs successive approximations via [23]

$$\mathbf{u}^{(t+1)} := \mathbf{u}^{(t)} + M^{-1}\mathbf{r}^{(t)}, \tag{2.15}$$

where $\mathbf{r}^{(t)} = \mathbf{f} - A\mathbf{u}^{(t)}$ is the residual and M is an invertible matrix approximating

A. The error $\mathbf{e}^{(t)} = \mathbf{u} - \mathbf{u}^{(t)}$ evolves as [23]

$$\mathbf{e}^{(t+1)} = \underbrace{(I - M^{-1}A)}_{E}\, \mathbf{e}^{(t)}, \tag{2.16}$$

where E is the error propagation operator. If A is symmetric positive semi-definite, the error decomposes over the eigenvectors $v^{(i)}$ of A [23],

$$\mathbf{e}^{(t)} = \sum_{i=1}^{n} c_i^{(t)}\, v^{(i)}. \tag{2.17}$$

Classical iterative methods damp error components corresponding to large eigenvalues — the high-frequency, oscillatory components — efficiently. However, components corresponding to small eigenvalues, spanning the *near null space* of A, are reduced extremely slowly and require prohibitively many iterations to eliminate [23]. This is the fundamental limitation that motivates the two approaches developed in this thesis. Algebraic multigrid (AMG), discussed in Chapter 3), resolves the near null space issue by building a hierarchy of coarser representations of A on which smooth error components appear oscillatory and can be corrected efficiently. The Richardson iteration discussed in Chapter 4 requires only applications of the discrete Laplacian ∇_h^2 at each step with no full matrix assembly or inversion, making it matrix-free and directly compatible within GKEYLL's DG infrastructure.

2.2.2 Motivating an Analytic Poisson Solving Scheme

Iterative methods solve Eqs. (2.12)–(2.13) numerically and can be subject to non-ideal eigenvalue behavior and a discretization error floor whose magnitude depends on the resolution of the velocity grid and the smoothness of f_s. An alternative is to exploit the integral structure of the potentials directly. As established in Chapter 1,

H_s and G_s are convolution integrals,

$$H_s(\mathbf{v}) = \int \frac{f_s(\mathbf{v}')}{|\mathbf{v} - \mathbf{v}'|}\, d^3v', \qquad G_s(\mathbf{v}) = \int f_s(\mathbf{v}')|\mathbf{v} - \mathbf{v}'|\, d^3v', \tag{2.18}$$

Rather than solving the Poisson equations numerically for the full distribution f_s, one can decompose f_s into a best-fit Maxwellian f_M, for which analytic expressions for H_M and G_M already exist, and a residual $f_1 = f_s - f_M$ that captures the non-Maxwellian content. The correction potentials are then defined as

$$H_1 = H_s - H_M, \qquad G_1 = G_s - G_M, \tag{2.19}$$

and the problem reduces to finding H_1 and G_1 from f_1. In the far field, the kernels in Eq. (2.18) admit an expansion in terms of velocity-space moments of f_1 — quantities already computed in Step 1 of the DG-FPO algorithm. This is the intuition developed in Chapter 5.

Alternative analytic representations exist — for example, expanding f_s in spherical harmonics reduces the three-dimensional convolution integrals to a series of one-dimensional radial integrals [24]. However, spherical harmonic coefficients are not a natural part of GKEYLL's moment infrastructure, which computes Cartesian velocity moments. Converting between the two representations introduces non-trivial trigonometric projections at every timestep, making the spherical harmonic approach less natural for direct integration with the existing DG-FPO algorithm.

2.3 Existing Discontinuous Galerkin Fokker-Planck Operator Algorithm

With the DG infrastructure and coupled Poisson system established, we can now present the existing DG-FPO algorithm in full and identify precisely where the Poisson solver contributions of this thesis extend it. The existing algorithm computes the Rosenbluth potentials analytically using a best-fit Maxwellian approximation to f_s [15]. We refer to this as the *Maxwellian-approximation* DG-FPO (**MA-DG-FPO**), and use this term throughout the thesis to distinguish it from the extensions developed in subsequent chapters.

Algorithm 1: MA-DG-FPO Collision Update (per timestep) [15]

Input : Distribution function f_h^j on DG basis

Output: Updated f_h^j after one collision timestep

Step 1: Compute velocity-space moments M_0, M_1, M_2 and primitive moments n_s, $\mathbf{u}_s$, $v_{th,s}^2$

Step 2: **For each cell in phase space:**

Evaluate analytic potentials H_s and G_s at Gaussian quadrature points and project to modal basis

For skin cells, evaluate H_s, G_s, $\partial_v H_s$, $\partial_v^2 G_s$ at edges and project to surface basis

Compute $\partial_v G_s$ transversely in skin cells and project to surface basis

Step 3: Iterate over phase-space cells to compute $\mathbf{a}_h$, $\mathbf{D}_h$ and their surface projections

Step 4: Compute boundary corrections and moments from conservation correction equations

Step 5: Form and solve linear system to obtain $\delta\mathbf{a}_h$ and δD_h; apply corrections to $\mathbf{a}_h$ and $\mathbf{D}_h$

Step 6: Compute sign information for corrected drag to determine Lax-Friedrichs flux direction

Step 7: Advance FPO update over all phase-space cells, applying drag and diffusion contributions

Step 8: Compute CFL using the most restrictive timestep; return to Vlasov-Maxwell loop

Each step serves a specific purpose. **Step 1** computes the velocity-space moments needed both to construct the best-fit Maxwellian f_M used in Step 2 and to supply the moments required by the conservation corrections in Steps 4–5. **Step 2** is the potential computation step: in the existing algorithm, the analytic Maxwellian potentials H_M and G_M are evaluated at quadrature points and projected onto the DG modal basis, assuming $f_s \approx f_M$. This assumption fails for non-Maxwellian distributions and is the step replaced by the Extended FPO in Chapter 6. **Step 3** uses the projected potentials to compute the drag coefficient $\mathbf{a}_h$ and diffusion tensor $\mathbf{D}_h$ via the recovery schemes described in Section 2.1.2. At domain boundaries in velocity space where neighboring cells are unavailable, analytic derivatives of H_s and G_s are supplied in place of recovery; the analytic representation used for this boundary correction is developed in Chapter 5.

Steps 4–5 enforce discrete conservation by solving for scalar corrections $\delta\mathbf{a}_h$ and δD_h that ensure the discrete analogues of the momentum and energy conservation constraints from Chapter 1 hold [15]. **Steps 6–8** complete the update: the sign of the corrected drag is precomputed to simplify the Lax-Friedrichs flux evaluation, the FPO update is advanced over all phase-space cells, and the CFL condition is evaluated to set the timestep before returning to the Vlasov-Maxwell loop.

The contribution of this thesis is the replacement of Step 2. Algebraic multigrid (Chapter 3) serves as a numerical benchmark for the Poisson solve, establishing performance and accuracy bounds against which the Richardson iteration (Chapter 4) is validated. The multipole expansion (Chapter 5) complements the iterative solve by providing a representation for the potentials in the far-field and an analytic boundary correction using the moments of $f_1 = f_s - f_M$ that are already available from Step 1.

2.4 Summary of Chapter 2

This chapter establishes the mathematical infrastructure connecting the DG discretization in GKEYLL to the Poisson solver contributions developed in subsequent chapters.

Section 2.1 introduces the DG method as implemented in GKEYLL: the modal Serendipity basis at $p = 2$, weak equality as the fundamental projection operation, and the three recovery variants — 1-cell, 2-cell, and 6-cell — used to evaluate derivatives of the Rosenbluth potentials at cell boundaries. The importance of the recovery scheme choice is established empirically. The DG discretization of the FPO is presented, including the semi-discrete weak form, the numerical flux structure, and the conservation correction procedure.

Section 2.2 frames the coupled Rosenbluth potential Poisson system as a computational problem. At target velocity resolutions for $p = 2$, the number of degrees of freedom per solve and the requirement of two solves per timestep per species place strict demands on solver efficiency, motivating the development of iterative methods — AMG and Richardson iteration — and an analytic representation based on the multipole expansion of the residual $f_1 = f_s - f_M$ using velocity-space moments already available in the DG-FPO algorithm.

Section 2.3 presents the existing DG-FPO algorithm in full and identifies Step 2 — the Rosenbluth potential computation — as the target of the Extended FPO contribution, to be replaced with a Poisson solver capable of handling arbitrary non-Maxwellian distributions.

Chapter 3

Reducing the Problem Size with Multigrid Methods

As established in Chapter 2, the Rosenbluth potentials H_s and G_s satisfy a coupled system of Poisson equations whose solution drives the computation of the drag and diffusion coefficients in the discontinuous Galerkin Fokker–Planck operator (DG-FPO). These coefficients must be evaluated at every timestep of the DG-FPO algorithm, and for each particle species, making the Poisson solves the dominant computational cost of the full collision operator. For most non-Maxwellian distribution functions — arising in multi-species collisional regimes with disparate thermal velocities — no closed-form expressions for H_s or G_s exist. Therefore, a robust numerical Poisson solver is vital for a complete implementation of a generalized FPO.

The challenge is not only mathematical but also computational: at the target velocity space resolutions of NV $\in \{16, 32, 64\}$ with $p = 2$ Serendipity basis, the DOF per solve ranges from $\mathcal{O}(10^4)$ to $\mathcal{O}(10^6)$ per configuration space point, and the solve must be performed twice per timestep per species, imposing strict requirements on solver efficiency and parallel scalability. Furthermore, because the DG-FPO

algorithm is implemented within GKEYLL'S DG infrastructure, any practical solver must integrate naturally with that framework.

Recall from Chapter 2 that the existing DG-FPO implementation in GKEYLL, which evaluates the Rosenbluth potentials analytically using a best-fit Maxwellian approximation, is referred to throughout this thesis as the MA-DG-FPO [15]. This chapter evaluates multigrid methods, specifically algebraic multigrid (AMG), as a benchmark candidate for the Rosenbluth potential solve. Multigrid methods are among the fastest known general-purpose solvers for elliptic and parabolic boundary-value problems, with near-optimal theoretical complexity, and are therefore represented as a numerical benchmark in this work. We implement an AMG solver using the HYPRE preconditioner and solver library [25, 26], benchmark its performance against the MA-DG-FPO analytic projection [15] on the Maxwellian test case, and assess GPU acceleration.

3.1 Multigrid Methods

Multigrid methods achieve fast convergence for large sparse linear systems by employing two complementary processes: *smoothing* and *coarse-grid correction* [25]. As discussed in Section 2.2.1, classical iterative methods such as Gauss–Seidel are effective at damping high-frequency, oscillatory error components — those corresponding to eigenvectors of the system matrix A with large eigenvalues — but are ineffective against smooth, low-frequency components corresponding to eigenvectors with small eigenvalues. These smooth components, referred to as the *near null space* of A, must be eliminated by coarse-grid correction [25]. The key observation is that smooth error on a fine grid appears oscillatory on a coarser grid, where the same local relaxation process can detect and correct it efficiently [27]. By applying relaxation at a hierarchy of grid levels, all error components can

be efficiently reduced, yielding the characteristic $\mathcal{O}(N)$ complexity of multigrid methods in the ideal case [27].

3.1.1 The V-Cycle

The V-cycle is the fundamental building block of multigrid algorithms [23]. Given an approximate solution u to the system $Au = f$, a single V-cycle reduces the error by alternating between smoothing on the fine grid and correction on a coarser grid, as illustrated in Figure 3.1 [23].

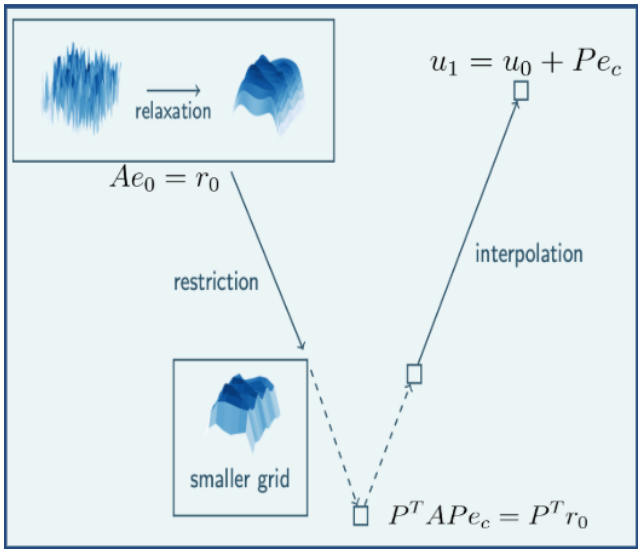

Figure 3.1: Schematic of a single multigrid V-cycle showing relaxation on the fine grid, restriction of the residual to a coarser grid, coarse-grid correction, and interpolation of the correction back to the fine grid.

The residual $r_0 = f - Au_0$ contains both high-frequency and low-frequency error components. A relaxation step (pre-smoothing) damps the high-frequency components, leaving a residual r_0 that is smooth and therefore poorly resolved by further relaxation. This smooth residual is *restricted* to a coarser grid via a carefully constructed restriction operator P^T where it appears oscillatory and can be corrected

efficiently. The coarse-grid error equation

$$P^T A P\, e_c = P^T r_0, \tag{3.1}$$

is solved approximately, where P is the interpolation (*prolongation*) operator and $P^T AP$ is the Galerkin coarse-grid operator [25]. The correction is then interpolated back to the fine grid and the solution is updated:

$$u_1 = u_0 + P e_c. \tag{3.2}$$

Post-smoothing sweeps are applied to u_1 to damp any high-frequency error introduced by the interpolation step. In implementation, parameters such as the number of pre and post-smoothing steps (ν_1, ν_2), the specific methods for relaxation, coarsening, and interpolation, etc., are tunable and affect both the convergence rate and cost per cycle. These parameters are carefully chosen through experimentation to obtain the most efficient scheme for the problem. Applying this two-grid correction approach recursively to the coarse-grid solve in Eq. (3.1) yields the full V-cycle over a hierarchy of grid levels [27].

The optimal complexity of multigrid follows from the geometric structure of the grid hierarchy. At each level, the number of grid points decreases by a factor of 2^d in d dimensions, giving $\log_2 N$ levels and $\mathcal{O}(N)$ total work per V-cycle in the ideal case [27]. In practice, the actual complexity depends on the problem structure, coarsening and interpolation strategy, and multigrid approach, that is if the grid is constructed from the geometry of the problem or the matrix coefficients alone. For general unstructured grids, the construction of the coarse-grid hierarchy requires $\mathcal{O}(N \log N)$ operations, and the total solver complexity is also $\mathcal{O}(N \log N)$ in the nearly optimal case [28]. This distinction becomes particularly relevant for algebraic multigrid (AMG), discussed in the following section, where no geometric

grid hierarchy is available and the coarsening must be inferred entirely from the connection and dependence of matrix coefficients.

3.2 Algebraic Multigrid (AMG) and Problem Setup

The multigrid framework described in Section 3.1 assumes that a hierarchy of coarser grids can be constructed from the geometry of the problem. In practice, this requires explicit knowledge of the mesh structure at each level — a requirement that is natural for structured problems on uniform grids but becomes restrictive for unstructured discretizations and make general application to arbitrary distribution functions non-trivial. AMG [29] caters to such problems by building the entire hierarchy — coarse grids, coarsening and interpolation operators — directly from the matrix coefficients of the large sparse matrix A, with the goal of representing smooth error components faithfully on coarser levels [25]. From the user's perspective and when thinking of integration with the larger DG-FPO algorithm, AMG is a black-box solver: it takes the assembled sparse matrix A and the right-hand side (rhs) b, and returns an approximate solution without any knowledge of the underlying physics or grid geometry. This makes it directly applicable to the Rosenbluth potential system for any arbitrary distribution function f_s without any internal modification to the solver.

3.2.1 Grid and Stencil

The velocity space domain $[-V_{\max}, V_{\max}]^3$ is discretized on a uniform cell-centered grid $p = 0$ with NV cells per dimension and mesh spacing

$$h = \frac{2V_{\max}}{\mathrm{NV}}. \tag{3.3}$$

The Rosenbluth potential equations are posed entirely in velocity space — there is no spatial dimension in the 0X3V configuration considered here — so the computational domain is the three-dimensional velocity grid alone, with no coupling to a physical spatial mesh. Therefore, each cell has a single DOF at its center, giving NV^3 DOF per potential. The discrete Laplacian is approximated by the standard 7-point finite difference stencil in three dimensions:

$$\begin{aligned}(Au)_{i,j,k} = 6u_{i,j,k} \\ - u_{i-1,j,k} - u_{i+1,j,k} \\ - u_{i,j-1,k} - u_{i,j+1,k} \\ - u_{i,j,k-1} - u_{i,j,k+1},\end{aligned} \tag{3.4}$$

which approximates $-h^2\nabla^2 u$ at each interior cell. The matrix A is symmetric positive definite (SPD), with diagonal entries equal to 6 and off-diagonal entries equal to -1. The SPD property guarantees that the system has a unique solution and admits symmetric Krylov solvers and preconditioners. The specific solver configuration used in this work is discussed in Section 3.3.

In our implementation, the matrix A is assembled once and reused for both the H and G solves. This is a critical efficiency property: the AMG setup cost — which includes constructing the coarse-grid hierarchy from the matrix entries — is paid once and shared between both Poisson solves.

3.2.2 Boundary Conditions

Dirichlet boundary conditions are imposed using a ghost-cell approach, which preserves the $\mathcal{O}(h^2)$ convergence of the interior scheme globally. For each boundary face, a ghost cell is placed one grid spacing h outside the domain and assigned the analytic boundary value. The outward-facing stencil entry of the corresponding

boundary cell is zeroed in A, and the ghost value is added to the right-hand side. This avoids modifying the structure of A and handles the edges and corners independently.

For H, the interior right-hand side is $h^2 \cdot 4\pi f_s$ and the ghost value encodes the Dirichlet condition on H at the domain boundary. For G, the same approach applies, but since A represents $-h^2\nabla^2$ and the equation is $\nabla^2 G = 2H$, the interior right-hand side is $-h^2 \cdot 2H$ (negative). The G right-hand side is constructed after the H solve, using the numerically computed H at interior cells and the analytic boundary value for G at ghost centers.

3.2.3 Coupled H–G Solve Algorithm

Algorithm 2: Coupled AMG Solve for Rosenbluth Potentials

Input : Distribution function $f_s(\mathbf{v})$; grid parameters NV, $V_{\max}$

Output: Rosenbluth potentials H_s, G_s on grid

Step 1: Construct 7-point stencil matrix A on $[-V_{\max}, V_{\max}]^3$

Step 2: Apply ghost-cell Dirichlet BCs: zero outward stencil entries on boundary faces; add analytic boundary values to b_H

Step 3: AMG setup: build coarse-grid hierarchy from A (shared by both solves)

Step 4: Solve H system: $A\,\mathbf{H} = h^2 \cdot 4\pi f_s$

Step 5: Build G right-hand side from solved H: set interior entries to $-h^2 \cdot 2H$; add analytic boundary values for G at ghost centers

Step 6: Solve G system: $A\,\mathbf{G} = \mathbf{b}_G$

The specific AMG type, coarsening strategy, interpolation type, relaxation scheme, and strength threshold were selected through systematic experimentation [23]. The

results of that parameter study are presented alongside the full benchmark results in Section 3.3. Some data on the parameter study can be found in Appendix C.

3.3 Benchmarking AMG for the Rosenbluth Potentials

With the discretization for our system and the solver infrastructure established in Section 3.2, we assess the performance of the coupled AMG algorithm across a range of grid sizes, solver parameters, and parallel computing configurations. The benchmark uses a 3D Maxwellian distribution function as the source f_s,

$$f_M(\mathbf{v}) = \frac{n}{(2\pi v_{\text{th}}^2)^{3/2}} \exp\left(-\frac{|\mathbf{v}|^2}{2v_{\text{th}}^2}\right), \tag{3.5}$$

where $|\mathbf{v}|^2 = v_x^2 + v_y^2 + v_z^2$, and n and v_{th} are the number density and thermal speed respectively. For this distribution, the Rosenbluth potentials have the following closed-form analytic expressions [15]:

$$H_M(\mathbf{v}) = \frac{n}{v} \operatorname{erf}\left(\frac{v}{\sqrt{2}\, v_{\text{th}}}\right), \tag{3.6}$$

$$G_M(\mathbf{v}) = \sqrt{2}\, n\, v_{\text{th}} \left[\frac{e^{-v^2/2v_{\text{th}}^2}}{\sqrt{\pi}} + \operatorname{erf}\left(\frac{v}{\sqrt{2}\, v_{\text{th}}}\right)\left(\frac{v_{\text{th}}}{\sqrt{2}\, v} + \frac{v}{\sqrt{2}\, v_{\text{th}}}\right)\right], \tag{3.7}$$

where $v = |\mathbf{v}|$ and $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, \mathrm{d}t$ is the error function. For a drifting Maxwellian with bulk velocity $\mathbf{u}$, the speed v is replaced by the relative speed $|\mathbf{v} - \mathbf{u}|$. These expressions serve as the reference solution for computing the L_2 discretization error, as well as the Dirichlet boundary conditions. For all tests we set $n = 1$, $v_{\text{th}} = 1$, and $V_{\text{max}} = 5.0$, giving a domain that extends to $5\, v_{\text{th}}$ in each velocity direction.

The L_2 error for a quantity Q between the numerical solution and the analytic

reference is measured as

$$E = \sqrt{\int (Q - Q_M)^2 \, d^3v}, \tag{3.8}$$

Solver convergence is monitored through the relative residual norm

$$\rho^{(k)} = \frac{\|b - Au^{(k)}\|_2}{\|b\|_2}, \tag{3.9}$$

where $u^{(k)}$ is the solution at iteration k. Three quantities are recorded for each configuration: the L_2 error in H and G relative to the analytic solution, the AMG setup time shared between both solves, and the individual wall times for the H and G solves.

3.3.1 Convergence Study and Solver Stability

We first study the convergence of H_s and G_s and verify that the coupled AMG solver achieves the $\mathcal{O}(h^2)$ convergence expected from the second-order finite difference stencil introduced in equation (3.4). In this subsection, we use a single MPI rank with the standalone AMG type—BoomerAMG in HYPRE [26]. Table 3.1 reports the L_2 errors and convergence ratios for H and G at grid sizes NV $\in \{32, 64, 128\}$.

NV	$L_2(H)$	Ratio	$L_2(G)$	Ratio
32	9.309e-03	—	6.446e-02	—
64	2.316e-03	2.01	1.588e-02	2.02
128	5.782e-04	2.00	3.943e-03	2.01

Table 3.1: L_2 errors and convergence ratios for the Rosenbluth potentials H and G at increasing grid resolution.

The convergence ratios are consistently close to 2.0 at each refinement for both H and G, confirming $\mathcal{O}(h^2)$ accuracy as expected from the 7-point stencil discretization.

The L^2 error in G is larger than that of H at every resolution, since the G right-hand side is constructed from the numerically computed H, resulting in a compounded error for G.

Figure 3.2 shows the relative residual norm $\rho^{(k)}$ as a function of V-cycle iteration for both H and G at NV $= 128$. Both potentials converge monotonically, confirming solver stability. The G potential converges in 12 cycles compared to 25 for H, consistent with the smoother right-hand side of the G system, which is constructed from the numerically computed H field rather than the raw source f_s. The solver is terminated once a tolerance of 10^{-8} is reached.

