DISCONTINOUS GALERKIN METHODS FOR VLASOV MODELS OF PLASMA

By

David C. Seal

A dissertation submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

(Mathematics)

at the

UNIVERSITY OF WISCONSIN – MADISON

2012

Date of final oral examination: May 30, 2012

The dissertation is approved by the following members of the Final Oral Committee:

James A. Rossmanith, Assistant Professor, Mathematics

Jean-Luc Thiffeault, Associate Professor, Mathematics

Carl R. Sovinec, Professor, Engineering Physics

Eftychios Sifakis, Assistant Professor, Computer Science

Fabian Waleffe, Professor, Mathematics

Abstract

The Vlasov-Poisson equations describe the evolution of a collisionless plasma, represented through a probability density function (PDF) that self-interacts via an electrostatic force. One of the main difficulties in numerically solving this system is the severe time-step restriction that arises from parts of the PDF associated with moderate-to-large velocities. The dominant approach in the plasma physics community for removing these time-step restrictions is the so-called particle-in-cell (PIC) method, which discretizes the distribution function into a set of macro-particles, while the electric field is represented on a mesh. Several alternatives to this approach exist, including fully Lagrangian, fully Eulerian, and so-called semi-Lagrangian methods. The focus of this work is the semi-Lagrangian approach, which begins with a grid-based Eulerian representation of both the PDF and the electric field, then evolves the PDF via Lagrangian dynamics, and finally projects this evolved field back onto the original Eulerian mesh.

We present a semi-Lagrangian and a hybrid semi-Lagrangian method for solving the Vlasov Poisson equations, based on high-order discontinuous Galerkin (DG) spatial representations of the solution. The Poisson equation is solved via a high-order local discontinuous Galerkin (LDG) scheme. The resulting methods are high-order accurate, which is demonstrably important for this problem in order to retain the rich phase-space structure of the solution; mass conservative; and provably positivity-preserving. We argue that our approach is a promising method that can produce very accurate results at relatively low computational expense. We demonstrate this through several examples for the (1+1)D case, using both the hybrid as well as the full semi-Lagrangian method.

In particular, the methods are validated on several numerical test cases, including the two-stream instability problem, Landau damping, and the formation of a plasma sheath. In addition, we propose a (2+2)D method that promises to be a productive avenue of future research. The (2+2)D method incorporates local time-stepping methods on unstructured grids in physical space and semi-Lagrangian time stepping on Cartesian grids in velocity space. This method is again high-order, mass conservative, and provably positivity-preserving.

Acknowledgements

Professional Acknowledgements

I would first and foremost like to thank my adviser, James Rossmanith, for being my guide throughout this difficult journey. James has been patient with me, and has always been there to discuss matters with me, research or otherwise. He gave me plenty of freedom to investigate problems on my own and at my own pace, while at the same time having enough foresight to steer me towards problems that could produce results given the limited time available during graduate school.

Secondly, I would like to thank my committee members, Jean-Luc Thiffeault, Carl Sovinec, Eftychios Sifakis, and Fabian Waleffe, for their valuable feedback and comments. I would also like to thank Jeff Hittinger, Jeff Banks, and Milo Dorr of Lawrence Livermore National Laboratory for welcoming me into their research groups for two consecutive summers.

Finally, I would like to thank the State of Wisconsin and the U.S. Federal Government. Without the aid of public funds, this dissertation would have never happened.

Personal Acknowledgements

There is zero doubt that the most important person in my life, my wife, Gwendolyn Miller Seal, deserves everything I can thank her for, and then some. I can honestly say that if it weren't for her, this dissertation would have never happened.

To that end, my parents, Carolee and Boyd deserve a thank you for the enduring sacrifices they made for me. This includes, but is not limited to, a large amount of grey

hair developed during my teen-age and college years, as well as providing a sanctuary that I could always call home.