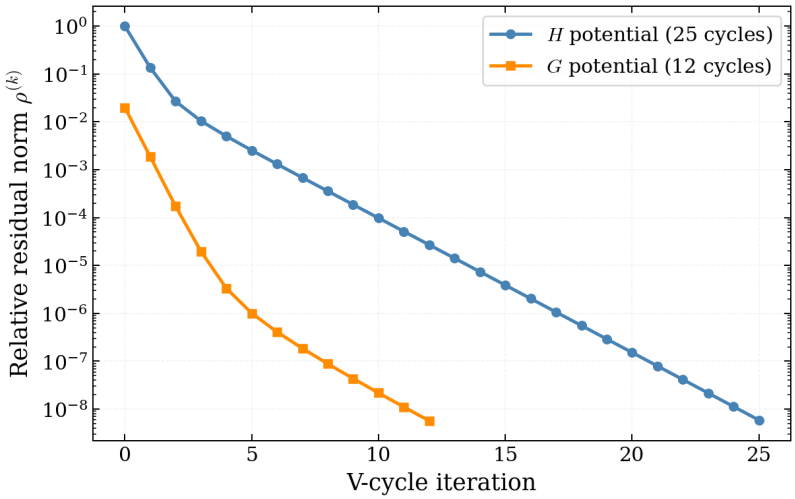

Figure 3.2: Residual norm convergence $\rho^{(k)}$ vs V-cycle iteration for H and G at NV $= 128$, standalone BoomerAMG, single MPI rank.

BoomerAMG is also tested with various preconditioners such as PCG and GMRES. These are options available within the HYPRE library [30]. However, the total compute time in each case depends on the overhead created by processes such as grid and stencil setup which are compounded by the use of algebraic preconditioners. The final configuration uses PMIS coarsening, extended-plus-i interpolation, hybrid symmetric Gauss–Seidel relaxation, and a strength threshold of 0.5, and is fixed for all scaling and GPU acceleration results reported in Section 3.3.2. All of these are iterative schemes chosen via parameter testing applied to obtain the most optimal

V-cycle configuration. The strength threshold plays a key role in deciding how quickly the scheme converges while accurately capturing the connections between the matrix coefficients per V-cycle level.

3.3.2 Scaling and GPU Acceleration

Now that we have established the accuracy and stability of the AMG solver for 0X3V $p = 0$, we move to test its parallelizability. Table 3.2 reports the wall times for the AMG setup phase and the H and G solves across grid sizes NV $\in \{32, 64, 128\}$ for single MPI rank, 8-rank MPI, and GPU configurations. GPU acceleration is implemented using HYPRE's unified memory interface [30], which enables execution on GPU hardware without changes to the solver logic.

NV	Config	Setup (s)	H solve (s)	G solve (s)	Total (s)	H/G iters
32	CPU 1-rank	0.077	0.065	0.043	0.185	15/10
32	CPU 8-rank	0.042	0.033	0.028	0.103	17/11
32	**GPU**	**0.064**	**0.141**	**0.062**	**0.267**	**16/10**
64	CPU 1-rank	0.681	0.664	0.383	1.728	19/11
64	CPU 8-rank	0.249	0.253	0.121	0.623	21/12
64	GPU	0.153	0.303	0.134	0.590	20/11
128	CPU 1-rank	5.930	7.398	3.756	17.084	25/12
128	CPU 8-rank	1.579	5.848	4.513	11.940	27/12
128	**GPU**	**0.678**	**1.050**	**0.434**	**2.162**	**25/12**

Table 3.2: Wall times for the AMG setup and solve processes across increasing velocity resolution and hardware implementations. Bold rows indicate the configurations directly compared in the text.

The AMG setup cost grows substantially with NV for a single MPI rank — and is not significantly reduced by adding more ranks, which reflects the cost of constructing the grids and stencils at every level of the AMG hierarchy without knowledge of

the problem's geometry. While the iteration count increases with increase in ranks or GPU acceleration, the compute time cost per iteration drastically reduces, suggesting a highly-parallelizable algorithm. The G solve remains consistently faster than H at all configurations, requiring roughly half the iterations, reflecting the smoother right-hand side. The 8-rank configuration provides meaningful speedup for the solve phase at all grid sizes, though setup times scale less favourably with additional ranks.

For the GPU configuration, the total solve time at NV $= 32$ is slightly slower than the CPU configurations, suggesting that as the problem size increases, we see the true speedup potential of AMG — at NV $= 128$ the GPU total represents a $7.90\times$ speedup over the single-rank CPU and a $5.52\times$ speedup over the 8-rank CPU. Figure 3.3 summarises the GPU speedup relative to both CPU configurations.

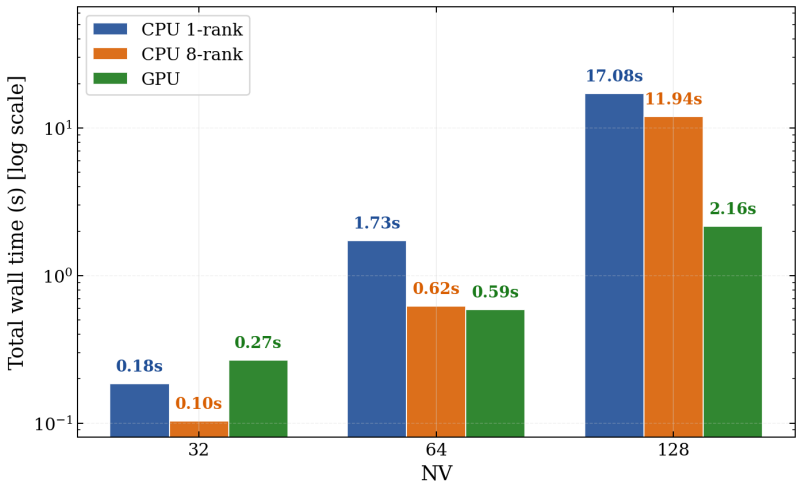

Figure 3.3: Total wall time for the AMG across CPU and GPU configurations, plotted on log scale.

3.3.3 Comparison with MA-DG-FPO Analytic Projection

We have also established that AMG serves as a good benchmark for highly-parallelizable numerical algorithms for our problem. However, we need to assess its performance against the MA-DG-FPO analytic projection scheme in order to have an upper

bound on computational cost of such numerical algorithms [15]. The analytic projection scheme evaluates the closed-form expressions for H_M and G_M directly at the $p = 2$ serendipity basis nodes and projects those values onto the DG basis, achieving machine-precision accuracy without solving any linear system. This is only possible because closed-form expressions for H_M and G_M exist for the Maxwellian distribution. Therefore, this test provides an upper bound on the additional computational cost introduced by a general numerical Poisson solver.

Table 3.3 reports the potential computation times for both schemes. The analytic projection time t_{pot} covers the full H and G projection together, equivalent to the combined H and G solve time in Table 3.2. In terms of a DOF comparison, AMG at NV $= 128$ ($p = 0$, 2.10×10^6 DOF) is compared against the analytic projection scheme at NV $= 32$ ($p = 2$, 8.85×10^5 DOF) and NV $= 64$ ($p = 2$, 7.08×10^6 DOF), which bracket the AMG DOF count from below and above respectively.

Scheme	Config	NV	DOF	t_{pot} (s)
Analytic Proj.	**CPU 1-rank**	**32**	**6.55e+05**	**0.3018**
Analytic Proj.	CPU 1-rank	64	5.24e+06	2.1209
Analytic Proj.	GPU	32	6.55e+05	0.0104
Analytic Proj.	**GPU**	**64**	**5.24e+06**	**0.0505**
AMG	**CPU 1-rank**	**128**	**2.10e+06**	**11.154**
AMG	CPU 8-rank	128	2.10e+06	9.361
AMG	**GPU**	**128**	**2.10e+06**	**1.484**

Table 3.3: Potential computation times for the MA-DG-FPO analytic projection scheme and AMG. Bold rows indicate the configurations directly compared in the text.

The AMG GPU solve at NV $= 128$, $p = 0$ is approximately $5\times$ slower than the analytic projection's CPU 1-rank evaluation at NV $= 32$, $p = 2$, and approximately $30\times$ slower than the analytic projection's GPU evaluation at NV $= 64$, $p = 2$.

Since AMG represents an upper bound on the cost of a general numerical Poisson solver — carrying algebraic overhead from coarse-grid hierarchy construction that geometry-aware algorithms avoid — any numerical scheme built with this in mind must perform better.

3.4 Caveats and Limitations of AMG

The benchmarking results in Section 3.3 demonstrate that the BoomerAMG solver correctly solves the coupled Rosenbluth potential system and scales well in parallel. However, several structural limitations make AMG an unsuitable solver for implementation within the DG-FPO algorithm in GKEYLL.

3.4.1 Scaling at Target Velocity Space Resolutions

Figure 3.4 shows the total wall time of the BoomerAMG solver plotted as $t^{1/3}$ vs NV. The BoomerAMG implementation follows the $\mathcal{O}(N \log N)$ bound closely, confirming that the solver is operating near its theoretical optimum for multigrid on unstructured grids [28].

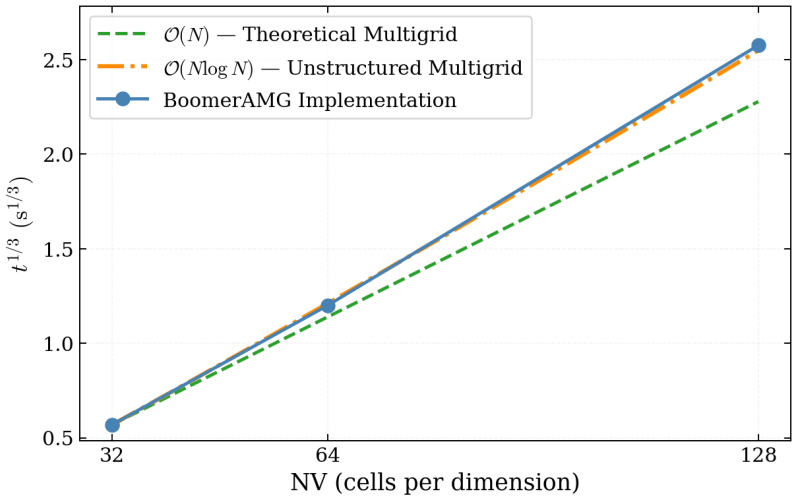

Figure 3.4: Total wall time of the BoomerAMG solver plotted as $t^{1/3}$ vs NV for CPU 1-rank.

While $\mathcal{O}(N \log N)$ is the theoretical optimal, the absolute cost is heavily dominated by the setup cost alone — by almost 35–42% — which is a significant overhead for a one-time cost. This clearly shows that the idea of constructing grids and stencils independent of the problem geometry is not an efficient one. Moreover, the true benefits of multigrid are seen at higher resolutions than required for our batched solves. A simpler solver that is geometric in construction would be more competitive.

3.4.2 Integration with GKEYLL

The AMG solver algorithm developed in this chapter uses a finite difference ($p = 0$) discretization on a semi-structured velocity space, while the DG-FPO algorithm in GKEYLL operates on a $p = 2$ serendipity basis. The AMG implementation will have to be changed to a DG discretization in order to cohesively integrate it as a piece of DG-FPO. This is non-trivial in terms of grid complexity and computational overhead. As a consequence, the tests conducted in this chapter are primarily performance benchmarks rather than accuracy tests.

3.4.3 A DG Geometric Multigrid

The Rosenbluth potential problem is currently being solved on a uniform semi-structured rectangular grid. Even with non-uniform meshes and disparate thermal velocities, the grid structure remains fairly rectangular, making a purely algebraic scheme unnecessarily restrictive. A natural extension of this work is a DG-native geometric multigrid solver for the Rosenbluth potentials. Such a solver would operate directly on GKEYLL'S serendipity basis, use the rectangular velocity grid geometry to construct the coarse-grid hierarchy analytically, and achieve $\mathcal{O}(N)$ complexity with full integration into the DG-FPO algorithm.

3.5 Summary of Chapter 3

This chapter evaluates algebraic multigrid as a benchmark solver for the coupled Rosenbluth potential Poisson system. A finite difference discretization of the 0X3V velocity space domain is implemented using BoomerAMG from HYPRE's preconditioner library, with ghost-cell Dirichlet boundary conditions that preserve $\mathcal{O}(h^2)$ global convergence. The matrix A is assembled once and shared between the H and G solves, with the G right-hand side constructed from the numerically computed H after the first solve.

The convergence study confirms second-order accuracy for both H and G at grid sizes $\mathrm{NV} \in \{32, 64, 128\}$. The solver converges monotonically for both potentials, with G requiring roughly half the iterations of H due to its smoother right-hand side. Iteration counts are hardware-independent across CPU and GPU configurations, confirming the parallel scalability of the AMG scheme.

Comparison with the analytic projection scheme on the Maxwellian case shows that the AMG GPU solve is reasonably more expensive than the analytic projection's CPU and GPU tests at comparable DOF counts. Since AMG represents an upper bound on the cost of a general numerical Poisson solver — carrying algebraic overhead that geometry-aware methods avoid — any solver built on a more efficient iterative scheme is expected to perform better. This establishes that the computational cost of extending the Rosenbluth potential solve to arbitrary distribution functions is not prohibitive.

Three structural limitations prevent seamless integration of AMG with the existing DG-FPO algorithm. First, the setup cost is disproportionately large at the target velocity space resolutions. Second, the finite difference discretization of our AMG algorithm is incompatible with GKEYLL'S DG infrastructure, requiring non-trivial conversion to the latter. Third, AMG's algebraic coarsening discards the structured grid information that a simpler geometry-aware solver would exploit, preventing it

from achieving the theoretically optimal $\mathcal{O}(N)$ complexity.

These findings motivate the development of a simpler iterative solver that operates natively on GKEYLL'S DG basis, carries minimal setup cost, and integrates directly with the DG-FPO algorithm for an arbitrary distribution function f_s. Such a solver is developed and benchmarked in Chapter 4.

Chapter 4

Error-damping using Richardson Iteration

Chapter 3 established algebraic multigrid (AMG) as a robust and parallelizable solver for the coupled Rosenbluth potential system, achieving second-order accuracy and near-optimal $\mathcal{O}(N \log N)$ complexity on the 0X3V $p = 0$ finite-difference (FD) discretization. However, due to its inefficiency at our target velocity space resolutions, its compounding grid and stencil setup cost, and incompatibility with the 1X3V $p = 2$ serendipity basis DG infrastructure of GKEYLL, its direct integration into the existing DG-FPO algorithm is non-trivial. Moreover, AMG's algebraic coarsening discards the structured rectangular velocity grid geometry of the problem entirely, preventing it from achieving the theoretically optimal $\mathcal{O}(N)$ complexity that a geometry-aware solver would exploit.

These limitations point to a specific set of requirements for a practical Rosenbluth potential solver within the DG-FPO framework. The solver must carry minimal computational cost per timestep, while being cohesive with the framework, and applicable to any arbitrary distribution function f_s, including cases arising in multi-species collisional regimes with disparate thermal velocities, where no closed-form

expressions for H_s or G_s exist.

This chapter develops a second-order Richardson iteration as a solver that satisfies these requirements. Our implementation for $p = 2$ uses a discrete RDG Laplacian operator to march a pseudo-time-dependent problem to convergence. At steady-state, the solver obtains a numerical solution for the Poisson equation, which is applied twice in sequence to solve the coupled H–G system — first for H_s, then for G_s using the converged H_s as source. We apply this scheme, both in an FD and RDG setup, along with GPU implementation, and application to the top-hat distribution to show source-independence of the scheme. We also demonstrate a direct performance comparison with AMG and the MA-DG-FPO analytic projection scheme [15], establishing Richardson $p = 2$ on GPU as a competitive alternative to AMG with significantly higher accuracy for the Rosenbluth potential solve in the full DG-FPO algorithm.

4.1 The Damped Wave Equation

The Rosenbluth potential equations (1.24)–(1.25) are elliptic boundary value problems of the form

$$\nabla^2 F = -s, \tag{4.1}$$

where F represents either H_s or G_s and s is the corresponding source term from Eqs. (1.24)–(1.25). Rather than solving Eq. (4.1) directly, the key idea is to introduce a pseudo-time τ and evolve the *damped wave equation* [31, 32, 33]

$$\frac{\partial^2 F}{\partial \tau^2} + \nu \frac{\partial F}{\partial \tau} = \nabla^2 F + s, \tag{4.2}$$

where $\nu > 0$ is a damping coefficient to be chosen. As $\tau \to \infty$, both $\partial_\tau F$ and $\partial_\tau^2 F$ vanish and Eq. (4.2) reduces exactly to Eq. (4.1): the Poisson solution is the

steady state of the damped wave equation. This scheme is a form of second-order Richardson iteration [32, 33].

To understand how quickly errors decay, we write $F = F_0 + F_1$, where F_0 satisfies Eq. (4.1) exactly and F_1 is the error. Subtracting the steady-state equation from Eq. (4.2) gives the homogeneous equation for the error [31]

$$\frac{\partial^2 F_1}{\partial \tau^2} + \nu \frac{\partial F_1}{\partial \tau} = \nabla^2 F_1. \tag{4.3}$$

Assuming a plane-wave mode $F_1 \sim e^{-i\omega\tau} e^{i\mathbf{k}\cdot\mathbf{v}}$ and substituting into Eq. (4.3) yields the dispersion relation [31]

$$\omega(\mathbf{k}) = -i\frac{\nu}{2} \pm \sqrt{k^2 - \frac{\nu^2}{4}}, \qquad k = |\mathbf{k}|. \tag{4.4}$$

Since each mode evolves as $e^{-i\omega\tau}$, the imaginary part of ω governs its decay rate. Two regimes follow:

- **Underdamped** ($k > \nu/2$): the square root is real; both roots carry imaginary part $-\nu/2$, and the mode decays as $e^{-\nu\tau/2}$ at the uniform rate $\nu/2$, independent of k.
- **Overdamped** ($k < \nu/2$): the square root is imaginary; the slower root gives decay rate $\gamma = \nu/2 - \sqrt{\nu^2/4 - k^2} < \nu/2$. The mode at $k_{\min}$ sets the overall convergence rate of the scheme.

The choice $\nu = k_{\min}$ ensures every mode satisfies $k \geq k_{\min} > \nu/2$, placing all modes in the underdamped regime where each decays at the uniform rate $\nu/2$ [31]. On the velocity domain $[-V_{\max}, V_{\max}]^3$, the minimum resolved wavenumber gives

$$\nu = k_{\min} = \frac{\pi}{V_{\max}}. \tag{4.5}$$

4.1.1 Discretization

The pseudo-time derivatives in Eq. (4.2) are replaced by central differences [31]. Central differencing of $\nu\,\partial_\tau F$ gives $\nu(F^{n+1} - F^{n-1})/(2\Delta\tau)$,

$$\frac{F^{n+1} - 2F^n + F^{n-1}}{\Delta\tau^2} + \frac{\nu}{2} \cdot \frac{F^{n+1} - F^{n-1}}{\Delta\tau} = \nabla_h^2 F^n + s, \tag{4.6}$$

where F^n is the field at pseudo-time level n, $\Delta\tau$ is the timestep, and ∇_h^2 is the discrete Laplacian. Solving for F^{n+1} gives the explicit Richardson update [31]

$$F^{n+1} = \frac{1}{C_+}\left(\frac{2}{\Delta\tau^2}F^n - C_- F^{n-1} + \nabla_h^2 F^n + s\right), \tag{4.7}$$

where

$$C_\pm = \frac{1}{\Delta\tau^2} \pm \frac{\nu}{2\Delta\tau}. \tag{4.8}$$

Each iteration of Eq. (4.7) requires exactly one application of ∇_h^2, which is already available in GKEYLL's DG infrastructure. The scheme is therefore matrix-free: no matrix is assembled, there is no setup phase, and there is no algebraic hierarchy. The fastest frequency that must be resolved on a grid with cell spacing $\Delta v = 2V_{\max}/\mathrm{NV}$ is $k_{\max} = \pi/\Delta v$, and the stability condition $k_{\max}\,\Delta\tau < 2$ gives [31]

$$\Delta\tau < \frac{2\,\Delta v}{\pi}. \tag{4.9}$$

For the $p = 0$ FD implementation, ∇_h^2 is the standard 7-point finite difference stencil (3.4) and $\Delta\tau = 0.9 \times 2\Delta v/\pi$, chosen experimentally. For the $p = 2$ RDG implementation, ∇_h^2 is the recovery-based DG Laplacian built from the serendipity basis in GKEYLL [31], whose larger spectral radius relative to the FD stencil requires a reduced pseudo-time step of $\Delta\tau = 0.3 \times 2\Delta v/\pi$.

4.2 Second-order Richardson Iteration for FPO

4.2.1 Coupled H–G Algorithm

The Richardson scheme derived in Section 4.1 is applied twice in sequence to solve the coupled Rosenbluth potential system: first to obtain H_s, then to obtain G_s using the converged H_s as source. Algorithm 3 summarises the full procedure.

Algorithm 3: Coupled Richardson Solve for Rosenbluth Potentials

Input : Distribution function $f_s(\mathbf{v})$; grid parameters NV, $V_{\max}$; pseudo-time step $\Delta\tau$; tolerances ϵ_H, ϵ_G

Output: Rosenbluth potentials H_s, G_s on grid

Step 1: Set $\nu = \pi / V_{\max}$, $C_\pm = 1/\Delta\tau^2 \pm \nu/(2\Delta\tau)$

Step 2: Build source $s_H = 4\pi f_s$; set ghost cells to Dirichlet values from H_M at domain boundary; initialize $F^0 = 0$ on interior

for $n = 0, 1, 2, \ldots$

until $R_2[F^n, s_H] < \epsilon_H$ **do**

> **Step 3**: Compute discrete Laplacian $\nabla_h^2 F^n$
>
> **Step 4**: Update solution: $F^{n+1} \leftarrow \frac{1}{C_+}\left(\frac{2}{\Delta\tau^2}F^n - C_- F^{n-1} + \nabla_h^2 F^n + s_H\right)$
>
> **Step 5**: Compute residual norm: $R_2 \leftarrow \|\nabla_h^2 F^n + s_H\|_2 \,/\, \|s_H\|_2$

end

Step 6: Store $H_{\text{sol}} \leftarrow F^n$; build $s_G = -2H_{\text{sol}}$ on interior; set ghost cells to Dirichlet values from G_M; initialize $F^0 = 0$

Step 7: Apply Steps **3–5** with source s_G and tolerance ϵ_G; store $G_{\text{sol}} \leftarrow F^n$

return H_{sol}, G_{sol}

The H and G solves are strictly sequential: the G source is constructed only after H

has fully converged. As a standalone algorithm, both solves use a *cold start* with $F^0 = 0$ on the interior. This guarantees that the initial residual satisfies $\|\nabla_h^2 F^0 + s\|_2 = \|s\|_2$, so R_2 begins at exactly 1.0 in both cases, giving a consistent convergence history. A *warm start* approach is explored when this scheme is integrated with the rest of the DG-FPO in Chapter 6. Dirichlet boundary conditions are imposed via ghost cells placed outside the domain boundary, assigned the analytic Maxwellian values of H_M and G_M respectively, consistent with the boundary treatment used in the AMG solver of Chapter 3.