List of Figures

1 Plasma Pressure-Temperature Diagram 3

2 Solar Corona Picture 4

3 Tokamak Design and Physical Picture 4

4 HLLE Approximate Riemann Solver 26

5 Illustration of 1D Shift+Project Method 39

6 Illustration of Forward-Backward Evolution of Semi-Lagrangian Scheme . 41

7 Illustration of Shift + Project Method for Quasi-1D Problem 46

8 The Two-Stream Instability Problem: Grid Refinement and Comparison of Time Splitting Methods 78

9 The Two-Stream Instability Problem: Grid Refinement, Comparison of Time Splitting Methods; Vertical Cross-Sections. 79

10 The Two-Stream Instability Problem: Positivity-Preserving Limiter. . . . 80

11 The Two-Stream Instability Problem: Fine Grid Results 81

12 The Two-Stream Instability Problem: Conserved Quantities. 82

13 The Weak Landau Damping Problem: Decay of Electric Field. 83

14 The Weak Landau Damping Problem: Conserved Quantities. 84

15 The Strong Landau Damping Problem: Phase-Space Plots. 86

16 The Strong Landau Damping Problem: Decay of Electric Field. 87

17 The Strong Landau Damping Problem: Conserved Quantities. 88

18 Landau Damping Problem: Vertical Slices for Large Velocities. 89

19 Plasma Sheath Problem: Phase-Space Plot. 92

20 Plasma Sheath Problem: Electric Field and Potential. 92

21 Landau Damping: HSLDG Results 106

22 Weak Landau Damping: HSLDG Results 107

23 Bump on Tail: HSLDG Results 111

24 Two-Stream Instability: HSLDG Grid Refinement 112

25 Illustration of 2D Evolution for Unstructured Grids 116

26 Unstructured Grid: Solid Body Rotation 120

List of Tables

1 High-Order Runge-Kutta-Nyström Operator Split Coefficients 51

2 Relative L_2-Norm Errors for Computing $f'(x)$ and $f''(x)$ Using the Legendre Coefficient Finite Difference Formulas 66

3 SLDG Convergence Study for the Linear Advection Equation 73

4 SLDG Convergence Study for the Forced Vlasov-Poisson Equation: Comparison of Two Operator Split Methods 76

5 HSLDG Convergence Study for the Linear Advection Equation. 103

6 HSLDG Convergence Study for the Forced Vlasov-Poisson Equation . . . 108

7 Comparison of Two High-Order Split Methods 109

8 Convergence Study on Unstructured Grids 118

9 Quadrature Weights and Points 130

Contents

Abstract **i**

Acknowledgements **iii**

1 Introduction **1**

1.1 Kinetic and Fluid Models in Plasma Physics 1

1.2 Vlasov Models of Plasma 4

1.2.1 Vlasov-Maxwell 5

1.2.2 Vlasov-Poisson 6

1.2.3 Non-Dimensionalization of Vlasov-Poisson 7

1.2.4 Moments and Conserved Quantities 9

1.2.5 Known Theoretical Results for Vlasov-Poisson 13

1.3 Numerical Methods for Vlasov-Poisson 14

1.3.1 Numerical Challenges 14

1.3.2 Numerical Methods 15

1.4 Scope of This Work 18

2 Discontinuous Galerkin Methods **20**

2.1 1D Hyperbolic Conservation Laws 22

2.1.1 HLLE approximate Riemann solver 25

2.1.2 High-Order Time Stepping 28

2.2 2D Problems on Cartesian Grids 29

2.3 2D Problems on Unstructured Grids 32

3 Semi-Lagrangian Methods for 1+1 Vlasov-Poisson 34

3.1 Introduction 35

3.2 Cheng and Knorr Splitting 35

3.3 SLDG Schemes for the Advection Equation 37

3.3.1 1D Constant Coefficient Problem 38

3.3.2 Quasi 1D Advection Equation 43

3.3.3 Operator Split Methods 48

3.3.4 Positivity-preserving limiter 52

3.4 Semi-Lagrangian Vlasov Poisson 57

3.4.1 1D Poisson solver 58

3.4.2 High-order Electric Field 62

3.5 Numerical examples 70

3.5.1 Linear advection 70

3.5.2 A forced problem: verifying order of accuracy 72

3.5.3 Two-stream instability 75

3.5.4 Weak Landau damping 77

3.5.5 Strong Landau damping 83

3.5.6 Plasma Sheath 87

4 Hybrid Semi-Lagrangian Methods for 1+1 Vlasov-Poisson 93

4.1 Extensions of SLDG to Higher Dimensions 93

4.2 Hybrid semi-Lagrangian Scheme 94

4.2.1 Description of Hybrid Method 95

4.2.2 Sub-Cycling for the Hybrid Scheme 99
4.2.3 Summary of the Hybrid (HSLDG) Method 101
4.3 Numerical Examples 102
4.3.1 Linear advection 102
4.3.2 A forced problem: verifying order of accuracy 104
4.3.3 Landau Damping 105
4.3.4 Bump-on-Tail 108
4.3.5 Two Stream Instability 110

5 Towards a Hybrid Semi-Lagrangian Method for 2+2 Vlasov-Poisson 113
5.1 Basic Scheme 114
5.2 Numerical Examples 118
5.2.1 Periodic Advection 118
5.2.2 Solid Body Rotation 118

6 Conclusions and Future Work 121
6.1 Conclusions 121
6.1.1 Semi-Lagrangian Discontinuous Galerkin 121
6.1.2 Hybrid Semi-Lagrangian Discontinuous Galerkin 122
6.1.3 Hybrid SLDG Methods for (2+2)D Unstructured Grids 123
6.2 Future Work 123

A Numerical evaluation of conserved quantities 126
A.1 Relative L_2-norm error in 1D 127
A.2 Relative L_2-norm error in 2D 128

B Numerical Integration **129**

Bibliography **132**

Chapter 1

Introduction

The purpose of this chapter is to provide context for the focus of this thesis, which is novel work on semi-Lagrangian and hybrid discontinuous Galerkin (DG) methods for the Vlasov-Poisson system. We begin with a brief description of plasma physics, and then provide an introduction to the Vlasov equations. We then briefly review various numerical methods that have been used for solving the Vlasov-Poisson equations.

Subsequent chapters deal with the specific numerical methods used in this work. In particular, in Chapter 2 we describe classical discontinuous Galerkin (DG) methods for solving hyperbolic problems. Chapters 3, 4, and 5 are the focus of this thesis, and describe in detail our novel work on positivity-preserving, high-order, semi-Lagrangian, discontinuous Galerkin methods for the Vlasov-Poisson equations.

1.1 Kinetic and Fluid Models in Plasma Physics

Plasma is the state of matter in which electrons are completely disassociated from their ions. This state, often referred to as the fourth state of matter, is the most abundant form of ordinary matter in the universe, and is typical of a state at high pressure, high temperature or a combination of both. (c.f. Figure 1). Given its abundance, scientific and engineering examples of plasma are numerous. Understanding the behavior of plasma is

central to studying astrophysical applications involving solar wind, solar corona, as well as laboratory examples, including magnetic confinement for nuclear fission and fusion prospects in laboratories and reactors (c.f. Figures 2 – 3).

Mathematical models of plasma can be roughly classified into two categories: fluid models and kinetic models. The equations involved in a fluid description are generally more complicated, but lower dimensional (i.e., 3-dimensional space), while the equations involved in a kinetic description are simpler to write down, but much more computationally expensive because they describe a high-dimensional system (i.e., 6-dimensional phase space).

Kinetic models of plasma are useful when a a plasma deviates significantly from thermodynamic equilibrium, because they provide a complete phase-space description. This can be the case, for example, when relatively few collisions occur. In the case of the Vlasov-Poisson system, we will see that kinetic simulations, and especially high-order kinetic simulations, reveal an incredibly rich solution structure for the phase-space distribution function.

Fluid descriptions of plasma reduce the dimensionality of the kinetic description by solving for moments of the distribution function (i.e., integrals of the distribution function against various powers of the velocity variables). One difficulty with this approach is that the moment hierarchy never closes: the evolution of the k^{th} moment involves the $(k+1)^{\text{st}}$ moment. In order to close the moment hierarchy we need to make some assumptions about the higher moments of the distribution function. Therefore, the fluid approach replaces the extra velocity variables with more equations involving physical observables, but at the expense of losing a full phase-space description. In particular,