4.2.2 Residual Norm and L_2 Error

Convergence of the Richardson iteration is monitored through the residual norm [31]

$$R_2[F, s] = \frac{\|\nabla_h^2 F + s\|_2}{\|s\|_2}, \tag{4.10}$$

where $\|\cdot\|_2$ is the discrete l_2 norm of the DG solution, defined as

$$\|F\|_2 = \sqrt{\sum_j \sum_{k=0}^{N_p-1} \left(\hat{F}_k^j\right)^2}, \tag{4.11}$$

where $\hat{F}_k^j$ are the modal expansion coefficients of F in cell j and N_p is the number of basis functions per cell. The physical accuracy of the converged solution is measured by the continuous L_2 error against the analytic Maxwellian solution,

$$E = \sqrt{\int (F - F_{\text{exact}})^2 \, d^3v}, \tag{4.12}$$

which for the $p = 0$ FD implementation reduces to the cell-averaged form already defined in Eq. (3.8).

4.2.3 Discretization Floor and Iteration Count Formula

The residual norm R_2 and the L_2 error are connected through the discretization floor [31], defined as the residual the exact analytic solution leaves when substituted into the discrete equation,

$$R_{\text{floor}} = R_2[F_{\text{exact}}, s] = \frac{\|\nabla_h^2 F_{\text{exact}} + s\|_2}{\|s\|_2}. \tag{4.13}$$

This is nonzero because the discrete Laplacian (FD and DG) applied to the analytic solution does not exactly recover the source on a finite grid. Once R_2 reaches R_{floor}, the L_2 error stops improving and further iterations gain nothing. The solver tolerance ϵ must therefore be set below R_{floor} to ensure the reported L_2 error reflects the true accuracy of the scheme. The effect of this is particularly seen for the $p = 2$ DG implementation where a tighter tolerance is required for G ($\epsilon_G = 10^{-10}$) compared to H ($\epsilon_H = 10^{-8}$).

A rough estimate for the number of iterations required for the second order Richardson scheme with Dirichlet boundary conditions to reduce R_2 to 10^{-n} can be given by [31],

$$N_{\text{iter}} = \ln(10^n) \left(\frac{\sqrt{N_d}\,(p+1)\,N_{\text{cells}}}{\pi} + 1 \right), \tag{4.14}$$

where p is the polynomial order, $N_d = 3$ is the number of velocity dimensions, and N_{cells} is the number of cells per direction. The linear dependence on N_{cells} follows directly from the stability condition (4.9), since $\Delta\tau \propto \Delta v \propto 1/N_{\text{cells}}$, doubling the resolution doubles the iteration count.

4.3 Convergence Studies of the Richardson Scheme

With the discretization and standalone algorithm described for the second-order Richardson scheme in Sections 4.1 and 4.2, we now assess the performance and

accuracy of the scheme against the theoretical predictions established in Section 4.2.3. All tests are done on a 0X3V domain with parameters $n = 1$, $v_{\text{th}} = 1.0$, and $V_{\text{max}} = 5.0$, identical to the AMG benchmark tests first with the 3D Maxwellian distribution given in Eq. 1.42 (with $\mathbf{u}_s = 0$),

$$f_M(\mathbf{v}) = \frac{n}{(2\pi v_{\text{th}}^2)^{3/2}} \exp\left(-\frac{|\mathbf{v}|^2}{2v_{\text{th}}^2}\right), \tag{4.15}$$

with the corresponding analytic potentials H_M and G_M, given in Eqs. (3.6)–(3.7), that serve as the reference solution for the L_2 error and as the Dirichlet boundary values at the domain boundary. These tests are performed both on CPUs and GPUs. After this, the source-independence is studied using a top-hat distribution. Since we perform a *cold start* approach for both the potential solves using the same Richardson infrastructure, the residual norm behavior is the same for both, that is, the G residual converges at a comparable rate to H. Throughout this section, N_{rich} denotes the observed iteration count at convergence and N_{form} denotes the predicted count from Eq. (4.14).

4.3.1 0X3V $p = 0$ Finite Difference (FD) Scheme (CPU)

Table 4.1 reports the L_2 errors, convergence orders, and iteration counts for NV $\in \{32, 64, 128\}$ with tolerance set at 10^{-8} for both H and G.

NV	$L_2(H)$	Order	$L_2(G)$	Order	$N_{\text{rich}}(H)$	$N_{\text{rich}}(G)$	N_{form}
32	9.31e-03	—	6.45e-02	—	324	324	344
64	2.32e-03	2.00	1.59e-02	2.02	646	646	668
128	5.78e-04	2.00	3.94e-03	2.01	1293	1294	1318

Table 4.1: Richardson solver convergence for 0X3V, FD p = 0, Maxwellian test case.

The scheme achieves second-order convergence in both H and G at each refinement,

consistent with the $\mathcal{O}(h^2)$ accuracy of the 7-point FD stencil. The L_2 error in G is larger than that of H at every resolution, reflecting the compounding of numerical error from the H solve into the G source $s_G = -2H_{\text{sol}}$, consistent with the behavior observed for AMG in Chapter 3. The H and G iteration counts are identical at every resolution: both solves use the same tolerance $\epsilon = 10^{-8}$ and the FD Laplacian introduces no additional discretization error beyond the standard second-order stencil, so the convergence rate depends only on the grid size and ν.

The observed iteration counts lie consistently below N_{form}, with the scheme obeying linear scaling in terms of iteration count. Figure 4.1 shows the H residual norm convergence at NV $= 128$, confirming stable decay of R_2.

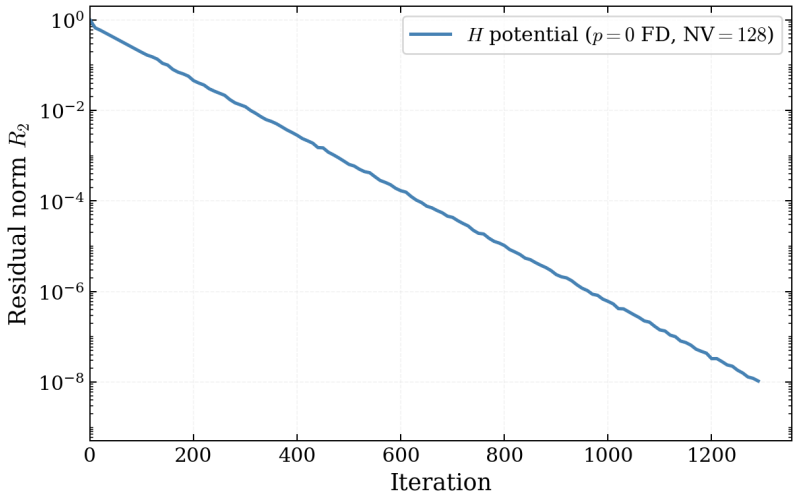

Figure 4.1: Residual norm R_2 vs iterations for the $p = 0$ FD Richardson solver, H potential, NV $= 128$, Maxwellian source.

4.3.2 0X3V $p = 2$ Recovery-based Discontinuous Galerkin (RDG) Scheme (CPU)

Table 4.2 reports the L_2 errors, convergence orders, and iteration counts for NV $\in \{16, 32, 64\}$ on CPU, with tolerances $\epsilon_H = 10^{-8}$ and $\epsilon_G = 10^{-10}$.

NV	$L_2(H)$	Order	$L_2(G)$	Order	$N_{\text{rich}}(H)$	$N_{\text{rich}}(G)$	$N_{\text{form}}(H)$	$N_{\text{form}}(G)$
16	6.86e-05	—	6.87e-05	—	483	604	506	632
32	4.09e-06	4.07	4.17e-06	4.04	961	1209	993	1241
64	**2.82e-07**	**3.86**	**8.45e-07**	**2.30**	**1925**	**2418**	**1968**	**2460**

Table 4.2: Richardson solver convergence for 0X3V, RDG $p = 2$, Maxwellian test case, CPU. N_{form} uses $n = 8$ for H and $n = 10$ for G. Bold row indicates the test where convergence order degrades due to discretization floor.

There are two major results here. First, the scheme achieves approximately fourth-order convergence in H across all tested resolutions, exceeding the third-order accuracy expected from a standard $p = 2$ DG discretization [31]. Second, the convergence order of G degrades at NV $= 64$ due to the discretization floor — once R_2 reaches R_{floor}, further iterations gain nothing and the reported convergence order reflects the floor rather than the true accuracy of the scheme. The scheme already achieves absolute L_2 errors of $\approx 10^{-7}$ for G at NV $= 64$, which is sufficient for the target accuracy of the DG-FPO algorithm. The iteration counts show linear scaling as suggested by the theoretical formula in Eq. (4.14).

Figure 4.2 shows the H residual history at NV $= 32$.

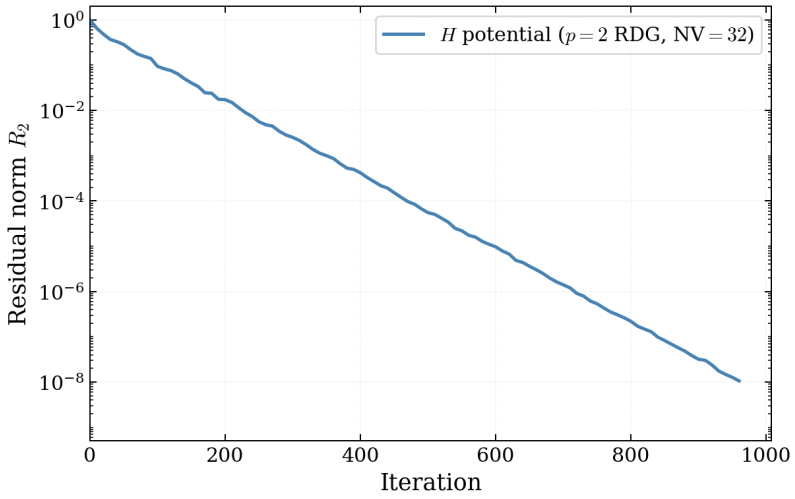

Figure 4.2: Residual norm R_2 vs iterations for the $p = 2$ RDG Richardson solver, H potential, NV $= 32$, Maxwellian source, CPU.

4.3.3 Arbitrary Source: Top-hat Distribution (GPU)

We now simultaneously demonstrate GPU capability and source-independence of the Richardson scheme by testing the $p = 2$ RDG solver on GPU with a top-hat distribution given in Eq. (1.45),

$$f_{\text{th}}(\mathbf{v}) = \begin{cases} A & |v_x| \leq v_0,\ |v_y| \leq v_0,\ |v_z| \leq v_0, \\ 0 & \text{otherwise}, \end{cases} \tag{4.16}$$

where $A = 0.5$ and $v_0 = v_{\text{th}}$. The test is run at NV $= 32$ with the same solver parameters and tolerances as Section 4.3.2. No closed-form analytic solution exists for H or G under this source, so convergence is assessed through the residual norm R_2 alone. Figure 4.3 shows the H residual history.

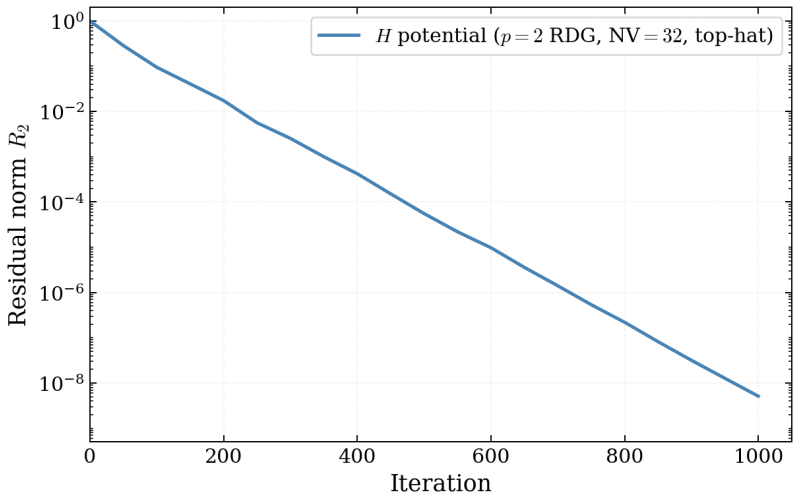

Figure 4.3: Residual norm R_2 vs iterations for the $p = 2$ RDG Richardson solver, H potential, NV $= 32$, top-hat source, GPU.

The GPU implementation for a Maxwellian gives identical L_2 errors and iteration counts with a reduction in compute time of approximately $12\times$ at NV $= 32$ (1.44 s vs 17.25 s). The residual norm convergence for the top-hat matches the Maxwellian case on GPUs clearly showing that the iteration count and wall-clock time are

source-independent. This source-independence will be further studied when the Richardson scheme is incorporated in the existing DG-FPO algorithm in Chapter 6.

4.4 Performance Comparison with AMG and MA-DG-FPO Analytic Projection

The previous section confirmed that the Richardson scheme converges in close agreement with the theoretical predictions of Section 4.2.3. Now we conduct a direct performance comparison with the AMG solver from Chapter 3 and the analytic projection scheme [15] on two metrics: accuracy and computational cost. Table 4.3 summarises the results across all three solvers at comparable DOF counts.

Solver	Config	NV (p)	DOF	$L_2(H)$	$L_2(G)$	Total (s)
AMG	CPU 1-rank	128 ($p = 0$)	2.09e+06	5.78e-04	3.94e-03	17.084
AMG	**GPU**	**128 ($p = 0$)**	**2.09e+06**	**5.78e-04**	**3.96e-03**	**2.162**
Analytic Proj.	CPU 1-rank	32 ($p = 2$)	6.55e+05	4.11e-15	7.89e-14	0.267
Analytic Proj.	**GPU**	**32 ($p = 2$)**	**6.55e+05**	**6.79e-15**	**1.54e-13**	**0.011**
Analytic Proj.	CPU 1-rank	64 ($p = 2$)	5.24e+06	4.08e-15	7.86e-14	2.120
Analytic Proj.	GPU	64 ($p = 2$)	5.24e+06	6.66e-15	1.56e-13	0.051
Richardson	CPU 1-rank	32 ($p = 2$)	6.55e+05	4.09e-06	4.17e-06	17.250
Richardson	**GPU**	**32 ($p = 2$)**	**6.55e+05**	**4.09e-06**	**4.13e-06**	**1.440**
Richardson	CPU 1-rank	64 ($p = 2$)	5.24e+06	2.74e-07	7.33e-07	277.43
Richardson	GPU	64 ($p = 2$)	5.24e+06	2.53e-07	1.21e-06	22.400

Table 4.3: Performance comparison across all three solvers on the Maxwellian test case. AMG total includes setup time. All Richardson and analytic projection totals cover the combined H and G solve. Bold rows indicate the configurations directly compared in the text.

Three major results stand out: First, Richardson $p = 2$ on GPUs at NV $= 32$ (1.44 s) is faster than AMG on GPUs at NV $= 128$ (2.16 s) while using fewer DOF and

achieving an accuracy roughly two orders of magnitude more accurate than AMG on GPUs. This establishes Richardson $p = 2$ on GPUs as the dominant solver for the Rosenbluth potential problem at the target operating resolution of the DG-FPO algorithm.

Second, while the analytic projection scheme is clearly the most accurate and computationally inexpensive it cannot be applied for an arbitrary distribution function. On the other hand, the Richardson scheme obtains reasonably high accuracy but is about $100\times$ slower (1.44 s vs 0.011 s), primarily because the standalone implementation uses a *cold start* with $F^0 = 0$ at every solve. In the extended DG-FPO algorithm developed in Chapter 6, a *warm start* from the previous timestep solution is used instead, which is expected to reduce the iteration count significantly as the potentials change slowly between timesteps.

Third, the CPU implementation of Richardson $p = 2$ is impractical for production use at NV $= 64$ (277.43 s). This is primarily a consequence of the RDG Laplacian having a larger spectral radius than the FD stencil, requiring a stricter $\Delta\tau$ constraint. However, the scheme does obey linear scaling throughout: iteration count grows proportionally with NV, which is fundamentally more favourable than the $\mathcal{O}(N^3)$ cost of a direct LU solver [31]. This leaves a concrete path to optimization — reducing the spectral radius of the discrete Laplacian, for example through a FEM Poisson solver [31].

4.5 Caveats and Limitations of DG Richardson

The Richardson scheme developed in this chapter is a robust and source-independent Poisson solver that could cater to a large class of distribution functions. Moreover, it integrates naturally with GKEYLL's DG infrastructure. Two limitations are worth noting which motivate the work in subsequent chapters and possible future work.

4.5.1 Discretization Floor

As established in Section 4.2.3, the iterative solver cannot reduce R_2 below the discretization floor R_{floor}, which is set by the finite-grid representation of the analytic solution. This is an inherent drawback of any iterative method applied to a discrete system. This limitation motivates analytic approaches to solving the Poisson system as we can obtain closed-form representations of the solution with machine precision accuracy. One such method is discussed in detail in Chapter 5.

4.5.2 An Alternative Discrete Laplacian Operator

The $p = 2$ RDG implementation requires a pseudo-time step three times smaller than the FD implementation due to the larger spectral radius of the RDG Laplacian relative to the FD stencil. This is an inherent property of the RDG operator [31] and cannot be addressed by modifying the recovery scheme. A more promising direction for future work is the use of a finite element method (FEM) discrete Laplacian operator [31], which may reduce the stability constraint on $\Delta\tau$ and improve wall-clock time. Since the Rosenbluth potential solve is primarily a performance bottleneck rather than an accuracy bottleneck at the target resolutions, such a trade-off is well-motivated.

4.6 Summary of Chapter 4

This chapter develops a second-order Richardson iteration as a practical Poisson solver for the coupled Rosenbluth potential system within GKEYLL's DG-FPO framework. The damped wave equation is introduced as the theoretical basis, from which the Richardson update, stability condition, and optimal damping parameter $\nu = \pi / V_{\text{max}}$ are derived. The coupled H–G algorithm is presented as a standalone solver with Dirichlet boundary conditions, and the residual norm R_2, L_2 error, and

discretization floor are defined as the quantities that drive convergence monitoring and accuracy assessment. The theoretical iteration count formula is discussed and shown to provide a reliable upper bound for all tests.

The convergence studies confirm that the scheme performs at or below the theoretical iteration count at all tested resolutions, with second-order accuracy at $p = 0$ and fourth-order convergence at $p = 2$, exceeding the third-order accuracy expected from a standard $p = 2$ DG discretization. GPU acceleration provides approximately a $12\times$ speedup over the CPU implementation at $\mathrm{NV} = 32$, and the top-hat test confirms that the iteration count and wall-clock time are completely source-independent, making the Richardson scheme suitable for any arbitrary distribution function.

In direct performance comparison, Richardson $p = 2$ on GPUs at $\mathrm{NV} = 32$ outperforms AMG on GPUs at $\mathrm{NV} = 128$ in both wall-clock time and accuracy, achieving roughly two orders of magnitude better L_2 error with $3\times$ fewer DOF. While the MA-DG-FPO analytic projection remains faster, it cannot be applied to an arbitrary non-Maxwellian distribution, making Richardson the practical iterative Poisson solver for integration with the DG-FPO algorithm.

Two limitations of the current implementation are worth noting. The discretization floor is an intrinsic property of iterative methods on finite grids and places a lower bound on achievable accuracy, motivating the development of an analytic Poisson solving scheme in Chapter 5. The RDG Laplacian imposes a stricter stability constraint on $\Delta\tau$ relative to the FD stencil due to its larger spectral radius, increasing wall-clock time at higher resolutions. Since this is an inherent property of the operator, a FEM discrete Laplacian is identified as the more promising direction for future work, offering reduced computational overhead without the pseudo-time stability constraint of the Richardson iteration.

Chapter 5

An Analytic Representation using Multipole Expansions

The iterative methods discussed in Chapters 3 and 4 provide numerical solutions to the Rosenbluth potential equations in differential form, but are subject to a discretization floor that places a fundamental lower bound on achievable accuracy. An alternative approach is to represent potentials analytically by exploiting the integral structure of H_s and G_s directly. This chapter develops such a representation using a multipole expansion in terms of the full Cartesian moments of the distribution function and derives the corresponding drag and diffusion coefficients. We also discuss where this approach can be optimally applied in the velocity domain and establish a multipole correction algorithm for the DG-FPO. Finally, we validate this algorithm through relaxation tests across a suite of non-Maxwellian distributions described in Chapter 1, ensuring conservation of moments and that the correction does not disrupt the rest of the existing DG-FPO algorithm.

5.1 Multipole Expansion of the Rosenbluth Potentials and Drag–Diffusion Coefficients

The Rosenbluth potentials H_s and G_s, introduced in Chapter 1, are defined through the velocity-space convolution integrals

$$H_s(\mathrm{v}) = \int \frac{f_s(\mathrm{v}')}{|\mathrm{v} - \mathrm{v}'|}\, d^3v', \tag{5.1}$$

$$G_s(\mathrm{v}) = \int f_s(\mathrm{v}')|\mathrm{v} - \mathrm{v}'|\, d^3v', \tag{5.2}$$

which are the Green's function solutions to the Poisson equations $\nabla^2 H_s = -4\pi f_s$ and $\nabla^2 G_s = 2H_s$ (Eqs. (1.24)–(1.25)). The integral form of H_s shares the mathematical structure of the Newtonian gravitational potential, with f_s playing the role of a mass density distributed in velocity space [34]. This analogy directly motivates a multipole expansion: just as the gravitational potential of a localized mass distribution can be expanded in inverse powers of the field point distance, H_s admits an expansion in inverse powers of $v = |\mathrm{v}|$ valid in the region $|\mathrm{v}| \gg |\mathrm{v}'|$ where $f_s(\mathrm{v}')$ is concentrated. This far-field condition is the fundamental restriction of the expansion and motivates the guard radius framework developed in Section 5.2. The use of multipole expansions as a correction to the Rosenbluth potentials in finite velocity domains is motivated in part by prior work in the FPO literature [35].

We expand the kernel $1/|\mathrm{v} - \mathrm{v}'|$ as a multivariate Taylor series about $\mathrm{v}' = \mathbf{0}$,

$$\frac{1}{|\mathrm{v} - \mathrm{v}'|} = \sum_{\ell=0}^{\infty} \frac{(-1)^\ell}{\ell!} v'^{\langle a_1} \cdots v'^{a_\ell\rangle} \partial_{a_1} \cdots \partial_{a_\ell} \frac{1}{v}, \tag{5.3}$$

where angle brackets denote the symmetric trace-free (STF) part of the enclosed tensor indices [34]. The derivative tensor $\partial_{a_1} \cdots \partial_{a_\ell}(1/v)$ is itself symmetric and trace-free away from the origin [36], a direct consequence of $\nabla^2(1/v) = -4\pi\delta^{(3)}(\mathrm{v})$,

so only the STF part of $v'^{a_1}\cdots v'^{a_\ell}$ survives the contraction at each order ℓ. Trace components of the source moments do not contribute to the far-field potential — this is the central advantage of the STF formulation over a generic Cartesian tensor expansion.