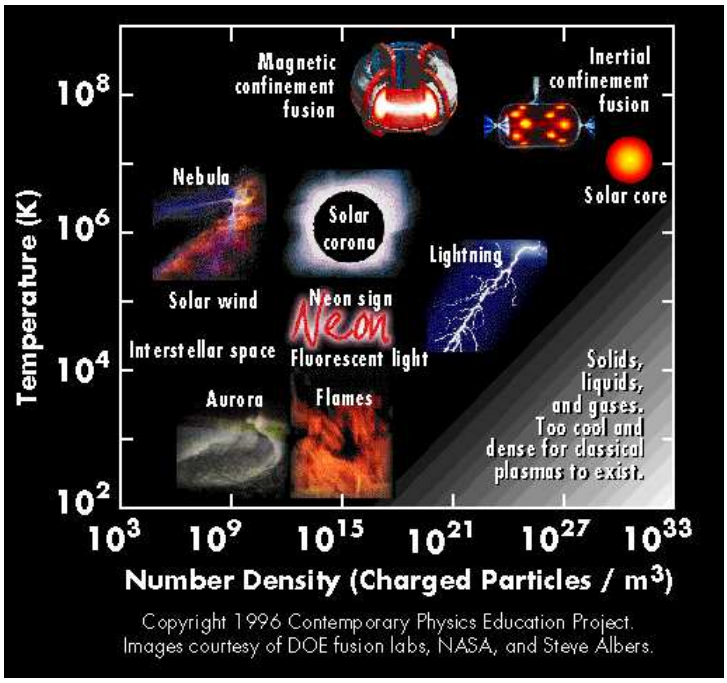


Figure 1: Plasma pressure-temperature diagram. Plasma is a state of matter that exists at high temperature, large pressure, or a combination of the two.

one usually assumes that the velocity distribution does not deviate too much from thermodynamic equilibrium. In certain scenarios, such as in the case of ample collisions, the fluid assumption is valid.

In summary, the trade-off between kinetic and fluid models of plasma are that: (1) kinetic models are valid over a broad range of physical phenomena and make no assumptions about being in or near thermodynamic equilibrium, but are computationally expensive due their high-dimensionality; (2) fluid models are generally valid only near thermodynamic equilibrium and require some closure assumptions, but are computationally less expensive due to their lower dimensionality. In this work we focus on pure kinetic models and attempt to obtain efficient numerical schemes by using high-order spatial and temporal discretizations and appealing to semi-Lagrangian time-stepping.

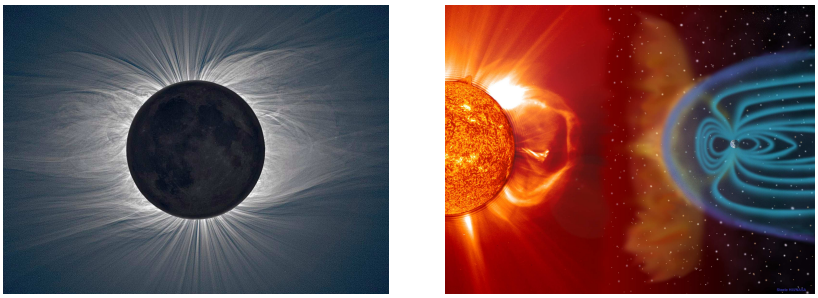


Figure 2: Solar corona and its interaction with the Earth's magnetic field.

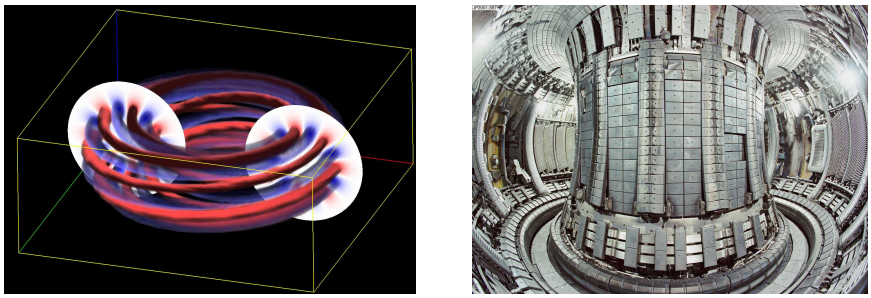


Figure 3: Tokamak design and physical picture.

1.2 Vlasov Models of Plasma

The Vlasov equation in its various incarnations (e.g., Vlasov-Maxwell, Vlasov-Darwin, and Vlasov-Poisson) models the dynamics of collisionless plasma. Because plasma is a mixture of interacting charged particles, it evolves though a variety of effects, including electromagnetic interactions and particle-particle collisions. In the collisionless limit, the mean free-path is much larger than the characteristic length scale of the plasma; and therefore, particle-particle collisions are dropped from the mathematical model. Vlasov models are widely used in both astrophysical applications (e.g., [8, 10, 55]), as well as in laboratory settings (e.g., [14, 62, 48, 13, 40]).

The Vlasov system describes the evolution of a charge density function in phase

space:

$$\tilde{f}_s(\tilde{t}, \tilde{\mathbf{x}}, \tilde{\mathbf{v}}) : \mathbb{R}^+ \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^S, \tag{1.1}$$

which describes the relative number of particles of species s at time t, at location $\tilde{\mathbf{x}}$, and with velocity $\tilde{\mathbf{v}}$. Here, $d = 1, 2$, or 3, is the spatial dimension under consideration, and S represents the number of plasma species.

1.2.1 Vlasov-Maxwell

Under the assumptions of a non-relativistic and collisionless plasma, this distribution function for each species obeys the Vlasov equation, which is an advection equation in $(\tilde{\mathbf{x}}, \tilde{\mathbf{v}})$ phase space:

$$\frac{\partial \tilde{f}_s}{\partial \tilde{t}} + \tilde{\mathbf{v}} \cdot \nabla_{\tilde{\mathbf{x}}} \tilde{f}_s + \frac{q_s}{m_s} \left(\tilde{\mathbf{E}} + \tilde{\mathbf{v}} \times \tilde{\mathbf{B}} \right) \cdot \nabla_{\tilde{\mathbf{v}}} \tilde{f}_s = 0. \tag{1.2}$$

The quantities presented here are mass of species s, m_s, and charge of species s, q_s. The "particles" represented by this kinetic description do not interact through collisional processes; rather they couple indirectly through the electromagnetic field $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{B}}$. In general, the electromagnetic field satisfies Maxwell's equations:

$$\frac{\partial}{\partial \tilde{t}} \begin{bmatrix} \tilde{\mathbf{B}} \\ \tilde{\mathbf{E}} \end{bmatrix} + \nabla_{\tilde{\mathbf{x}}} \times \begin{bmatrix} \tilde{\mathbf{E}} \\ -c^2 \tilde{\mathbf{B}} \end{bmatrix} = \begin{bmatrix} 0 \\ -c^2 \mu_0 \tilde{\mathbf{J}} \end{bmatrix}, \tag{1.3}$$

$$\nabla_{\tilde{\mathbf{x}}} \cdot \tilde{\mathbf{B}} = 0, \quad \nabla_{\tilde{\mathbf{x}}} \cdot \tilde{\mathbf{E}} = \frac{\tilde{\sigma}}{\epsilon_0}, \tag{1.4}$$

where c is the speed of light in vacuum, ϵ_0 is the vacuum permittivity, and μ_0 is the vacuum permeability. The charge density $\tilde{\sigma}$, the net current $\tilde{\mathbf{J}}$, the number density $\tilde{n}$,

and the thermal velocity $\tilde{\mathbf{u}}$ are given by:

$$\tilde{\sigma} = \sum_s q_s \tilde{n}_s, \quad \tilde{\mathbf{J}} = \sum_s q_s \tilde{n}_s \tilde{\mathbf{u}}_s,$$
$$\tilde{n} = \int_{\tilde{\mathbf{v}}} \tilde{f}_s \, d\tilde{\mathbf{v}}, \quad \tilde{\mathbf{u}} = \frac{1}{\tilde{n}} \int_{\tilde{\mathbf{v}}} \tilde{\mathbf{v}} \tilde{f}_s \, d\tilde{\mathbf{v}}.$$

We note that every quantity participating in the evolution of Maxwell's equations depends only on time and spatial coordinates, $\tilde{\mathbf{x}}$. Global coupling occurs through moment integrals of $\tilde{f}_s$.