Substituting Eq. (5.3) into Eq. (5.1) and interchanging sum and integral — valid in the far-field where the series converges [34] — gives the general multipole form

$$H_s(\mathrm{v}) = \sum_{\ell=0}^{\infty} \frac{(-1)^\ell}{\ell!} \mathcal{T}_s^{\langle a_1\cdots a_\ell\rangle}\, \partial_{a_1}\cdots\partial_{a_\ell}\frac{1}{v}, \tag{5.4}$$

where the STF multipole moments of f_s are defined as

$$\mathcal{T}_s^{\langle a_1\cdots a_\ell\rangle} = \int f_s(\mathrm{v}')\, v'^{\langle a_1\cdots a_\ell\rangle}\, d^3v'. \tag{5.5}$$

Applying the derivative identity [34]

$$\partial_{a_1}\cdots\partial_{a_\ell}\frac{1}{v} = (-1)^\ell(2\ell-1)!!\,\frac{\hat{v}^{\langle a_1\cdots a_\ell\rangle}}{v^{\ell+1}}, \tag{5.6}$$

where $\hat{v}^{\langle a_1\cdots a_\ell\rangle}$ is the STF tensor of the unit vector $\hat{\mathrm{v}} = \mathrm{v}/v$ and $(2\ell-1)!! = 1\cdot 3\cdot 5\cdots(2\ell-1)$, the expansion takes the compact form

$$H_s(\mathrm{v}) = \sum_{\ell=0}^{\infty} \frac{(2\ell-1)!!}{\ell!}\, \mathcal{T}_s^{\langle a_1\cdots a_\ell\rangle}\, \frac{\hat{v}^{\langle a_1\cdots a_\ell\rangle}}{v^{\ell+1}}. \tag{5.7}$$

Writing out the first four terms explicitly:

$$H_s(\mathrm{v}) = \underbrace{\frac{n_s}{v}}_{\ell=0} + \underbrace{\frac{\mathcal{T}_s^{\langle a\rangle}v^a}{v^3}}_{\ell=1} + \underbrace{\frac{3\mathcal{T}_s^{\langle ab\rangle}v^a v^b}{2v^5}}_{\ell=2} + \underbrace{\frac{5\mathcal{T}_s^{\langle abc\rangle}v^a v^b v^c}{2v^7}}_{\ell=3} + \mathcal{O}(v^{-5}), \tag{5.8}$$

where the individual moments at each order are

$$\ell = 0: \quad \mathcal{T}_s^{(0)} = \int f_s(\mathrm{v}')\, d^3v' = n_s, \tag{5.9}$$

$$\ell = 1: \quad \mathcal{T}_s^{\langle a\rangle} = \int f_s(\mathrm{v}')\, v'^a\, d^3v' = n_s u_s^a, \tag{5.10}$$

$$\ell = 2: \quad \mathcal{T}_s^{\langle ab\rangle} = \int f_s(\mathrm{v}') \left(v'^a v'^b - \tfrac{1}{3} v'^2 \delta^{ab}\right) d^3v', \tag{5.11}$$

$$\ell = 3: \quad \mathcal{T}_s^{\langle abc\rangle} = \int f_s(\mathrm{v}')\, v'^{\langle abc\rangle}\, d^3v', \tag{5.12}$$

with the STF combination at octupole order

$$v'^{\langle abc\rangle} = v'^a v'^b v'^c - \frac{v'^2}{5}\left(v'^a \delta^{bc} + v'^b \delta^{ac} + v'^c \delta^{ab}\right). \tag{5.13}$$

The monopole ($\ell = 0$) gives the number density n_s; the dipole ($\ell = 1$) gives $n_s u_s^a$, the first velocity moment. For symmetric distributions the dipole of the residual $f_1 = f_s - f_M$ vanishes, but for asymmetric distributions such as the bump-on-tail it is the dominant far-field correction, decaying only as v^{-2} compared to v^{-4} for the quadrupole — its omission would introduce the largest error in the computed potentials. The quadrupole ($\ell = 2$) captures pressure anisotropy through the traceless part of the second velocity moment, and the octupole ($\ell = 3$) captures anisotropic heat flux through the traceless part of the third moment. All four terms are computed unconditionally in the implementation.

Although the STF expansion and a spherical harmonic expansion are equivalent representations of the same potential, STF tensors carry the advantage of coordinate independence — they do not require a preferred axis in velocity space and permit calculations entirely in Cartesian coordinates [24]. More importantly for the implementation, GKEYLL computes the full Cartesian velocity moments $M_s^{(0)}$, $M_{s,i}^{(1)}$, $M_{s,ij}^{(2)}$, $M_{s,ijk}^{(3)}$ as part of its standard FPO update [15, 37], so no additional moment calculation is required. When these Cartesian moments are substituted into Eq. (5.8),

the trace components cancel identically within each radial term. For example, the quadrupole contribution to H_s evaluates to

$$H_s^{(\ell=2)}(\mathrm{v}) = \frac{3M_{s,ab}^{(2)} v^a v^b}{2v^5} - \frac{M_{s,aa}^{(2)}}{2v^3}, \tag{5.14}$$

where the second term subtracts the trace of $M_{s,ij}^{(2)}$ exactly, leaving only the anisotropic contribution. An explicit STF projection is therefore not required, making the expansion directly compatible with the existing moment infrastructure in the DG-FPO algorithm without additional overhead.

The same Taylor expansion applied to the kernel $|\mathrm{v} - \mathrm{v}'|$ in Eq. (5.2) yields a parallel multipole form for G_s, using the same STF moments $\mathcal{T}_s^{\langle a_1 \cdots a_\ell \rangle}$:

$$G_s(\mathrm{v}) = \sum_{\ell=0}^{\infty} \frac{(2\ell-1)!!}{\ell!}\, \mathcal{T}_s^{\langle a_1 \cdots a_\ell \rangle}\, \frac{\hat{v}^{\langle a_1 \cdots a_\ell \rangle}}{v^{\ell-1}}. \tag{5.15}$$

The key structural difference from H_s is the $v^{-(\ell-1)}$ dependence rather than $v^{-(\ell+1)}$: G_s decays more slowly in the far field, with the monopole term growing as v rather than decaying. This slower decay means G_s carries more far-field content than H_s, making the multipole correction particularly important for the diffusion tensor $\mathbf{D}_s$. The explicit term-by-term expressions for G_s are given in Appendix D.

The drag coefficient $\mathbf{a}_s$ and diffusion tensor $\mathbf{D}_s$ follow by differentiation of H_s and G_s respectively, as established in Chapter 1:

$$a_{s,i} = 2\Gamma \frac{\partial H_s}{\partial v_i}, \qquad D_{s,ij} = \Gamma \frac{\partial^2 G_s}{\partial v_i \partial v_j}. \tag{5.16}$$

All three components of $\mathbf{a}_s$ and all six independent components of the symmetric diffusion tensor $\mathbf{D}_s$ are computed by applying the corresponding partial derivatives of the multipole expansions of H_s and G_s up to order $\ell = 3$, using the pre-computed Cartesian moments. The explicit expressions for all nine quantities are given in

Appendix E.

5.2 Far-Field Application of the Multipole Correction

The Maxwellian approximation f_M, constructed to match the zeroth and first two moments of f_s, provides an analytic baseline for the Rosenbluth potentials through the closed-form expressions H_M, G_M established in Chapter 1. For distributions with significant non-Maxwellian structure, the residual $f_1 = f_s - f_M$ carries non-trivial higher-order moments that propagate into errors in the drag and diffusion coefficients:

$$H_1 = H_s - H_M, \quad G_1 = G_s - G_M, \quad \mathbf{a}_1 = \mathbf{a}_s - \mathbf{a}_M, \quad \mathbf{D}_1 = \mathbf{D}_s - \mathbf{D}_M. \tag{5.17}$$

The multipole expansions in Section 5.1 provide an analytic representation of these residual quantities in terms of the moments of f_1. However, since the expansion is an asymptotic series in $|\mathrm{v}'|/|\mathrm{v}|$, it converges only where $|\mathrm{v}|$ lies outside the bulk support of f_1. Inside this region the $1/|\mathrm{v}|$ terms diverge and the series is invalid. The multipole correction is therefore applied only in the far field, beyond a guard radius v_{guard}, where the series is mathematically convergent. Whether higher-order moments ($\ell \geq 4$) could improve the accuracy of the multipole correction is still an open question; however, these terms share the same near-field divergence and do not extend the region of convergence. Inside the guard radius, the Maxwellian baseline H_M, G_M, $\mathbf{a}_M$, $\mathbf{D}_M$ is used without any correction. To ensure a smooth transition between the two regimes, a smooth weight function replaces a hard cutoff, as described below.

The weight function is a smooth tanh ramp,

$$w(|\mathrm{v}|) = \frac{1}{2}\left(1 + \tanh\left(\frac{|\mathrm{v}| - v_{\text{guard}}}{\delta}\right)\right), \tag{5.18}$$

where v_{guard} is the guard radius and δ controls the width of the transition. At $|\mathrm{v}| = v_{\text{guard}}$ the weight is exactly $\frac{1}{2}$; for $|\mathrm{v}| \gg v_{\text{guard}}$ it approaches unity and the full multipole correction is applied; for $|\mathrm{v}| \ll v_{\text{guard}}$ it approaches zero and the Maxwellian baseline is retained without modification. The hyperbolic tangent is chosen for its smoothness, monotonicity, and interpretable parameters, and introduces no discontinuities into the DG representation of the corrected potentials. The choice of v_{guard} and δ for each distribution is discussed in Section 5.2.3.

5.2.1 The Multipole Correction Algorithm

Given a distribution function f_s projected onto the DG basis, the Multipole-Corrected DG-FPO (MC-DG-FPO) computes the corrected potentials in seven steps. **Steps 1–4** prepare the Maxwellian baseline and the residual moments. **Step 5** evaluates the multipole correction from the residual moments. **Step 6** applies a dual condition mask to determine where the correction is valid. **Step 7** assembles the final corrected quantities. The Cartesian velocity moments required in **Step 1** are already computed as part of the standard FPO update [15, 37], so no additional moment calculation is needed beyond what is already available in the existing algorithm.

Step 6 implements a *dual condition mask* combining two independent checks before applying the correction to any cell. Condition A is the far-field guard: the tanh weight $w(|\mathrm{v}|)$ smoothly suppresses the correction in the near-field where the multipole series diverges. Condition B is a significance check: cells where $|f_1(\mathrm{v})| < \varepsilon |f_s(\mathrm{v})|$ for a tolerance ε are skipped, preventing the correction from amplifying numerical noise in regions where $f_s \approx f_M$. Both conditions must be

satisfied for the correction to be applied.

Algorithm 4: Multipole-Corrected FPO (MC-DG-FPO) Potential Computation

Input : Distribution function f_s on DG basis; grid (NV, p, basis)

Output: H_s, G_s, $a_{s,x}$, $a_{s,y}$, $a_{s,z}$, $D_{s,xx}$, $D_{s,yy}$, $D_{s,zz}$, $D_{s,xy}$, $D_{s,xz}$, $D_{s,yz}$

Step 1: Compute Cartesian moments $M_s^{(0)}$, $M_{s,i}^{(1)}$, $M_{s,ij}^{(2)}$, $M_{s,ijk}^{(3)}$ of f_s

Step 2: Fit best-fit isotropic Maxwellian f_M matching $M_s^{(0)}$ and $\mathrm{Tr}(M_{s,ij}^{(2)})$

Step 3: Evaluate analytic Maxwellian potentials H_M, G_M; derive $\mathbf{a}_M$, $\mathbf{D}_M$ by differentiation

Step 4: Compute $f_1 = f_s - f_M$; evaluate moments $M_1^{(0)}$, $M_{1,i}^{(1)}$, $M_{1,ij}^{(2)}$, $M_{1,ijk}^{(3)}$ of f_1

Step 5: Evaluate multipole terms for H_1, G_1 from moments of f_1; derive $\mathbf{a}_1$, $\mathbf{D}_1$ by differentiation

Step 6: *Dual condition mask*

for *each cell at* v **do**

> *Condition A:* compute $w(|\mathrm{v}|)$ via Eq. (5.18)
>
> **if** $w < 10^{-6}$ **then** skip cell
>
> *Condition B:* **if** $|f_1| < \varepsilon |f_s|$ **then** skip cell
>
> Both conditions passed: apply weighted correction

end

Step 7: **for** *each*

$Q \in \{H_s, G_s, a_{s,x}, a_{s,y}, a_{s,z}, D_{s,xx}, D_{s,yy}, D_{s,zz}, D_{s,xy}, D_{s,xz}, D_{s,yz}\}$ **do**

> $Q_{\text{final}} \leftarrow Q_M + w(|\mathrm{v}|) \cdot Q_1$

end

return Q_{final} for all output quantities

5.2.2 Validation on the Loss-Cone Distribution

The loss-cone distribution, introduced in Section 1.4.4, is the natural first validation target for the multipole correction: its characteristic pressure anisotropy — $M^{(2)}_{1,xx} \neq M^{(2)}_{1,zz}$ — produces non-trivial quadrupole and octupole contributions to $f_1 = f_{LC} - f_M$ that feed directly into errors in the off-diagonal diffusion coefficients D_{xy} and D_{xz}, where the Maxwellian approximation is expected to be least accurate. We define the loss-cone distribution as

$$f_{LC}(\mathrm{v}) = \frac{n_0}{\pi^{3/2} v_{\mathrm{th}}^5} v_\perp^2 \, \exp\left(-\frac{v_\perp^2 + v_z^2}{v_{\mathrm{th}}^2} \right), \tag{5.19}$$

where $v_\perp^2 = v_x^2 + v_y^2$, $n_0 = 1.0$, and $v_{\mathrm{th}} = 1.0$. The simulation is in 1X3V with $p = 2$ Serendipity basis, $\mathrm{NV} = 64$ cells per velocity dimension, and domain $[-10, 10]\, v_{\mathrm{th}}$ in each direction. The guard radius is $v_{\mathrm{guard}} = 2.5\, v_{\mathrm{th}}$ with $\delta = 0.5\, v_{\mathrm{th}}$; the heuristic for this choice is established in Section 5.2.3.

Figure 5.1 shows D_{xy} for the MA-DG-FPO and MC-DG-FPO in the v_x–v_z plane at $v_y = 0$, together with the difference $\Delta D_{xy} = D^{\mathrm{MC}}_{xy} - D^{\mathrm{MA}}_{xy}$. The two panels are visually similar, reflecting that the Maxwellian approximation captures the leading-order structure of D_{xy}. The difference ΔD_{xy} reveals the correction concentrated in the 2.5–6.5 v_{th} shell where the series is convergent, with an anisotropic pattern directly reflecting the pressure anisotropy of f_1.

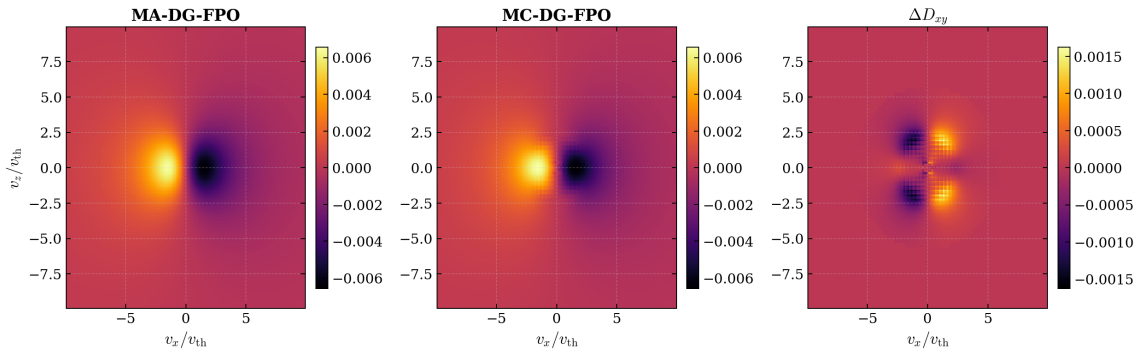

Figure 5.1: D_{xy} for the loss-cone distribution in the v_x–v_z plane ($v_y = 0$). Left: MA-DG-FPO. Centre: MC-DG-FPO. Right: difference ΔD_{xy}. NV $= 64$, $p = 2$, $v_{\mathrm{guard}} = 2.5\, v_{\mathrm{th}}$, $\delta = 0.5\, v_{\mathrm{th}}$.

5.2.3 Guard Radius Selection

The Taylor expansion underlying the multipole representation of H_1 converges only in the far field [34, 36]. Applying the multipole formula inside the bulk support of f_1 produces non-physical divergences in the drag and diffusion coefficients. The guard radius v_{guard} must therefore be chosen large enough to exclude the bulk of f_1 while small enough to leave a meaningful far-field region where the correction is active.

To determine v_{guard} empirically, we sweep it over $[0.5, 8.0]\, v_{\mathrm{th}}$ at fixed $\delta = 0.5\, v_{\mathrm{th}}$, NV $= 32$, $p = 2$ for the loss-cone distribution, evaluating the relative L_2 norm of the correction in a_x, D_{xx}, and D_{xy},

$$\mathcal{E}(Q) = \frac{\|Q^{\mathrm{MC}} - Q^{\mathrm{MA}}\|_{L_2}}{\|Q^{\mathrm{MA}}\|_{L_2}}, \tag{5.20}$$

where Q^{MC} and Q^{MA} are the MC-DG-FPO and MA-DG-FPO solutions respectively. Each value of v_{guard} corresponds to a separate test in which the MC-DG-FPO and MA-DG-FPO evaluate the drag and diffusion coefficients on the same fixed loss-cone distribution without time evolution, with $\mathcal{E}(Q)$ quantifying the shift in Q relative to the MA-DG-FPO baseline. The results are shown in Figure 5.2.

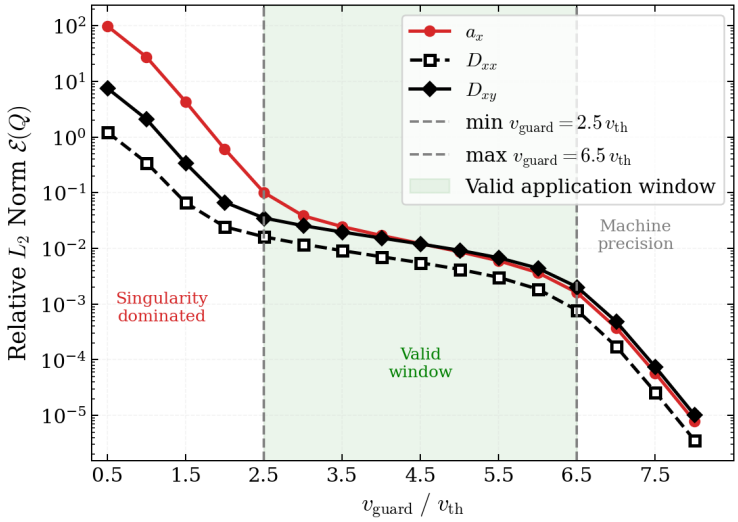

Figure 5.2: Relative L_2 norm $\mathcal{E}(Q)$ for a_x, D_{xx}, and D_{xy} as a function of v_{guard} for the loss-cone distribution (NV $= 32$, $\delta = 0.5\,v_{\text{th}}$). Dashed vertical lines mark the boundaries of the valid application window.

v_g/v_{th}	$\mathcal{E}(D_{xy})$ ratio	v_g/v_{th}	$\mathcal{E}(D_{xy})$ ratio	v_g/v_{th}	$\mathcal{E}(D_{xy})$ ratio
$0.5 \to 1.0$	3.601	$3.0 \to 3.5$	1.304	$5.5 \to 6.0$	1.540
$1.0 \to 1.5$	6.172	$3.5 \to 4.0$	1.281	$6.0 \to 6.5$	2.168
$1.5 \to 2.0$	5.014	$4.0 \to 4.5$	1.278	$6.5 \to 7.0$	4.183
$2.0 \to 2.5$	**1.923***	$4.5 \to 5.0$	1.299	$7.0 \to 7.5$	6.526
$2.5 \to 3.0$	1.371	$5.0 \to 5.5$	1.361	$7.5 \to 8.0$	7.254

*Recommended $v_{\text{guard}} = 2.5\,v_{\text{th}}$

Table 5.1: Successive ratios of $\mathcal{E}(D_{xy})$ from the guard radius sweep for the loss-cone distribution (NV $= 32$, $\delta = 0.5\,v_{\text{th}}$). The bold row marks the onset of the plateau and the recommended guard radius.

Figure 5.2 exhibits three qualitatively distinct regimes. For $v_{\text{guard}} \lesssim 2.0\,v_{\text{th}}$, all three quantities are singularity-dominated: $\mathcal{E}(a_x)$ reaches $\mathcal{O}(10^2)$ and $\mathcal{E}(D_{xy}) = \mathcal{O}(10^0\text{–}10^1)$, reflecting the divergence of the multipole series inside the bulk of f_1. Between 2.5 and $6.5\,v_{\text{th}}$ the residuals decrease monotonically, constituting the valid window in which the series is convergent and the correction physically meaningful. Beyond $6.5\,v_{\text{th}}$ the residuals descend steeply toward machine precision as the guard

is pushed past the effective support of f_1 and the tanh weight suppresses the correction almost entirely. The ratios in Table 5.1 confirm this structure: they are large and variable in the singularity-dominated regime, settle into a stable near-plateau between 2.5 and 6.5 v_{th}, and grow again beyond 6.5 v_{th}. The entry at $v_{\text{guard}} = 2.5\, v_{\text{th}}$ (ratio $= 1.923$) marks the onset of the plateau and is adopted as the recommended guard for the loss-cone distribution. The guard radii for the remaining distributions are selected by the same criterion and are listed in Table 5.2, to be applied in Section 5.3.

5.3 Relaxation Tests under the MC-DG-FPO

Section 5.2.2 establishes that the multipole correction reduces pointwise errors in the drag and diffusion coefficients at the velocity-space boundary and in the far field. The central question is whether this improvement propagates into more accurate relaxation dynamics when the FPO is evolved in time. To answer this, we compare the MA-DG-FPO introduced in Chapter 2 — which approximates H_s and G_s using the best-fit Maxwellian f_M everywhere in velocity space — against the MC-DG-FPO, which applies Algorithm 4 to accumulate the residual contribution of $f_1 = f_s - f_M$ onto the potentials beyond v_{guard}. Both are implementations of the same FPO, differing only in how the Rosenbluth potentials are approximated. The relaxation tests validate the MC-DG-FPO across four non-Maxwellian distributions, confirming that the correction does not disrupt smooth relaxation to the Maxwellian equilibrium and that moment conservation is preserved to machine precision.
The simulation parameters for each distribution are summarised in Table 5.2.

Distribution	NV	Domain	t_{end}	v_{guard}
Top-hat	16	$[-4, 4]\, v_{\text{th}}$	$1/\nu$	$3.0\, v_{\text{th}}$
Kappa	32	$[-8, 8]\, v_{\text{th}}$	$3/\nu$	$3.0\, v_{\text{th}}$
Loss-cone	32	$[-10, 10]\, v_{\text{th}}$	$2/\nu$	$3.0\, v_{\text{th}}$
Bump-on-tail	16	$[-8, 8]\, v_{\text{th}}$	$3/\nu$	$3.5\, v_{\text{th}}$

Table 5.2: Simulation parameters for the relaxation tests. All runs use the 1X3V configuration with $p = 2$ Serendipity basis, $\delta = 0.5\, v_{\text{th}}$, and collision frequency $\nu = n/(3\sqrt{\pi}\, v_{\text{th}}^3)$ with $n = \Gamma = v_{\text{th}} = 1.0$.