1.2.2 Vlasov-Poisson

In this work we will not consider the full Vlasov-Maxwell system for a many species plasma; and instead, we only consider the single-species Vlasov-Poisson equation. To arrive at the Vlasov-Poisson system, we start with Vlasov-Maxwell and make two assumptions. First, we assume that the charges are slow-moving in comparison to the speed of light, and, in particular, we assume that $\nabla \times \mathbf{E} = 0$. This allows us to replace the full electromagnetic equations with electrostatics. Furthermore, we consider only two-species: one dynamically evolving species, which we take, without loss of generality, to have positive charge, and in addition, we assume a stationary background species that has a charge of opposite sign to the dynamic species. Because the background charge is stationary, we will only need to solve a single-species Vlasov equation. These assumptions conspire to form the Vlasov-Poisson equations, and with SI units, these are

prescribed by:

$$\tilde{f}_{,\tilde{t}} + \tilde{\mathbf{v}} \cdot \tilde{f}_{,\tilde{\mathbf{x}}} - \frac{q}{m} \nabla_{\tilde{\mathbf{x}}} \tilde{\phi} \cdot \tilde{f}_{,\tilde{\mathbf{v}}} = 0, \tag{1.5}$$

$$-\nabla^2_{\tilde{\mathbf{x}}} \tilde{\phi} = \frac{q}{m\epsilon_0} \left(\tilde{\rho}(\tilde{t}, \tilde{\mathbf{x}}) - \tilde{\rho}_0 \right), \tag{1.6}$$

$$\tilde{\rho}(\tilde{t}, \tilde{\mathbf{x}}) = \int_{\tilde{\mathbf{v}}} m \, \tilde{f} \, d\tilde{\mathbf{v}}, \tag{1.7}$$

$$\tilde{f}(\tilde{t} = 0, \tilde{\mathbf{x}}, \tilde{\mathbf{v}}) = \tilde{f}_0(\tilde{\mathbf{x}}, \tilde{\mathbf{v}}), \tag{1.8}$$

where $\tilde{\phi}$ is the electric potential: $\tilde{\mathbf{E}} = -\nabla_{\tilde{\mathbf{x}}} \tilde{\phi}$, and $-\frac{q}{m}\tilde{\rho}_0$ is the stationary background charge density.

1.2.3 Non-Dimensionalization of Vlasov-Poisson

In order to non-dimesionalize equations (1.5) – (1.8), for each independent and dependent variable we introduce a scaling factor times a non-dimensional variable:

$$\tilde{f} = F f, \quad \tilde{t} = T t, \quad \tilde{\mathbf{x}} = L \mathbf{x}, \quad \tilde{\mathbf{v}} = \left(\frac{L}{T} \right) \mathbf{v},$$

$$\tilde{\mathbf{E}} = E_0 \tilde{\mathbf{E}}, \quad \tilde{\rho} = mN\rho, \quad \tilde{\phi} = \Phi_0 \, \phi.$$

Plugging in these scaled variables yields:

$$\left(\frac{F}{T} \right) f_{,t} + \left(\frac{L}{T} \right) \left(\frac{F}{L} \right) \mathbf{v} \cdot f_{,\mathbf{x}} + \frac{q}{m} \left(\frac{FT}{L} \right) E_0 \mathbf{E} \cdot f_{,\mathbf{v}} = 0, \tag{1.9}$$

$$\left(\frac{E_0}{L} \right) \nabla_{\mathbf{x}} \cdot \mathbf{E} = \frac{q}{m\epsilon_0} \left(mN \right) \left(\rho - \rho_0 \right), \tag{1.10}$$

$$mN\rho = \frac{mFL^d}{T^d} \int_{\mathbb{R}^d} f \left(t, \mathbf{x}, \mathbf{v} \right) \, d\mathbf{v}. \tag{1.11}$$

Simplifying this yields:

$$f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} + \left(\frac{qE_0T^2}{mL}\right) \mathbf{E} \cdot f_{,\mathbf{v}} = 0, \tag{1.12}$$

$$\nabla_{\mathbf{x}} \cdot \mathbf{E} = \left(\frac{qLN}{\epsilon_0 E_0}\right) (\rho - \rho_0), \tag{1.13}$$

$$\rho = \left(\frac{FL^d}{NT^d}\right) \int_{\mathbb{R}^d} f(t, \mathbf{x}, \mathbf{v}) \, d\mathbf{v}. \tag{1.14}$$