5.3.1 Top-Hat Distribution

The top-hat distribution, introduced in Section 1.4.1, is defined as a uniform distribution over a finite velocity cube,

$$f_{\text{TH}}(\mathbf{v}) = \begin{cases} 0.5 & |v_x| \le v_0,\ |v_y| \le v_0,\ |v_z| \le v_0, \\ 0 & \text{otherwise}, \end{cases} \tag{5.21}$$

where $v_0 = v_{\text{th}}$. It represents a large initial departure from equilibrium with a sharp discontinuity at $|v_i| = v_{\text{th}}$. Since the top-hat is isotropic, the residual $f_1 = f_{\text{TH}} - f_M$ has zero quadrupole and octupole moments by symmetry. The multipole correction therefore contributes no physically meaningful correction for this distribution, and the two operator implementations produce identical relaxation trajectories as shown in Figure 5.3, confirming that the MC-DG-FPO correctly reduces to the MA-DG-FPO when f_1 carries no anisotropic content.

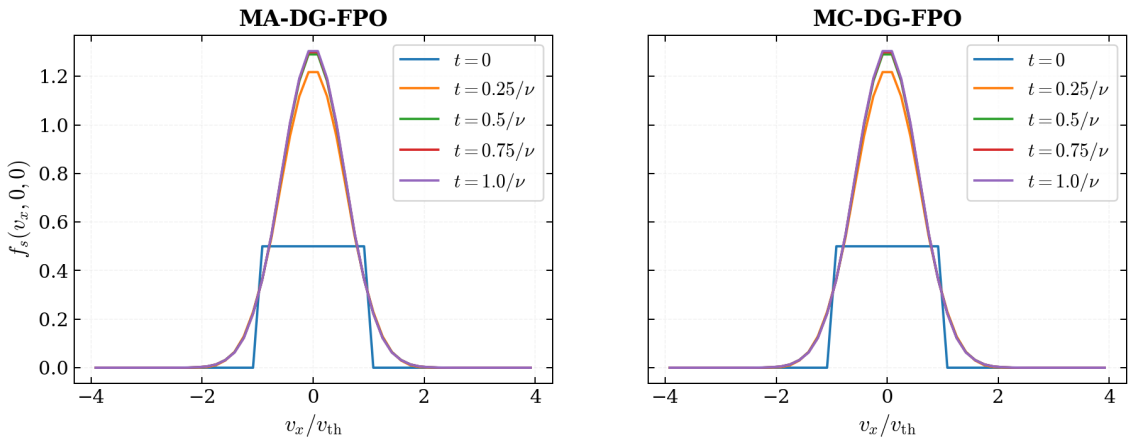

Figure 5.3: Relaxation of the top-hat distribution to a Maxwellian through FPO collisions. Left: MA-DG-FPO. Right: MC-DG-FPO. NV $= 16$, $p = 2$, domain $= [-4, 4]\, v_{\text{th}}$, $v_{\text{guard}} = 3.0\, v_{\text{th}}$, $t_{\text{end}} = 1/\nu$.

The domain-integrated moments M_0 and M_2 are conserved to machine precision by both operators throughout the simulation, with the small monotonic drift of $\mathcal{O}(10^{-13})$ arising from SSP-RK timestepper damping [15], as shown in Figure 5.4.

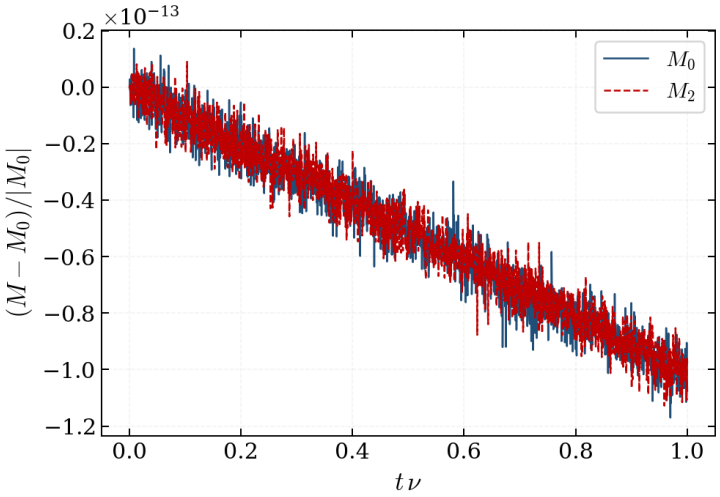

Figure 5.4: Domain-integrated M_0 and M_2 relative change vs time for the top-hat distribution under the MC-DG-FPO. NV $= 16$, $p = 2$, $t_{\text{end}} = 1/\nu$.

5.3.2 Kappa Distribution

The kappa distribution, introduced in Section 1.4.3, is defined as

$$f_\kappa(\mathrm{v}) = n_0 \left(\pi\theta^2\kappa\right)^{-3/2} \frac{\Gamma(\kappa+1)}{\Gamma(\kappa-1/2)} \left(1 + \frac{|\mathrm{v}|^2}{\kappa\,\theta^2}\right)^{-(\kappa+1)}, \tag{5.22}$$

where $\theta^2 = v_{\mathrm{th}}^2(2\kappa-3)/\kappa$ is the effective thermal speed and Γ denotes the gamma function. We use $\kappa = 2$, giving a suprathermal tail decaying as $v^{-2(\kappa+1)} = v^{-6}$, which is significantly slower than the decay of a Maxwellian.

Since the kappa distribution is isotropic, the residual $f_1 = f_\kappa - f_M$ has zero dipole moment by symmetry. The leading correction enters through the quadrupole and higher moments of the suprathermal tail, which are non-zero because the kappa tail decays more slowly than the best-fit Maxwellian — precisely the far-field region where the multipole correction is active.

Figure 5.5 shows the relaxation of f_κ under both operators. The initial distribution is visibly non-Maxwellian, with a narrow core and oscillations at the tails. As collisions redistribute energy from the suprathermal tail into the bulk, the distribution broadens progressively toward the Maxwellian. The two operator implementations produce nearly identical relaxation trajectories, confirming that the multipole correction does not disrupt the relaxation dynamics for an isotropic distribution. However, a stronger method is required to eliminate the oscillations in the suprathermal tail, which will be discussed in Chapter 6.

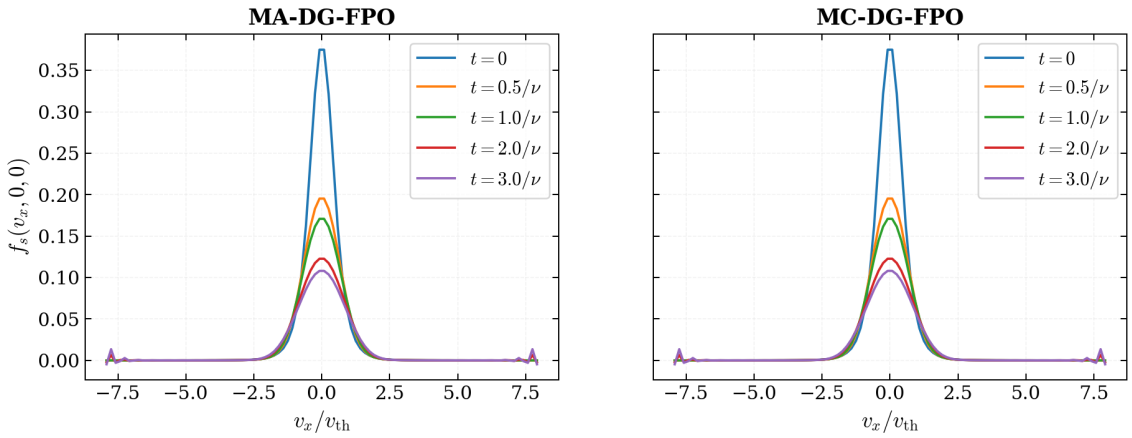

Figure 5.5: Relaxation of the kappa distribution ($\kappa = 2$) to a Maxwellian through FPO collisions. Left: MA-DG-FPO. Right: MC-DG-FPO. NV $= 32$, $p = 2$, domain $= [-8, 8]\, v_{\text{th}}$, $v_{\text{guard}} = 3.0\, v_{\text{th}}$, $t_{\text{end}} = 3/\nu$.

The domain-integrated moments M_0 and M_2 are conserved to $\mathcal{O}(10^{-13})$ throughout the simulation, as shown in Figure 5.6.

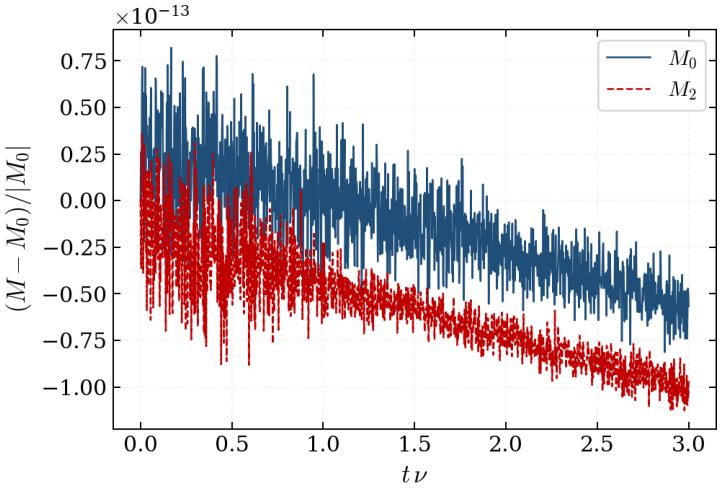

Figure 5.6: Domain-integrated M_0 and M_2 relative change vs time for the kappa distribution under the MC-DG-FPO. NV $= 32$, $p = 2$, $t_{\text{end}} = 3/\nu$.

5.3.3 Loss-Cone Distribution

The loss-cone distribution, defined in Eq. (5.19) and discussed in Section 5.2.2, is the anisotropic test case where the multipole correction is expected to have a measurable

effect. Its characteristic pressure anisotropy — $M^{(2)}_{1,xx} \neq M^{(2)}_{1,zz}$ — produces non-trivial quadrupole and octupole contributions to $f_1 = f_{LC} - f_M$ that the best-fit Maxwellian cannot capture, feeding directly into errors in the off-diagonal diffusion coefficients D_{xy} and D_{xz} as demonstrated in Section 5.2.2.

Figure 5.7 shows the relaxation of f_{LC} under both operator implementations. The characteristic double-peaked structure at $t = 0$ with the hole at $v_x = 0$ progressively fills as collisions drive the distribution toward equilibrium. By $t = 2/\nu$ the distribution has relaxed to a single-peaked Maxwellian. Both implementations produce identical relaxation trajectories, confirming that the multipole correction does not disrupt the relaxation dynamics even for an anisotropic distribution with non-trivial higher-order moment content.

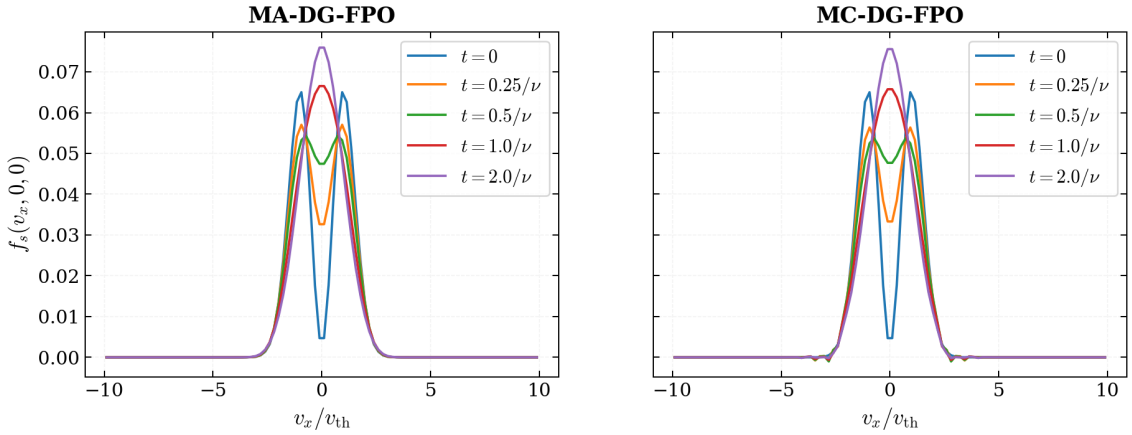

Figure 5.7: Relaxation of the loss-cone distribution to a Maxwellian through FPO collisions. Left: MA-DG-FPO. Right: MC-DG-FPO. NV $= 32$, $p = 2$, domain $= [-10, 10]\, v_{\text{th}}$, $v_{\text{guard}} = 3.0\, v_{\text{th}}$, $t_{\text{end}} = 2/\nu$.

The domain-integrated moments M_0 and M_2 are conserved to $\mathcal{O}(10^{-14})$ throughout the simulation, as shown in Figure 5.8.

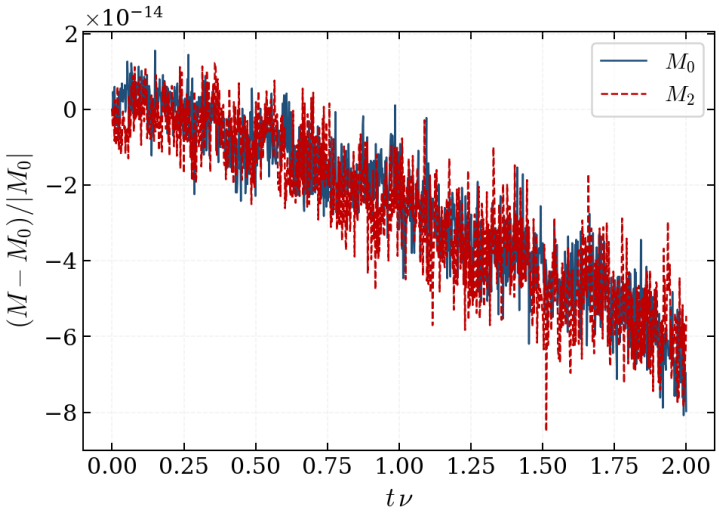

Figure 5.8: Domain-integrated M_0 and M_2 relative change vs time for the loss-cone distribution under the MC-DG-FPO. NV $= 32$, $p = 2$, $t_{\text{end}} = 2/\nu$.

5.3.4 Bump-on-Tail Distribution

The bump-on-tail distribution, introduced in Section 1.4.2, consists of a bulk Maxwellian background with an additional beam population displaced in velocity space. Following [15], we decompose the distribution as

$$f_{BOT}(\mathrm{v}) = f_M(\mathrm{v}) + f_b(\mathrm{v}), \tag{5.23}$$

where the background Maxwellian and beam components are given by

$$f_M(\mathrm{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\text{th}}\right)^3} \exp\left(-\frac{|\mathrm{v}|^2}{2v_{\text{th}}^2}\right), \tag{5.24}$$

$$f_b(\mathrm{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\text{th,b}}\right)^3} \frac{A_b^2}{(v_x - u_{x,b})^2 + v_y^2 + v_z^2 + S_b^2} \exp\left(-\frac{(v_x - u_{x,b})^2 + v_y^2 + v_z^2}{2v_{\text{th,b}}^2}\right), \tag{5.25}$$

with $n = v_{\text{th}} = 1$, $A_b = \sqrt{0.15}$, $u_{x,b} = 4\,v_{\text{th}}$, $S_b = 0.14$, and $v_{\text{th,b}} = 3\,v_{\text{th}}$. The beam is displaced to $v_x = 4\,v_{\text{th}}$, producing a large asymmetry in the residual $f_1 = f_{BOT} - f_M$. Since f_M matches the total first moment of f_{BOT}, the dipole moment of f_1 vanishes by construction and is enforced explicitly in the implementation to suppress numerical projection noise. The leading non-trivial correction therefore enters at quadrupole order through the beam's anisotropic pressure contribution.

Figure 5.9 shows the relaxation of f_{BOT} under both operator implementations on a logarithmic scale to resolve the beam component at $v_x \approx 4\,v_{\text{th}}$. The beam peak decays progressively as collisions scatter beam particles into the bulk, relaxing toward the Maxwellian background by $t = 3/\nu$. Both FPO implementations show clean monotonic decay of the beam. However, the MC-DG-FPO shows inaccuracies beyond $v_x \approx -4\,v_{\text{th}}$ over time, suggesting that the correction becomes less reliable as f_1 changes character for strongly asymmetric distributions where the far-field condition is harder to satisfy uniformly across the full velocity domain.

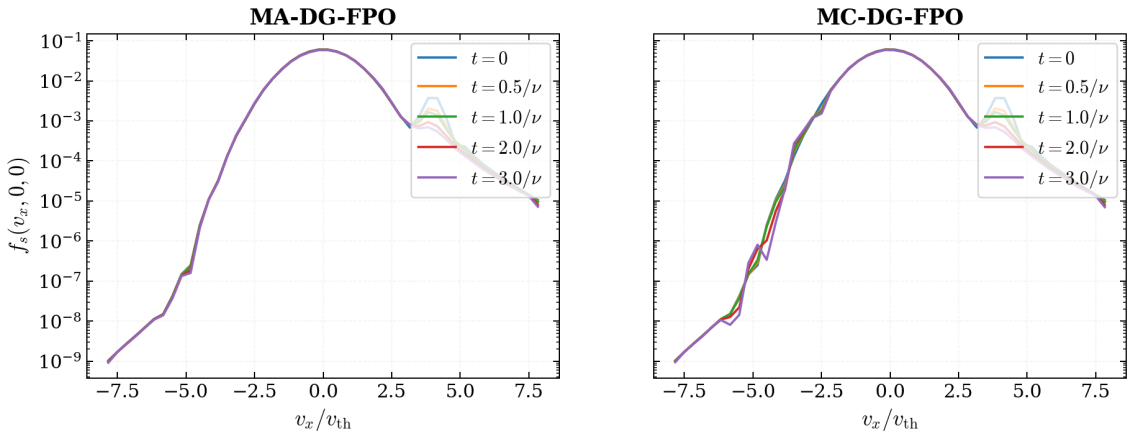

Figure 5.9: Relaxation of the bump-on-tail distribution to a Maxwellian through FPO collisions (logarithmic scale). Left: MA-DG-FPO. Right: MC-DG-FPO. NV $= 16$, $p = 2$, domain $= [-8, 8]\,v_{\text{th}}$, $v_{\text{guard}} = 3.5\,v_{\text{th}}$, $t_{\text{end}} = 3/\nu$.

The domain-integrated moments M_0, M_{1x}, and M_2 are conserved to $\mathcal{O}(10^{-14})$ by the MC-DG-FPO throughout the simulation, as shown in Figure 5.10.

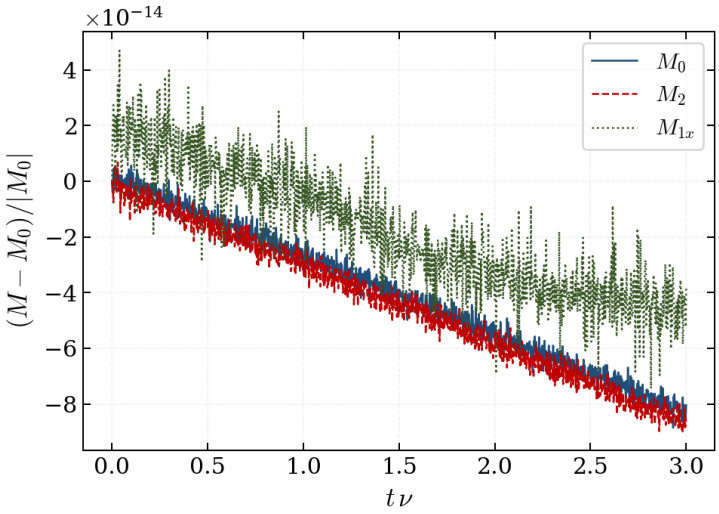

Figure 5.10: Domain-integrated M_0, M_{1x}, and M_2 relative change vs time for the bump-on-tail distribution under the MC-DG-FPO. NV $= 16$, $p = 2$, $t_{\text{end}} = 3/\nu$.

5.4 Summary of Chapter 5

This chapter develops an analytic representation of the Rosenbluth potentials using a multipole expansion and validates the MC-DG-FPO across a suite of non-Maxwellian distributions.

We begin by deriving the multipole expansion of H_s and G_s up to octupole order ($\ell = 3$) using the gravitational analogy of the Rosenbluth potentials and the symmetric trace-free tensor formalism. The resulting expressions for the drag coefficient $\mathbf{a}_s$ and diffusion tensor $\mathbf{D}_s$ are computed directly from the Cartesian moments already available in the DG-FPO update, making the expansion directly compatible with the existing moment infrastructure in GKEYLL without additional overhead.

We then establish the far-field application framework, addressing the fundamental restriction that the asymptotic multipole series converges only outside the bulk support of $f_1 = f_s - f_M$. The correction is applied beyond a guard radius v_{guard} via a smooth tanh ramp, with a dual condition mask combining the far-field guard with a significance check to prevent amplification of numerical noise. The guard radius

for each distribution is determined empirically from a plateau sweep of the relative L_2 norm of the correction. For the loss-cone distribution, the correction reduces pointwise errors in the off-diagonal diffusion coefficient D_{xy} in the far-field shell between 2.5 and 6.5 v_{th}, where the multipole series is convergent and the pressure anisotropy of f_1 produces a correction that the Maxwellian approximation cannot capture.

Finally, the relaxation tests confirm that for isotropic distributions — the top-hat and kappa — the multipole correction produces no visible difference from the MA-DG-FPO, as expected when f_1 carries no anisotropic content. For the loss-cone, both operators produce identical relaxation trajectories and conserve M_0 and M_2 to $\mathcal{O}(10^{-14})$, confirming that the correction does not disrupt the relaxation dynamics even for an anisotropic distribution with non-trivial higher-order moment content. For the bump-on-tail, the correction reproduces the beam absorption dynamics of the MA-DG-FPO but shows inaccuracies in the far negative tail, highlighting a limitation of the current implementation for strongly asymmetric distributions where the far-field condition is harder to satisfy uniformly across the full velocity domain. In all cases, moment conservation is preserved to machine precision, confirming that the multipole correction does not disrupt the conservation properties of the underlying FPO scheme.

Chapter 6

Introducing the Extended DG-FPO (X-DG-FPO)

Chapters 4 and 5 developed and validated two complementary Poisson solving schemes for the Rosenbluth potentials: the second-order Richardson iteration, which provides a source-independent numerical solver for any arbitrary distribution function f_s, and the multipole expansion, which provides an analytic representation for the error in the potential solves in the far field and at the boundary. This chapter integrates both schemes into the existing DG-FPO algorithm (MA-DG-FPO) in GKEYLL — producing the *Extended DG-FPO* (X-DG-FPO). The Maxwellian-approximation DG-FPO (MA-DG-FPO), introduced in Chapter 2, serves as the baseline against which the X-DG-FPO is assessed.

Section 6.1 presents the full algorithm and describes how the two schemes are integrated into the MA-DG-FPO infrastructure. Section 6.2 validates the X-DG-FPO through relaxation tests across the four non-Maxwellian distributions in this thesis, comparing against the MA-DG-FPO and the LBO. Section 6.3 presents a thermalization test on a bi-Maxwellian initial condition, comparing the X-DG-FPO against the MA-DG-FPO, LBO, and BGK operators on their ability to drive an

anisotropic distribution towards thermal equilibrium.

6.1 The Algorithm

The X-DG-FPO modifies the existing MA-DG-FPO algorithm in Chapter 2 (Algorithm 1) by replacing Step 2 with a Richardson iteration for the interior Poisson solve, combined with a multipole boundary correction. All other steps remain unchanged. Algorithm 5 presents the full procedure.

Algorithm 5: Extended DG-FPO (X-DG-FPO) Collision Update (per timestep)

Input : Distribution function f_h^j on DG basis; H_{prev}, G_{prev} from previous timestep

Output: Updated f_h^j after one collision timestep

Step 1: Compute velocity-space moments M_0, M_1, M_2 and primitive moments n_s, $\mathbf{u}_s$, $v_{\text{th},s}^2$

Step 2: **Extended potential solve**

begin

Warm-start: $H_s \leftarrow H_{\text{prev}}$, $G_s \leftarrow G_{\text{prev}}$ as initial iterates

Richardson iteration $\rightarrow H_s$, G_s on interior velocity domain

if *final timestep* **then**

Multipole boundary correction $\rightarrow$ accumulate H_1, G_1 onto H_s, G_s in the outermost cell layer ($v_{\text{guard}} = V_{\max} - \Delta v$)

end

Project H_s, G_s to surface basis in skin cells

$H_{\text{prev}} \leftarrow H_s$, $G_{\text{prev}} \leftarrow G_s$ as initial iterates for next timestep

end

Step 3: Compute $\mathbf{a}_h$, $\mathbf{D}_h$ and surface projections over all phase-space cells

Step 4: Compute boundary corrections and moments from conservation correction equations

Step 5: Solve for $\delta\mathbf{a}_h$, δD_h; apply to $\mathbf{a}_h$, $\mathbf{D}_h$

Step 6: Compute sign of corrected drag for Lax-Friedrichs flux

Step 7: Advance FPO update over all phase-space cells

Step 8: Compute CFL; return to Vlasov-Maxwell loop

The three extensions to Step 2 are:

Richardson interior solve. The Maxwellian projection is replaced by Algorithm 3, which solves Eqs. (1.24)–(1.25) directly from the arbitrary f_s without assuming $f_s \approx f_M$.

Warm start. Rather than initializing with $F^0 = 0$ as in Chapter 4, the converged solution from the previous timestep is used as the initial iterate. Since H_s and G_s change slowly between successive timesteps, this significantly reduces the iteration count per update. Working arrays are pre-allocated at initialization and reused every timestep. This impacts computational time as will be discussed in Section 6.2.5.

Multipole boundary correction. Richardson provides an accurate solution in the interior velocity domain; the residual $H_s - H_{\text{rich}}$ is nonzero only near the domain boundary where the Dirichlet conditions introduce error. The guard radius is therefore set to $v_{\text{guard}} = V_{\max} - \Delta v$, where $\Delta v = 2V_{\max}/\text{NV}$, confining the multipole correction to the outermost cell layer. To avoid interference with the Richardson solve during time evolution, the correction is applied only at the final timestep, after the distribution in the interior has fully relaxed.

6.2 Relaxation Tests under the X-DG-FPO

With the X-DG-FPO algorithm established, we now validate it across the four non-Maxwellian distributions introduced in Chapter 1, comparing three operators: the MA-DG-FPO, the X-DG-FPO, and the LBO. The expected physics result is that the LBO should relax the distribution to equilibrium the fastest due to its velocity independent collision frequency. The X-DG-FPO and the MA-DG-FPO should relax at similar rates with the X-DG-FPO being more accurate. Simulation parameters are identical to those of Chapter 5 and are summarised in Table 6.1.

Distribution	NV	Domain	t_{end}	v_{guard}
Top-hat	16	$[-4, 4]\, v_{\text{th}}$	$1/\nu$	$3.5\, v_{\text{th}}$
Kappa	32	$[-8, 8]\, v_{\text{th}}$	$3/\nu$	$7.5\, v_{\text{th}}$
Loss-cone	32	$[-10, 10]\, v_{\text{th}}$	$2/\nu$	$9.375\, v_{\text{th}}$
Bump-on-tail	16	$[-8, 8]\, v_{\text{th}}$	$3/\nu$	$7.0\, v_{\text{th}}$

Table 6.1: Simulation parameters for the X-DG-FPO relaxation tests. All runs use 1X3V, $p = 2$ Serendipity basis, and $\nu = n/(3\sqrt{\pi}\, v_{\text{th}}^3)$ with $n = v_{\text{th}} = 1$. The guard radius $v_{\text{guard}} = V_{\text{max}} - \Delta v$ in all cases, restricting the multipole correction to the outermost cell layer.

6.2.1 Top-Hat Distribution

The top-hat distribution, as introduced in Section 1.4.1,

$$f_{\text{TH}}(\mathrm{v}) = \begin{cases} 0.5 & |v_x| \le v_0,\ |v_y| \le v_0,\ |v_z| \le v_0, \\ 0 & \text{otherwise}, \end{cases} \tag{6.1}$$

where $v_0 = v_{\text{th}} = 1.0$. Figure 6.1 shows the relaxation of f_{TH} under all three operators.

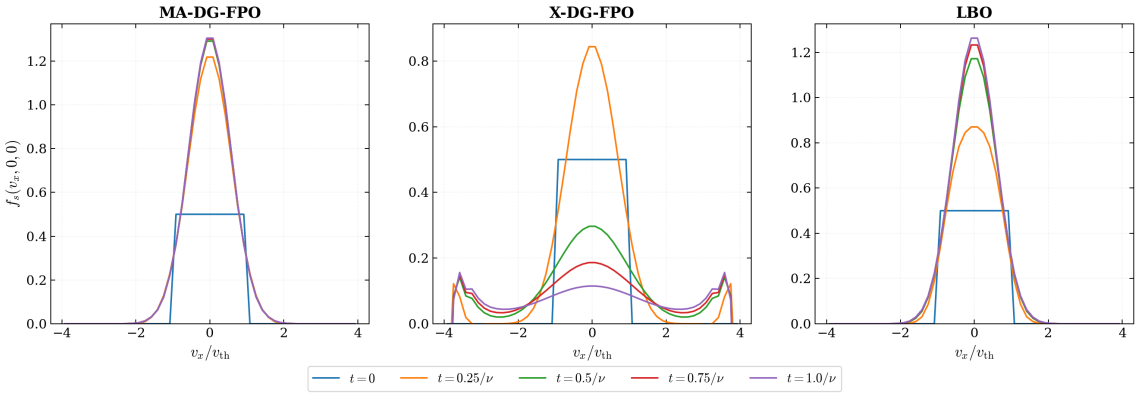

Figure 6.1: Relaxation of the top-hat distribution. NV $= 16$, $p = 2$, domain $= [-4, 4]\, v_{\text{th}}$, $v_{\text{guard}} = 3.5\, v_{\text{th}}$, $t_{\text{end}} = 1/\nu$.

The MA-DG-FPO and LBO both produce clean, monotonic relaxation toward the Maxwellian equilibrium. The X-DG-FPO, however, becomes unstable for the top-hat distribution. The Gibbs-like oscillations that develop at the sharp discontinuity at $|v_i| = v_{\text{th}}$ are amplified by the Richardson iteration over successive timesteps, causing the scheme to diverge. This is a fundamental limitation of the current Richardson solver implementation for distributions with sharp discontinuities. Replacing the RDG Laplacian operator with an FEM Laplacian, or reducing the damping of the Richardson scheme might mitigate this problem [31].

The domain-integrated moments M_0 and M_2 are conserved to $\mathcal{O}(10^{-13})$ by the X-DG-FPO throughout the simulation despite the collapse of the distribution function, as shown in Figure 6.2. This reflects the fact that the conservation correction (Steps 4–5 of Algorithm 5) operates independently of the potential solve and enforces moment conservation discretely regardless of the quality of the Poisson solution.

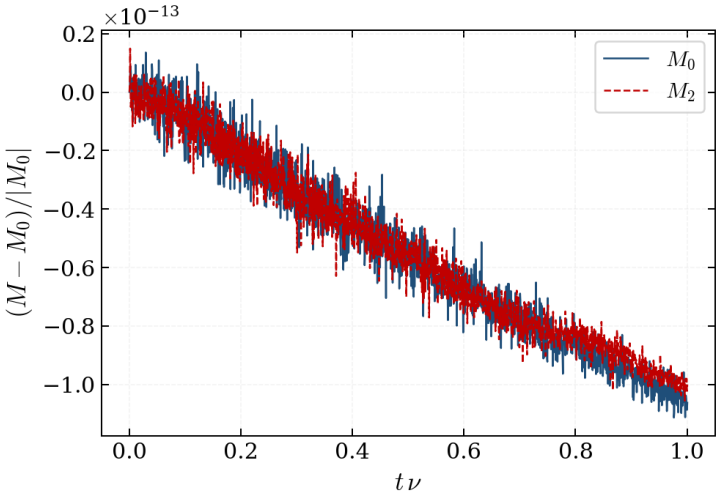

Figure 6.2: Domain-integrated M_0 and M_2 relative change vs time for the top-hat distribution under the X-DG-FPO. NV $= 16$, $p = 2$, $t_{\text{end}} = 1/\nu$.

6.2.2 Kappa Distribution

The kappa distribution, introduced in Section 1.4.3, is defined as

$$f_\kappa(\mathrm{v}) = n_0 \left(\pi\theta^2\kappa\right)^{-3/2} \frac{\Gamma(\kappa+1)}{\Gamma(\kappa-1/2)} \left(1 + \frac{|\mathrm{v}|^2}{\kappa\,\theta^2}\right)^{-(\kappa+1)}, \tag{6.2}$$

where $\theta^2 = v_{\mathrm{th}}^2(2\kappa-3)/\kappa$ and Γ denotes the gamma function. We use $\kappa = 2$ with $n_0 = v_{\mathrm{th}} = 1.0$. Figure 6.3 shows the relaxation of f_κ under all three operators on both linear and logarithmic scales.

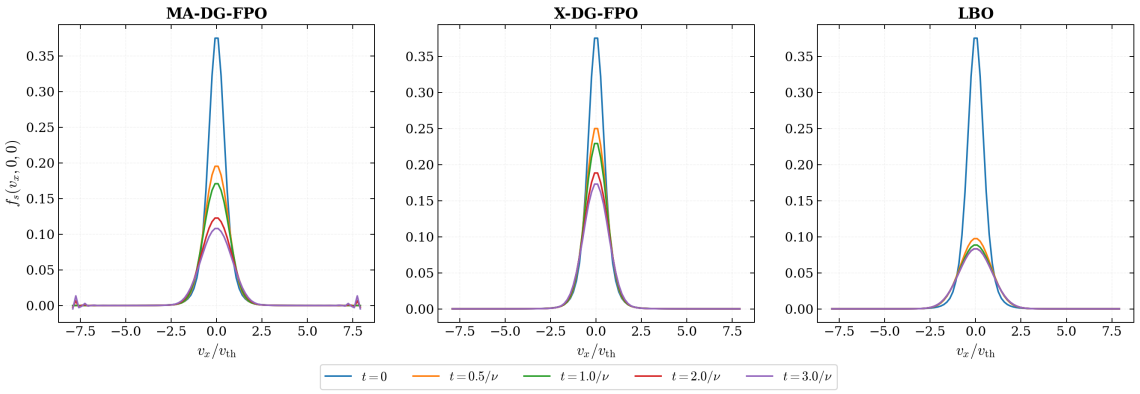

Figure 6.3: Relaxation of the kappa distribution ($\kappa = 2$). NV $= 32$, $p = 2$, domain $= [-8, 8]\, v_{\mathrm{th}}$, $v_{\mathrm{guard}} = 7.5\, v_{\mathrm{th}}$, $t_{\mathrm{end}} = 3/\nu$.

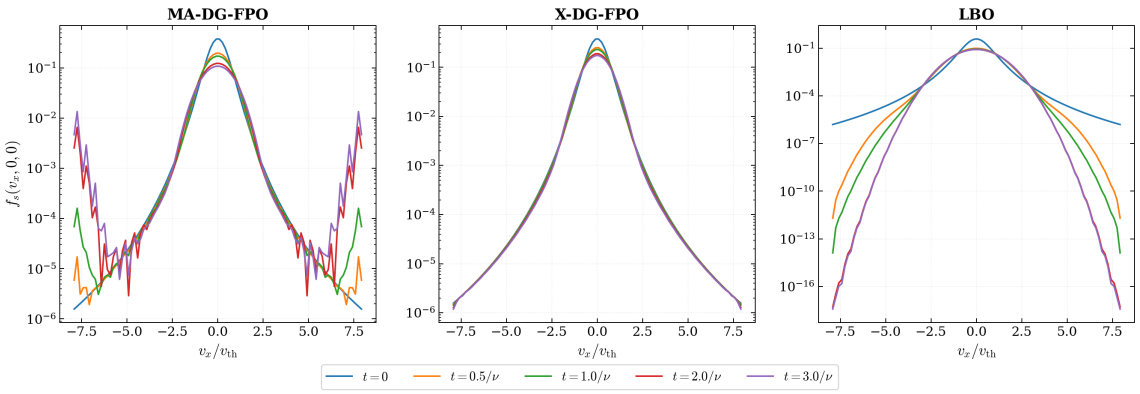

Figure 6.4: Same as Figure 6.3 on a logarithmic scale, revealing the suprathermal tail behavior.

The X-DG-FPO eliminates the heavy-tail oscillations that persist under the MA-

DG-FPO, producing a cleaner relaxation toward the Maxwellian equilibrium as confirmed by the logarithmic scale plot. The LBO relaxes fastest due to its velocity-independent collision frequency overestimating the scattering rate at suprathermal velocities, producing minor tail oscillations in the process. The X-DG-FPO correctly handles the slowly decaying suprathermal tail of the kappa distribution, confirming that the Richardson solver is more accurate than the Maxwellian approximation for non-Maxwellian sources.

The domain-integrated moments M_0 and M_2 are conserved to $\mathcal{O}(10^{-13})$ throughout the simulation, as shown in Figure 6.5.

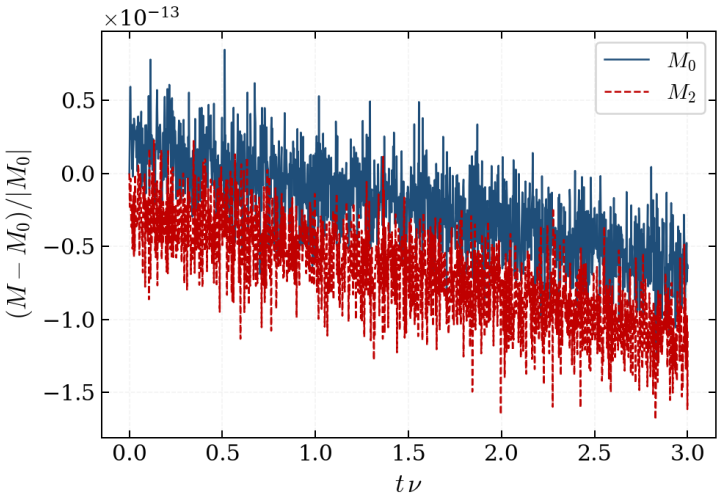

Figure 6.5: Domain-integrated M_0 and M_2 relative change vs time for the kappa distribution under the X-DG-FPO. NV $= 32$, $p = 2$, $t_{\text{end}} = 3/\nu$.

6.2.3 Loss-Cone Distribution

The loss-cone distribution, introduced in Section 1.4.4, is defined as

$$f_{LC}(\mathrm{v}) = \frac{n_0}{\pi^{3/2} v_{\text{th}}^5} v_\perp^2 \exp\left(-\frac{v_\perp^2 + v_z^2}{v_{\text{th}}^2}\right), \tag{6.3}$$

where $v_\perp^2 = v_x^2 + v_y^2$ and $n_0 = v_{\text{th}} = 1.0$. Figure 6.6 shows the relaxation of f_{LC} under all three operators.

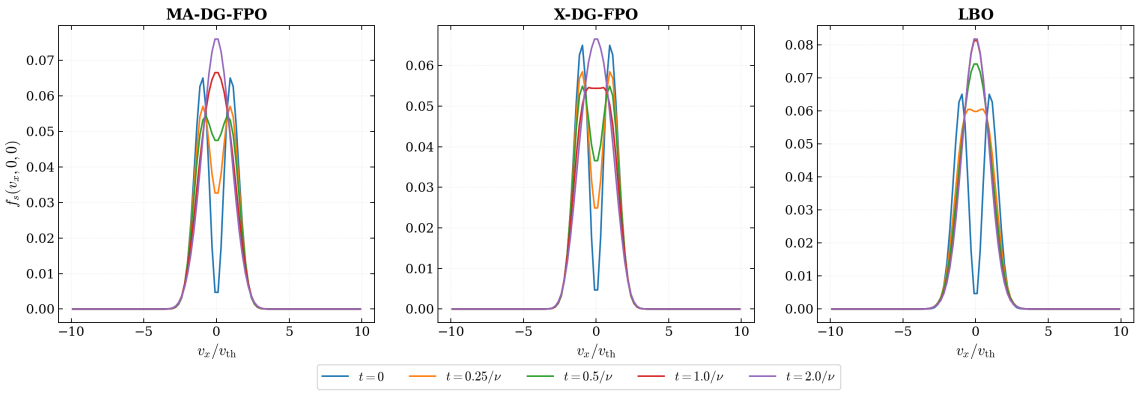

Figure 6.6: Relaxation of the loss-cone distribution. NV $= 32$, $p = 2$, domain $= [-10, 10]\, v_{\mathrm{th}}$, $v_{\mathrm{guard}} = 9.375\, v_{\mathrm{th}}$, $t_{\mathrm{end}} = 2/\nu$.

All three operators successfully relax the loss-cone to a Maxwellian equilibrium, progressively filling the hole in the characteristic double-peaked structure. The LBO relaxes fastest, consistent with its velocity-independent collision frequency. The MA-DG-FPO and X-DG-FPO produce nearly identical relaxation trajectories, confirming that the Richardson solver does not disrupt the relaxation dynamics for anisotropic distributions.

The domain-integrated moments M_0 and M_2 are conserved to $\mathcal{O}(10^{-14})$ throughout the simulation, as shown in Figure 6.7.

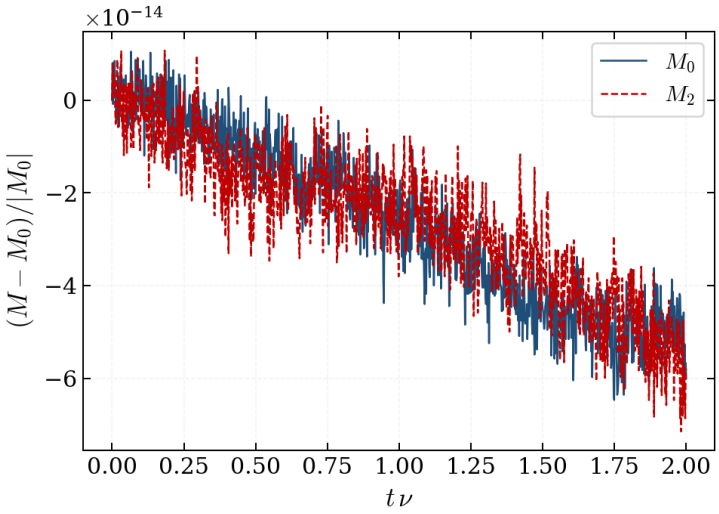

Figure 6.7: Domain-integrated M_0 and M_2 relative change vs time for the loss-cone distribution under the X-DG-FPO. NV $= 32$, $p = 2$, $t_{\text{end}} = 2/\nu$.

6.2.4 Bump-on-Tail Distribution

The bump-on-tail distribution, introduced in Section 1.4.2, consists of a bulk Maxwellian background with an additional beam population displaced in velocity space,

$$f_{BOT}(\mathrm{v}) = f_M(\mathrm{v}) + f_b(\mathrm{v}), \tag{6.4}$$

where

$$f_M(\mathrm{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\text{th}}\right)^3} \exp\left(-\frac{|\mathrm{v}|^2}{2v_{\text{th}}^2}\right), \tag{6.5}$$

$$f_b(\mathrm{v}) = \frac{n}{\left(\sqrt{2\pi}\, v_{\text{th,b}}\right)^3} \frac{A_b^2}{(v_x - u_{x,b})^2 + v_y^2 + v_z^2 + S_b^2} \exp\left(-\frac{(v_x - u_{x,b})^2 + v_y^2 + v_z^2}{2v_{\text{th,b}}^2}\right), \tag{6.6}$$

with $n = v_{\text{th}} = 1$, $A_b = \sqrt{0.15}$, $u_{x,b} = 4\, v_{\text{th}}$, $S_b = 0.14$, and $v_{\text{th,b}} = 3\, v_{\text{th}}$. Figure 6.8 shows the relaxation of f_{BOT} under all three operators on a logarithmic scale to

resolve the beam component at $v_x \approx 4\,v_{\text{th}}$.

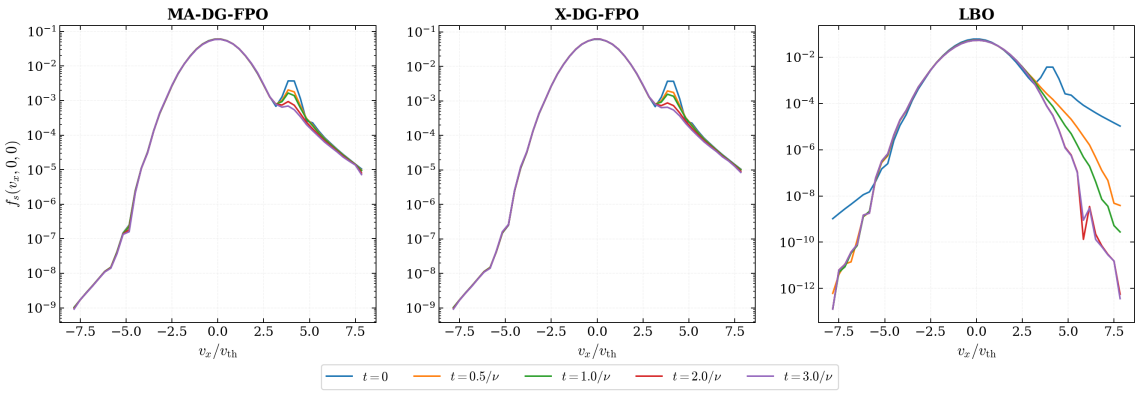

Figure 6.8: Relaxation of the bump-on-tail distribution (logarithmic scale). NV $= 16$, $p = 2$, domain $= [-8, 8]\,v_{\text{th}}$, $v_{\text{guard}} = 7.0\,v_{\text{th}}$, $t_{\text{end}} = 3/\nu$.

The MA-DG-FPO and the X-DG-FPO both show a clean and monotonic decay of the beam peak as collisions scatter beam particles into the bulk, with nearly identical relaxation trajectories confirming that the Richardson solver handles strongly asymmetric distributions correctly. The LBO panel shows a qualitatively different tail with oscillatory structure, reflecting the velocity-independent collision frequency overestimating the scattering rate at $v_x \approx 4\,v_{\text{th}}$.

Figure 6.9 shows the evolution of the domain-integrated entropy $S = -\int f_s \ln f_s \, d^3v$ under the X-DG-FPO. As collisions drive f_{BOT} toward the Maxwellian equilibrium, entropy increases monotonically, which is consistent with Boltzmann's H-theorem established in Section 1.3.2. We notice that the slope of the curve is decreasing which suggests that as the distribution approaches Maxwellian equilibrium, it starts to plateau [15]. This monotonic behavior throughout the simulation confirms that the X-DG-FPO is stable for the bump-on-tail distribution.

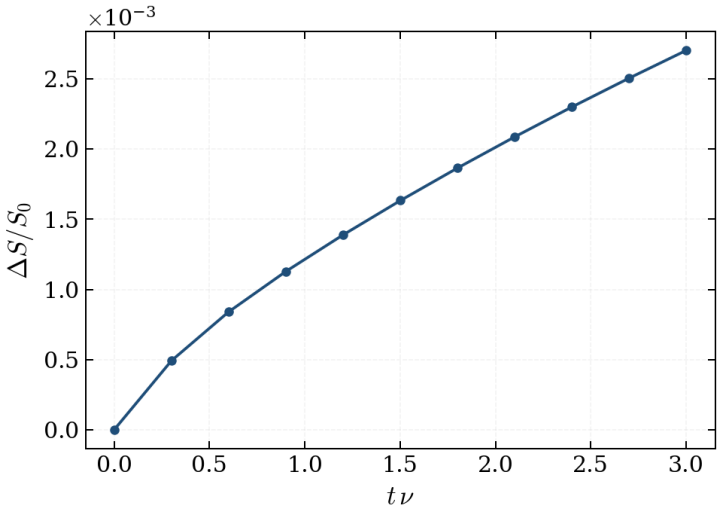

Figure 6.9: Domain-integrated entropy $\Delta S/S_0$ vs time for the bump-on-tail distribution under the X-DG-FPO. NV $= 16$, $p = 2$, $t_{\text{end}} = 3/\nu$.

The domain-integrated moments M_0, M_{1x}, and M_2 are conserved to $\mathcal{O}(10^{-13})$ throughout the simulation, as shown in Figure 6.10.

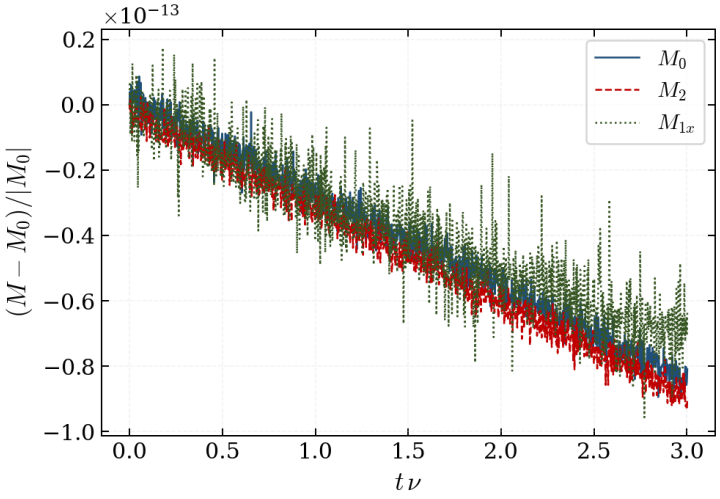

Figure 6.10: Domain-integrated M_0, M_{1x}, and M_2 relative change vs time for the bump-on-tail distribution under the X-DG-FPO. NV $= 16$, $p = 2$, $t_{\text{end}} = 3/\nu$.

6.2.5 Compute Time Analysis (GPU)

Table 6.2 summarises the total wall-clock time for each operator across all four distributions. All runs were performed on a single GPU.

Distribution	NV	LBO (s)	MA-DG-FPO (s)	X-DG-FPO (s)
Top-hat	16	3.43	144.15	433.6*
BOT	16	13.79	100.92	236.9
Kappa	32	99.28	608.21	1909.4
Loss-cone	32	64.69	508.92	1492.9

*26 RK stage-2 failures due to Richardson instability on the discontinuous top-hat source.

Table 6.2: Total wall-clock time on a single GPU for the relaxation tests using LBO, MA-DG-FPO, and X-DG-FPO across all four distributions.

Three results stand out. First, the LBO is the fastest operator by a large margin across all distributions, as it requires no Poisson solve per timestep. Second, the X-DG-FPO is consistently 2–3$\times$ slower than the MA-DG-FPO, reflecting the cost of the Richardson iteration per timestep even with the warm start. Third, the top-hat X-DG-FPO accumulated 26 RK stage-2 failures, reflecting the Richardson instability for the discontinuous top-hat source discussed in Section 6.2.1.

The X-DG-FPO compute cost is dominated by the Richardson iteration, which scales linearly with NV as established in Chapter 4. The primary path to reducing this cost is the adoption of a FEM discrete Laplacian for the Richardson solver, which could allow a less stringent pseudo-timestep constraint and reduce the iteration count per timestep.

6.3 Thermalization Test

As a further validation of the X-DG-FPO, we test its ability to drive an anisotropic distribution toward thermal equilibrium. The bi-Maxwellian distribution provides a controlled benchmark for this purpose: its temperature components T_{xx}, T_{yy}, T_{zz} are set to unequal values and all collision operators must drive them toward a common equilibrium temperature while conserving total energy [38]. We define the bi-Maxwellian as

$$f_{\mathrm{BM}}(\mathrm{v}) = \frac{n_0}{(2\pi)^{3/2} v_{tx} v_{ty} v_{tz}} \exp\left(-\frac{v_x^2}{2v_{tx}^2} - \frac{v_y^2}{2v_{ty}^2} - \frac{v_z^2}{2v_{tz}^2} \right), \tag{6.7}$$

with $v_{tx} = v_{tz} = v_{\mathrm{th}}$ and $v_{ty} = 2\,v_{\mathrm{th}}$, giving $T_{xx} = T_{zz} = 1$ and $T_{yy} = 4$ in normalized units with $n_0 = \Gamma = v_{\mathrm{th}} = 1.0$. The equilibrium temperature is $T_{\mathrm{eq}} = (T_{xx} + T_{yy} + T_{zz})/3 = 2$. Four operators are compared in this thermalization test: LBO, BGK, MA-DG-FPO, and X-DG-FPO, all using the same collision frequency $\nu = n_0 \Gamma / (3\sqrt{\pi}\, v_{\mathrm{th}}^3)$ and run to $t = 3\,t_{\mathrm{coll}}$ where $t_{\mathrm{coll}} = 1/\nu$ on a NV $= 16$ velocity grid with domain $[-5,\, 5]\, v_{\mathrm{th}}$ in each direction.

Figure 6.11 shows T_{xx} (solid) and T_{yy} (dashed) as a function of t/t_{coll} for all four operators.

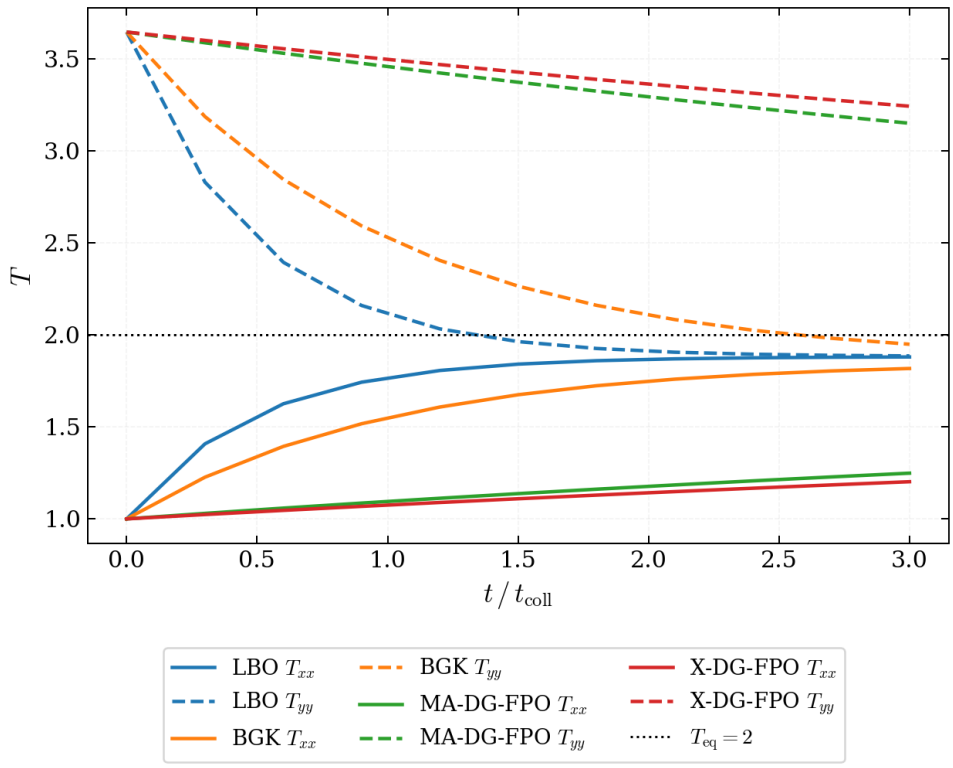

Figure 6.11: Temperature relaxation of the bi-Maxwellian distribution. Solid lines show T_{xx}; dashed lines show T_{yy}. Equilibrium temperature $T_{\text{eq}} = 2$. NV $= 16$, $p = 2$, domain $= [-5, 5]\, v_{\text{th}}$, $t_{\text{end}} \approx 2.7\, t_{\text{coll}}$.

The plot shows that the X-DG-FPO and the MA-DG-FPO produce nearly identical relaxation trajectories with X-DG-FPO doing slightly better. This confirms that the Richardson solver produces correct potentials for a near-Maxwellian distribution and that the X-DG-FPO preserves the thermalization dynamics of the underlying algorithm.

LBO and BGK relax significantly faster than the FPO operators as they overestimate the collision rate. The FPO operators correctly preserve the slower relaxation rate at high velocities, demonstrating the physical advantage of a fully velocity-dependent collision operator for anisotropic distributions.

6.4 Summary of Chapter 6

This chapter presents the culmination of all the novel methods and results discussed in this thesis. We integrated the second-order Richardson iteration and multipole expansions with the existing DG-FPO algorithm (MA-DG-FPO), to produce the X-DG-FPO algorithm. The algorithm replaces Step 2 of the MA-DG-FPO with a Richardson interior solve, a warm start from the previous timestep solution, and a multipole boundary correction restricted to the outermost cell layer.

The relaxation tests confirm that the X-DG-FPO eliminates the heavy-tail oscillations as seen in the Kappa distribution better than the MA-DG-FPO and LBO. However, it is not able to tackle non-Maxwellian distributions with discontinuities, such as the top-hat because of the implementation of the RDG Laplacian operator in its Richardson solver. Despite this, the X-DG-FPO and the MA-DG-FPO produce identical relaxation trajectories for the loss-cone and bump-on-tail distributions. Moreover, moment conservation is preserved to machine precision for every relaxation test, including top-hat. The LBO relaxes faster than both FPO operators across all distributions, consistent with its velocity-independent collision frequency overestimating the scattering rate at high velocities relative to the v^{-3} scaling of the Coulomb collision frequency.

The thermalization test confirms that the X-DG-FPO produces nearly identical if not better temperature relaxation trajectories to the MA-DG-FPO for a bi-Maxwellian initial condition, validating that the Richardson solver correctly reproduces the MA-DG-FPO result for near-Maxwellian distributions. The LBO and BGK operators relax significantly faster for this test as well.

In terms of compute time, the X-DG-FPO is an expensive algorithm even with warm start implemented, which means future work must be done to optimize it, including the replacement of the RDG Laplacian operator.

Chapter 7

Summary and Conclusion

The Fokker-Planck collision operator is the most physically complete model for Coulomb collisions in a plasma, capturing the velocity-dependent drag and diffusion that govern the evolution of non-Maxwellian distribution functions. Its numerical implementation in a discontinuous Galerkin (DG) framework requires the repeated computation of the Rosenbluth potentials H_s and G_s, which satisfy a coupled Poisson system sourced by the distribution function f_s itself. The existing DG-FPO in GKEYLL (MA-DG-FPO) bypasses this Poisson solve by assuming a Maxwellian approximation and evaluating the potentials analytically from the best-fit Maxwellian, which fails precisely in scenarios most relevant in plasma physics: suprathermal tails, loss-cone anisotropic structures, and beam populations that depart significantly from a Maxwellian. The central contribution of this thesis is the development and integration of two complementary schemes that remove this assumption and extend the DG-FPO to handle arbitrary f_s [4].

The first step is to establish a benchmark for what an exact Poisson solve looks like in terms of performance and accuracy. Algebraic multigrid (AMG) is applied to the coupled H-G system, achieving $\mathcal{O}(N \log N)$complexity in the number of degrees of freedom N — a significant advantage over direct solvers in the 3V velocity

space where N grows as NV^3. Moreover, its implementation within the HYPRE preconditioner library made for a highly-parallelizable numerical solver [25, 30]. This reference solver establishes the accuracy floor against which all subsequent schemes are validated, and confirms that the coupled Poisson system is well-posed under exact Dirichlet boundary conditions derived from the Maxwellian potentials. However, since it is not the most efficient for our problem size and its integration with GKEYLL is non-trivial, a natural short-term future direction is the development of a DG-native multigrid solver that exploits the discretization and infrastructure inside GKEYLL without any computational overhead [15].

The numerical solver candidate to be compared with AMG in this thesis is the second-order Richardson iteration. Unlike AMG, it does not make use of any matrix construction. Instead, it recasts the Poisson equation as a damped wave equation and discretizes it with the RDG Laplacian infrastructure already present in GKEYLL. This makes it source-independent, GKEYLL-compatible and trivially parallelizable. The coupled H-G solve via Richardson is validated against AMG and the analytic projection approach of the existing MA-DG-FPO on CPUs and GPUs. Two major caveats with the implementation of the Richardson iteration are a bound on accuracy due to the discretization floor and a stringent restriction on the pseudo-timestep of the scheme that came with the incorporation of the DG Laplacian. This makes Richardson computationally expensive per timestep. The most direct short-term remedy to both issues is the replacement of the RDG Laplacian with a FEM discrete Laplacian, which would allow a less stringent pseudo-timestep $\Delta\tau$ and improve wall-clock time.

Another way around numerical issues such as a discretization floor, is the development of an analytic form of the solution, which leads to the study of multipole expansions of H_s and G_s in the far field velocity space and at the boundaries. By expanding the integral form of the potentials in a Taylor series and expressing the

coefficients in terms of symmetric trace-free (STF) Cartesian moments of the non-Maxwellian deviation $f_1 = f_s - f_M$, the potentials are computed analytically without much computational overhead as the moments are pre-computed in the existing MA-DG-FPO algorithm. The scheme is validated against four non-Maxwellian distribution functions — top-hat, kappa ($\kappa = 2$), loss-cone, and bump-on-tail with relaxation trajectories matching the MA-DG-FPO and moment conservation preserved to machine precision. A short-term extension is to retain higher-order STF moments, improving accuracy and pushing the scheme to be applied closer to the bulk. However, this raises a deeper question: if an analytic representation exists for the far field, can one be constructed for the near field as well? This would eliminate any errors caused by numerical schemes and if constructed using pre-computed quantities, it reduces computational cost significantly. One such popular approach is the use of spherical harmonics. However, transformation between cartesian and polar coordinate systems and trigonometric coefficients, could make the formulation highly non-trivial.

Both multipole and Richardson are integrated into the existing MA-DG-FPO algorithm as a Rosenbluth potential solver piece, to create the X-DG-FPO. Richardson solves the interior Poisson system directly from f_s; the multipole correction is applied at the outermost cell layer for an exact solution at the boundaries. To provide the algorithm with a computational boost, a warm start from the previous timestep solution is applied. The X-DG-FPO algorithm is validated through relaxation tests on all four non-Maxwellian distributions and a thermalization test on a bi-Maxwellian initial condition. The relaxation tests confirm that the X-DG-FPO eliminates the far-tail numerical noise present in the MA-DG-FPO for the kappa distribution, and produces identical relaxation trajectories for the loss-cone and bump-on-tail distributions. However, due to the ineffectiveness of the RDG Laplacian operator on discontinuities, the X-DG-FPO collapses for the top-hat distribution.

This motivates the implementation of an alternative Laplacian operator even more. However, the X-DG-FPO obeys moment conservation for all four distributions, including top-hat. The thermalization test confirms that the X-DG-FPO correctly reproduces the temperature relaxation dynamics for a near-Maxwellian distribution. While GPU implementation is achieved for the X-DG-FPO, its optimization remains a short-term priority.

The contributions of this thesis are steps toward a larger goal: an efficient, fully conservative, continuum-kinetic algorithm for the multi-species nonlinear Rosenbluth/Fokker-Planck equations that handles arbitrary distribution functions without approximation, over disparate velocity and collisional regimes. Several long-term directions follow naturally. Non-uniform velocity meshes would concentrate resolution where f_s is significant in the domain, substantially reducing computation. Without it, we cannot hope to further extend this work to multiple species of particles and achieve generalization. The most non-trivial case of multi-species — cross-species collisions requires the derivation of cross-species Rosenbluth potential equations in the DG framework, careful treatment of the mass ratio dependence in the drag and diffusion tensors, and conservation corrections that enforce momentum and energy exchange between species consistently. The LBO currently serves as the ion-electron collision operator in GKEYLL; the X-DG-FPO provides a path to replacing it with a fully nonlinear operator.

Achievement of all these computational goals allows for tractable continuum-kinetic simulation capability in previously inaccessible collisional scenarios with complex plasma dynamics. Some potential plasma regimes of application include nonlocal electron transport in laser plasmas, sheath formation, ionospheric instabilities, and various fusion configurations [15].

Bibliography

[1] Francis F. Chen. *Introduction to Plasma Physics and Controlled Fusion*. 3rd ed. Springer, 2015.

[2] Richard Fitzpatrick. *Plasma Physics: An Introduction*. Boca Raton, FL: CRC Press, 2014. ISBN: 978-1-4665-9426-5.

[3] James Juno et al. "Discontinuous Galerkin algorithms for fully kinetic plasmas". In: *Journal of Computational Physics* 353 (2018), pp. 110–147. DOI: 10.1016/j.jcp.2017.10.009.

[4] Ammar Hakim et al. *Gkeyll simulation framework*. 2024. URL: https://gkeyll.readthedocs.io.

[5] A. J. Brizard and T. S. Hahm. "Foundations of nonlinear gyrokinetic theory". In: *Reviews of Modern Physics* 79.2 (2007), pp. 421–468. DOI: 10.1103/RevModPhys.79.421.

[6] Yu. L. Klimontovich. *The Statistical Theory of Non-Equilibrium Processes in a Plasma*. Cambridge, Massachusetts: MIT Press, 1967.

[7] Robert Bird et al. "VPIC 2.0: Next Generation Particle-in-Cell Simulations". In: *IEEE Transactions on Parallel and Distributed Systems* 33.4 (2022), pp. 952–963. DOI: 10.1109/TPDS.2021.3084795.

[8] D. Pokhotelov, S. von Alfthan, Y. Kempf, et al. “Ion distributions upstream and downstream of the Earth’s bow shock: First results from Vlasiator”. In: *Annales Geophysicae* 31.12 (2013), pp. 2207–2212. DOI: 10.5194/angeo-31-2207-2013.

[9] P. L. Bhatnagar, E. P. Gross, and M. Krook. “A model for collision processes in gases”. In: *Physical Review* 94 (1954), pp. 511–525. DOI: 10.1103/PhysRev.94.511.

[10] Marshall N. Rosenbluth, William M. MacDonald, and David L. Judd. “Fokker-Planck equation for an inverse-square force”. In: *Physical Review* 107 (1957), pp. 1–6. DOI: 10.1103/PhysRev.107.1.

[11] A. Lenard and Ira B. Bernstein. “Plasma oscillations with diffusion in velocity space”. In: *Physical Review* 112 (1958), pp. 1456–1459. DOI: 10.1103/PhysRev.112.1456.

[12] J. P. Dougherty. “Model Fokker-Planck equation for a plasma and its solution”. In: *Physics of Fluids* 7 (1964), pp. 1788–1799. DOI: 10.1063/1.2746779.

[13] L. Landau. “The transport equations in the case of Coulomb interactions”. In: *Phys. Z. Sowjet* 10 (1936), p. 154.

[14] L. D. Landau. *The kinetic equation in the case of Coulomb interaction*. Tech. rep. San Diego, CA: General Dynamics/Astronautics, 1958.

[15] John M. Rodman. “Discontinuous Galerkin Studies of Collisional Dynamics in Continuum-Kinetic Plasma”. Available at https://hdl.handle.net/10919/124380. PhD thesis. Blacksburg, Virginia: Virginia Polytechnic Institute and State University, Nov. 2024.

[16] Ammar Hakim. *The Fokker-Planck collision operator in Gkeyll*. https://ammar-hakim.org/sj/_static/files/fpo-notes.pdf. Accessed: 2025.

[17] Ludwig Boltzmann. "Further Studies on the Thermal Equilibrium of Gas Molecules". In: DOI: 10.1142/9781848161337_0015.

[18] Sydney Chapman and T. G. Cowling. *The Mathematical Theory of Non-uniform Gases*. Cambridge University Press. ISBN: 052140844X.

[19] S. Sarkar, S. Paul, and R. Denra. "Bump-on-tail instability in space plasmas". In: *Physics of Plasmas* 22.10 (2015), p. 102109. DOI: 10.1063/1.4933041.

[20] M. Lazar et al. "Dual Maxwellian-Kappa modeling of the solar wind electrons: new clues on the temperature of Kappa populations". In: *Astronomy & Astrophysics* 602 (2017), A44. DOI: 10.1051/0004-6361/201630194.

[21] Bram van Leer and S. Nomura. "Discontinuous Galerkin for Diffusion". In: *17th AIAA Computational Fluid Dynamics Conference*. 2005, pp. 2005–5108. DOI: 10.2514/6.2005-5108.

[22] H. T. Huynh. *A Flux Reconstruction Approach to High-Order Schemes Including Discontinuous Galerkin Methods*. Tech. rep. AIAA 2007-4079. 2007, pp. 1–42.

[23] Sophie Boileau et al. *Parallel Algebraic Multigrid for Fusion and Higher-Order PDEs*. Tech. rep. LLNL-SR-857062. Research in Industrial Projects for Students (RIPS), IPAM. https://doi.org/10.2172/2205732. Lawrence Livermore National Laboratory, Aug. 2023.

[24] Viktor T. Toth and Slava G. Turyshev. "Efficient trace-free decomposition of symmetric tensors of arbitrary rank". In: *International Journal of Modern Physics A* 37.20n21 (2022), p. 2250201. DOI: 10.1142/S0219887822502012.

[25] Robert D. Falgout. "An Introduction to Algebraic Multigrid". In: *Computing in Science & Engineering* 8.6 (2006), pp. 24–33. DOI: 10.1109/MCSE.2006.105.

[26] Allison H. Baker et al. “Scaling hypre’s Multigrid Solvers to 100,000 Cores”. In: *High-Performance Scientific Computing: Algorithms and Applications*. Ed. by Michael W. Berry et al. LLNL-JRNL-479591. London: Springer, 2012, pp. 261–279. DOI: 10.1007/978-1-4471-2437-5_13.

[27] Irad Yavneh. “Why Multigrid Methods Are So Efficient”. In: *Computing in Science & Engineering* 8.6 (2006), pp. 12–22. DOI: 10.1109/MCSE.2006.125.

[28] Lars Grasedyck, Lu Wang, and Jinchao Xu. “A Nearly Optimal Multigrid Method for General Unstructured Grids”. In: *Numerische Mathematik* 134.4 (2016), pp. 637–666. DOI: 10.1007/s00211-015-0785-7.

[29] John W. Ruge and Klaus Stüben. “Algebraic Multigrid (AMG)”. In: *Multigrid Methods*. Ed. by Stephen F. McCormick. Vol. 3. Frontiers in Applied Mathematics. Philadelphia, PA: SIAM, 1987. Chap. 4, pp. 73–130. DOI: 10.1137/1.9781611971057.ch4.

[30] HYPRE Development Team. *HYPRE High Performance Preconditioners: User’s Manual*. 2023. URL: https://hypre.readthedocs.io/en/latest/.

[31] Ammar Hakim. *JE35: Benchmarking an iterative discontinuous Galerkin Poisson solver*. Gkeyll Simulation Journal, https://ammar-hakim.org/sj/je/je35/je35-iter-poisson.html. 2024.

[32] G. H. Golub and R. S. Varga. “Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods”. In: *Numerische Mathematik* 3 (1961), pp. 147–168.

[33] S. Jardin. *Computational Methods in Plasma Physics*. Chapman & Hall/CRC Computational Science Series, 2010.

[34] Slava G. Turyshev and Viktor T. Toth. “Multipole decomposition of gravitational lensing”. In: *Physical Review D* 105 (2022), p. 024022. DOI: 10.1103/PhysRevD.105.024022.

[35] W.T. Taitano, L. Chacón, and A.N. Simakov. "An adaptive, conservative 0D-2V multispecies Rosenbluth–Fokker–Planck solver for arbitrarily disparate mass and temperature regimes". In: *J. Comput. Phys.* 318 (2016), pp. 391–420.

[36] Leo C. Stein. *Notes: Magnetostatic multipole expansion using STF tensors*. https://duetosymmetry.com/notes/magnetostatics-stf-mpoles/. Accessed: [insert date]. 2023.

[37] Ammar Hakim et al. "Conservative discontinuous Galerkin schemes for nonlinear Dougherty–Fokker–Planck collision operators". In: *Journal of Plasma Physics* 86.4 (2020), p. 905860403. DOI: 10.1017/S0022377820000586.

[38] E. S. Yoon and C. S. Chang. "A Fokker–Planck–Landau collision equation solver on two-dimensional velocity grid and its application to particle-in-cell simulation". In: *Physics of Plasmas* 21.3 (2014), p. 032503. DOI: 10.1063/1.4867359.

Appendix A

Rosenbluth Potentials and Corresponding Drag and Diffusion Coefficients for a Maxwellian Distribution

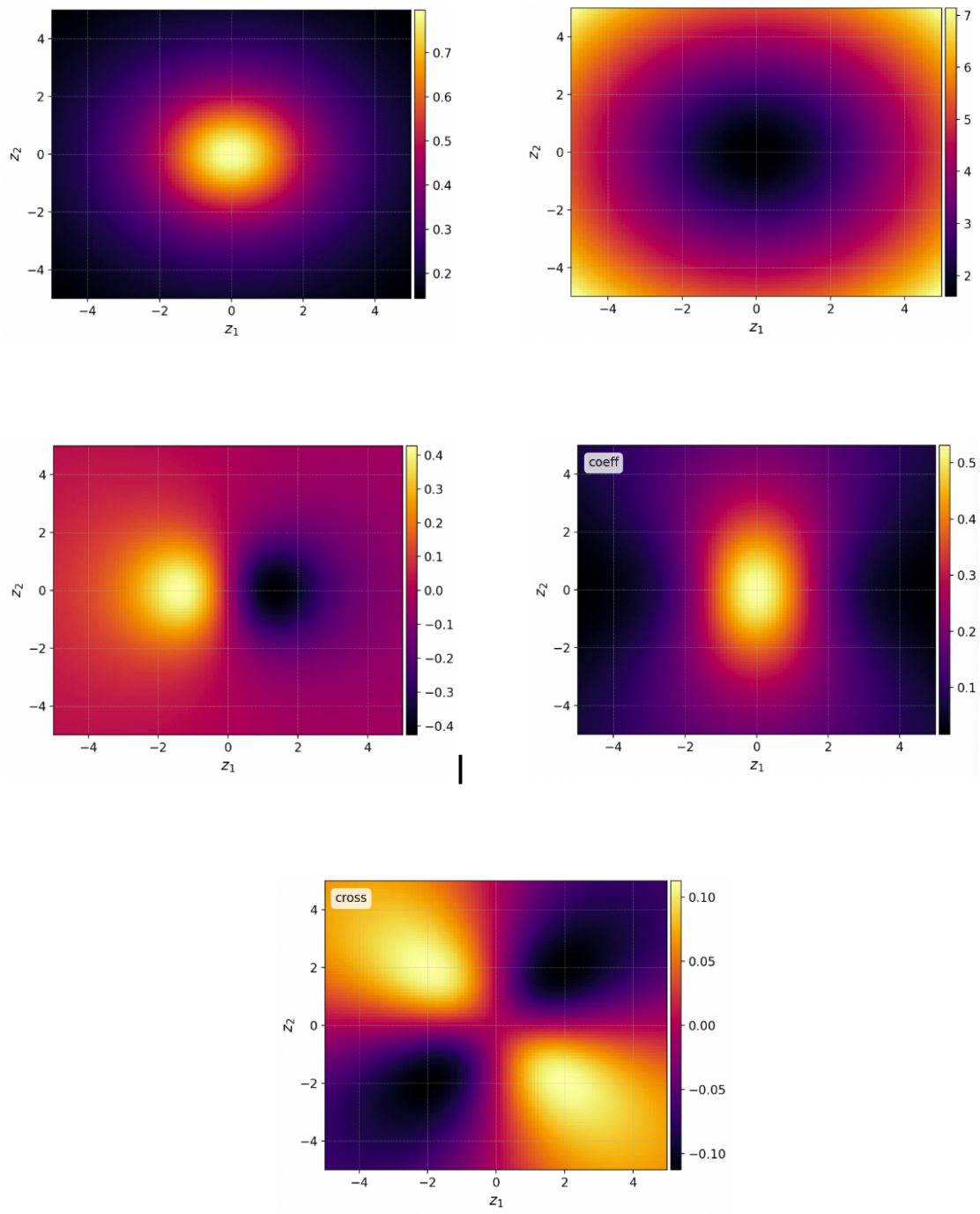

Figure A.1: Rosenbluth potentials H_M (top left) and G_M (top right), drag coefficient a_x (center left), diffusion tensor element D_{xx} (center right), and D_{xy} (bottom). All quantities evaluated at $v_z = 0$ for a Maxwellian f_M, $p = 2$ Serendipity basis.

Appendix B

Distribution Function Profiles

The five distribution functions studied in this thesis are shown below. Parameters for all of them are defined in Chapter 1. All plots show the one-dimensional slice $f_s(v_x, 0, 0)$ as a function of v_x/v_{th}. The bump-on-tail and kappa distributions are shown on a logarithmic scale to reveal important tail structure and information; the remaining distributions are shown on a linear scale.

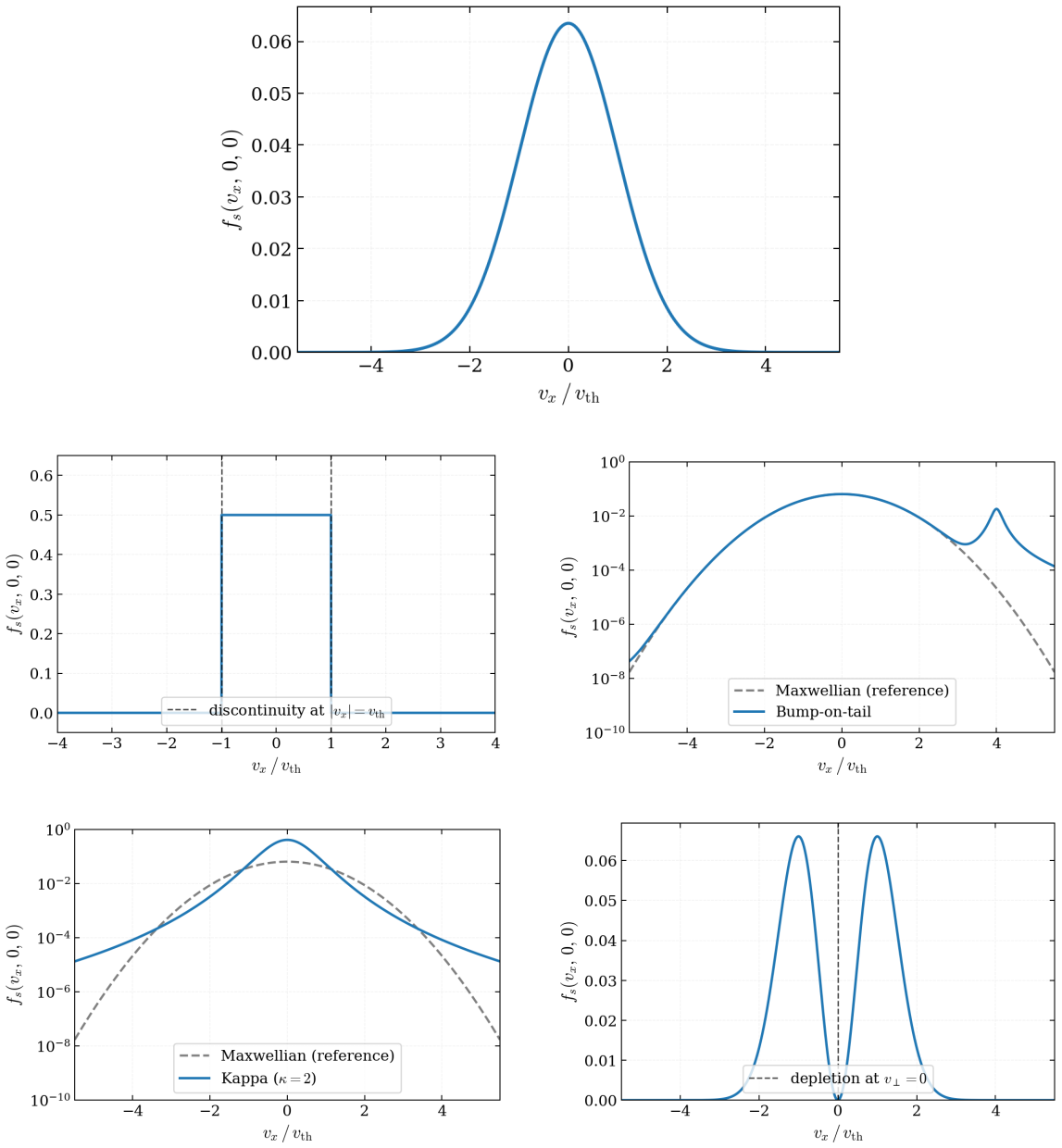

Figure B.1: (A) Maxwellian distribution. (B) Top-hat distribution. (C) Bump-on-tail distribution. (D) Kappa distribution with $\kappa = 2$. (E) Loss-cone distribution.

Appendix C

HYPRE BoomerAMG Parameter Study

Coarsen Type	Time (s)	Rel. Residual	Iterations	Time/Iter (s)
CLJP	4.828	7.84×10^{-7}	14	0.345
RS-NBT	1.914	4.33×10^{-7}	10	0.191
RS-3	1.900	4.33×10^{-7}	10	0.190
Falgout	2.051	3.84×10^{-7}	10	0.205
CLJP-FV	4.776	7.84×10^{-7}	14	0.341
PMIS	**1.507**	$\mathbf{4.41 \times 10^{-7}}$	**13**	**0.116**
PMIS-FV	1.516	4.41×10^{-7}	13	0.117
HMIS	1.707	3.33×10^{-7}	11	0.155
RS-1	1.750	3.33×10^{-7}	11	0.159
CGC	1.913	4.34×10^{-7}	10	0.191
CGC-E	1.959	4.34×10^{-7}	10	0.196

Table C.1: Coarsen type sweep. Bold row indicates the selected configuration.

Interpolation Type	Time (s)	Rel. Residual	Iterations	Time/Iter (s)
CM	1.941	1.63×10^{-3}	20	0.097
Dir-WS	1.953	2.29×10^{-3}	20	0.098
Mult	1.900	8.51×10^{-5}	20	0.095
Mult-WS	1.969	5.49×10^{-4}	20	0.098
Ext-I	**1.559**	$\mathbf{4.41 \times 10^{-7}}$	**13**	**0.120**
Ext-I-NCC	1.508	5.29×10^{-7}	13	0.116
Std	1.790	8.05×10^{-7}	16	0.112
Std-WS	1.651	6.04×10^{-7}	15	0.110
CB-D	4.704	1.32×10^{-2}	20	0.235
FF	1.734	5.70×10^{-7}	14	0.124
FF1	1.853	6.98×10^{-7}	16	0.116
Ext	1.604	6.13×10^{-7}	14	0.115
AW	1.999	2.29×10^{-3}	20	0.100
Ext-MM	1.646	9.96×10^{-7}	14	0.118
Ext-I-MM	1.497	9.32×10^{-7}	12	0.125
Ext-E-MM	1.502	9.50×10^{-7}	12	0.125

Table C.2: Interpolation type sweep. Bold row indicates the selected configuration.

Relaxation Type	Time (s)	Rel. Residual	Iterations	Time/Iter (s)
J	1.521	3.63×10^{-3}	20	0.076
GS-S	1.213	5.04×10^{-7}	16	0.076
GS-PI	1.286	5.04×10^{-7}	16	0.080
SOR-F	1.247	5.04×10^{-7}	16	0.078
SOR-B	1.443	4.97×10^{-7}	16	0.090
GS-C	1.781	5.04×10^{-7}	16	0.111
SSOR	**1.539**	$\mathbf{9.50 \times 10^{-7}}$	**12**	**0.128**
J-Mat	1.328	3.63×10^{-3}	20	0.066
GS-SHS	1.552	9.50×10^{-7}	12	0.129
2S	2.923	8.32×10^{-7}	16	0.183
2S-S	4.200	8.18×10^{-7}	16	0.263
GS-F	1.300	5.04×10^{-7}	16	0.081
GS-B	1.382	4.97×10^{-7}	16	0.086
CG	2.598	3.55×10^{-4}	20	0.130
C	1.964	4.81×10^{-7}	15	0.131
J-FCF	2.862	7.12×10^{-1}	20	0.143
J-S	1.319	1.73×10^{-5}	20	0.066
K	3.333	1.07×10^{-6}	20	0.167
GS-SCL1	1.619	9.50×10^{-7}	12	0.135
GS-L1H	1.573	9.50×10^{-7}	12	0.131
Mat	1.370	9.79×10^{-7}	16	0.086

Table C.3: Relaxation type sweep. Bold row indicates the selected configuration.

CFL Fraction	Time (s)	Rel. Residual	Iterations	Time/Iter (s)
0.0	0.866	9.85×10^{-7}	13	0.067
0.1	0.905	4.11×10^{-7}	13	0.070
0.2	0.922	4.00×10^{-7}	13	0.071
0.3	0.962	4.72×10^{-7}	13	0.074
0.4	1.065	4.85×10^{-7}	14	0.076
0.5	**1.367**	$\mathbf{5.04 \times 10^{-7}}$	**16**	**0.085**
0.6	1.604	9.05×10^{-7}	18	0.089
0.7	1.807	3.02×10^{-6}	20	0.090
0.8	1.834	2.87×10^{-5}	20	0.092
0.9	1.694	2.57×10^{-4}	20	0.085
1.0	0.330	5.41×10^{-2}	20	0.016

Table C.4: CFL fraction (strength threshold) sweep. Bold row indicates the selected configuration.

Appendix D

Multipole Expansion of G_S

The multipole expansion of G_s follows the same STF framework as H_s in Section 5.1, applied to the kernel $|\mathrm{v} - \mathrm{v}'|$ in Eq. (5.2). Writing out the first four terms explicitly:

$$G_s(\mathrm{v}) = \underbrace{n_s\, v}_{\ell=0} - \underbrace{\frac{\mathcal{T}_s^{\langle a\rangle} v^a}{v}}_{\ell=1} + \underbrace{\frac{\operatorname{tr} M_s^{(2)}}{2v} - \frac{M_{s,ab}^{(2)}\, v^a v^b}{2v^3}}_{\ell=2} + \underbrace{\frac{T_i v^i}{2v^3} - \frac{C}{2v^5}}_{\ell=3} + \mathcal{O}(v^{-1}), \tag{D.1}$$

where the scalar contractions Q, C, T_i, and M_{1v} are defined as in the H_s expansion,

$$M_{1v} = M_{s,i}^{(1)}\, v^i, \tag{D.2}$$

$$Q = M_{s,ij}^{(2)}\, v^i v^j, \tag{D.3}$$

$$\operatorname{tr} M_s^{(2)} = M_{s,xx}^{(2)} + M_{s,yy}^{(2)} + M_{s,zz}^{(2)}, \tag{D.4}$$

$$T_i = M_{s,ijj}^{(3)} \quad \text{(trace over last two indices)}, \tag{D.5}$$

$$C = M_{s,ijk}^{(3)}\, v^i v^j v^k, \tag{D.6}$$

and the STF moments $\mathcal{T}_s^{\langle a\rangle}$, $M_{s,ij}^{(2)}$, $M_{s,ijk}^{(3)}$ are as defined in Eqs. (5.10)–(5.12). The key structural difference from H_s is the $v^{\ell-1}$ dependence rather than $v^{-(\ell+1)}$: the monopole term grows as v in the far field rather than decaying, reflecting the slower decay of the G_s kernel.

Appendix E

Multipole Expansions of the Drag and Diffusion Coefficients

The drag coefficient $\mathbf{a}_s$ and diffusion tensor $\mathbf{D}_s$ are obtained from H_s and G_s via Eq. (5.16). Below we list the explicit multipole expressions for all three components of $\mathbf{a}_s$ and all six independent components of $\mathbf{D}_s$, ordered by multipole contribution $\ell = 0, 1, 2, 3$. The scalar contractions M_{1v}, Q, C, T_i, $\partial_x Q$, $\partial_x C$ are as defined in Appendix D, and $\Gamma \equiv \Gamma_{ss'}$.

Drag Coefficient $a_{s,x}$

$$
\begin{aligned}
a_{s,x}^{(\ell=0)} &= -2\Gamma \frac{n_s\, v_x}{v^3}, \\
a_{s,x}^{(\ell=1)} &= 2\Gamma \left(\frac{M_{s,x}^{(1)}}{v^3} - \frac{3 M_{1v}\, v_x}{v^5} \right), \\
a_{s,x}^{(\ell=2)} &= 2\Gamma \left(\frac{3}{2} \frac{\partial_x Q}{v^5} - \frac{15 Q\, v_x}{2v^7} + \frac{3\,\mathrm{tr}\, M_s^{(2)}\, v_x}{2v^5} \right), \\
a_{s,x}^{(\ell=3)} &= 2\Gamma \left(\frac{5\,\partial_x C}{2v^7} - \frac{35 C\, v_x}{2v^9} - \frac{3 T_x}{2v^5} + \frac{15 T_i v^i\, v_x}{2v^7} \right),
\end{aligned}
\tag{E.1}
$$

where $\partial_x Q = 2(M^{(2)}_{s,xx} v_x + M^{(2)}_{s,xy} v_y + M^{(2)}_{s,xz} v_z)$ and $\partial_x C = 3M^{(3)}_{s,xxx} v_x^2 + 6M^{(3)}_{s,xxy} v_x v_y + 6M^{(3)}_{s,xxz} v_x v_z + 3M^{(3)}_{s,xyy} v_y^2 + 6M^{(3)}_{s,xyz} v_y v_z + 3M^{(3)}_{s,xzz} v_z^2$. The expressions for $a_{s,y}$ and $a_{s,z}$ follow by cyclic permutation of (x, y, z).

Diffusion Tensor $D_{s,xx}$

$$
\begin{aligned}
D^{(\ell=0)}_{s,xx} &= \Gamma \frac{n_s (v_y^2 + v_z^2)}{v^3}, \\
D^{(\ell=1)}_{s,xx} &= \Gamma \left(\frac{2M^{(1)}_{s,x} v_x}{v^3} + \frac{M_{1v}}{v^3} - \frac{3M_{1v}\, v_x^2}{v^5} \right), \\
D^{(\ell=2)}_{s,xx} &= \Gamma \Big(-\frac{\operatorname{tr} M^{(2)}_s}{2v^3} + \frac{3 \operatorname{tr} M^{(2)}_s\, v_x^2}{2v^5} - \frac{M^{(2)}_{s,xx}}{v^3} \\
&\qquad + \frac{3\, \partial_x Q\, v_x}{v^5} + \frac{3Q}{2v^5} - \frac{15 Q\, v_x^2}{2v^7} \Big), \\
D^{(\ell=3)}_{s,xx} &= \Gamma \Big(-\frac{3T_x\, v_x}{2v^5} - \frac{3T_i v^i}{2v^5} + \frac{15 T_i v^i\, v_x^2}{2v^7} \\
&\qquad - \frac{\partial_{xx} C}{2v^5} + \frac{5\, \partial_x C\, v_x}{v^7} + \frac{5C}{2v^7} - \frac{35 C\, v_x^2}{2v^9} \Big),
\end{aligned}
\tag{E.2}
$$

where $\partial_{xx} C = 6(M^{(3)}_{s,xxx} v_x + M^{(3)}_{s,xxy} v_y + M^{(3)}_{s,xxz} v_z)$.

Diffusion Tensor $D_{s,xy}$

$$
\begin{aligned}
D_{s,xy}^{(\ell=0)} &= -\Gamma\,\frac{n_s\,v_x v_y}{v^3},\\
D_{s,xy}^{(\ell=1)} &= \Gamma\left(\frac{M_{s,x}^{(1)}v_y + M_{s,y}^{(1)}v_x}{v^3} - \frac{3M_{1v}\,v_x v_y}{v^5}\right),\\
D_{s,xy}^{(\ell=2)} &= \Gamma\left(\frac{3\,\mathrm{tr}\,M_s^{(2)}\,v_x v_y}{2v^5} - \frac{M_{s,xy}^{(2)}}{v^3} + \frac{3(\partial_x Q\,v_y + \partial_y Q\,v_x)}{2v^5} - \frac{15Q\,v_x v_y}{2v^7}\right),\\
D_{s,xy}^{(\ell=3)} &= \Gamma\Big(-\frac{3(T_x v_y + T_y v_x)}{2v^5} + \frac{15T_i v^i\,v_x v_y}{2v^7} - \frac{\partial_{xy}C}{2v^5}\\
&\qquad + \frac{5(\partial_x C\,v_y + \partial_y C\,v_x)}{2v^7} - \frac{35C\,v_x v_y}{2v^9}\Big),
\end{aligned}
\tag{E.3}
$$

where $\partial_y Q = 2(M_{s,xy}^{(2)}v_x + M_{s,yy}^{(2)}v_y + M_{s,yz}^{(2)}v_z)$, $\partial_{xy}C = 6(M_{s,xxy}^{(3)}v_x + M_{s,xyy}^{(3)}v_y + M_{s,xyz}^{(3)}v_z)$, and $\partial_y C$ follows by cyclic substitution. The remaining components $D_{s,xz}$, $D_{s,yy}$, $D_{s,yz}$, $D_{s,zz}$ follow by cyclic permutation of (x,y,z) applied to the $D_{s,xx}$ and $D_{s,xy}$ expressions above.

Appendix F

Code Samples

The code used to implement the iterative and analytic Poisson solving schemes developed in this thesis, along with the runtime files used to produce the results and the plotting scripts used to generate all figures, is provided in the following GitHub repository:

https://github.com/atmikdas-plasma/MS-Thesis/tree/main