In order to simplify this problem, we strive to make the factors in parentheses in the above formulas equal to one. We assume that the electron density variable, N, and the length scale, L, are fixed. First we look at the constant in equation (1.13) and set it to one:

$$\left(\frac{qLN}{\epsilon_0 E_0}\right) = 1 \quad \Longrightarrow \quad E_0 = \frac{qLN}{\epsilon_0}. \tag{1.15}$$

We continue by setting the parameters in (1.12) and (1.14) equal to one:

$$\left(\frac{qE_0T^2}{mL}\right) = \left(\frac{q^2NT^2}{\epsilon_0 m}\right) = 1 \quad \Longrightarrow \quad T = \sqrt{\frac{\epsilon_0 m}{q^2 N}}; \tag{1.16}$$

$$\left(\frac{FL^d}{NT^d}\right) = \frac{F}{N}\left(\frac{q^2L^2N}{\epsilon_0 m}\right)^{\frac{d}{2}} = 1 \quad \Longrightarrow \quad F = \frac{N^{\frac{2-d}{2}}(\epsilon_0 m)^{\frac{d}{2}}}{q^d L^d}. \tag{1.17}$$

Finally we note the following relationship between the electrostatic potential and the electric field:

$$\tilde{\mathbf{E}} = -\nabla_{\tilde{\mathbf{x}}}\tilde{\phi} \quad \Longrightarrow \quad \mathbf{E} = -\left(\frac{\Phi_0}{E_0 L}\right)\nabla_{\mathbf{x}}\phi \quad \Longrightarrow \quad \Phi_0 = E_0 L. \tag{1.18}$$

After scaling, we arrive at (with non-dimensional units) what will henceforth be referred to as the Vlasov-Poisson system,

$$f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} - \nabla_{\mathbf{x}}\phi \cdot f_{,\mathbf{v}} = 0, \tag{1.19}$$

$$-\nabla_{\mathbf{x}}^2 \phi = \rho(t, \mathbf{x}) - \rho_0, \tag{1.20}$$

where $\mathbf{E} = -\nabla_{\mathbf{x}}\phi$.

To summarize, for a given problem there are two free parameters to choose, N and L. The parameters E_0, T, F and Φ_0 are then determined based on these two parameters. In SI units, these two free parameters have units $[N] = \text{meters}^{-d}$ and $[L] = \text{meters}$.

For the bulk of this work, we consider the case of the 1+1 dimensional version of the above equations with periodic boundary conditions in x. In this case, the Vlasov-Poisson system on $\Omega = (t, x, v) \in \mathbb{R}^+ \times [0, L] \times \mathbb{R}$ is:

$$f_{,t} + v f_{,x} + E(t, x)\, f_{,v} = 0, \tag{1.21}$$

$$E_{,x} = \rho(t, x) - \rho_0. \tag{1.22}$$

The total and background densities are

$$\rho(t, x) := \int_{-\infty}^{\infty} f(t, x, v)\, dv \qquad \text{and} \qquad \rho_0 := \frac{1}{L}\int_0^L \rho(t = 0, x)\, dx. \tag{1.23}$$

Note that ρ_0 is determined at the start of the problem. However, in the case of a periodic domain, conservation of mass implies that $\rho_0 = \frac{1}{L}\int_0^L \rho(t, x)\, dx$ for all time. For 1D problems, we will use:

$$x := x^1, \quad v := v^1, \quad \text{and} \quad E := E^1,$$

and for 2D problems, we will use the shorthand notation:

$$\mathbf{x} := (x^1, x^2), \quad \mathbf{v} = (v^1, v^2), \quad \mathbf{E} = (E^1, E^2).$$

1.2.4 Moments and Conserved Quantities

The Vlasov-Poisson system contains an infinite number of quantities that are conserved in time. Any number of these can be used as diagnostics in a numerical discretization,

and in equations (1.50) – (1.52) we summarize the four conserved quantities we compute with our numerical scheme. Although the probability density function (PDF) is not itself a physical observable, moments of the PDF represent various physically observable quantities and are necessary for describing conserved quantities. The first three moments are defined as:

$$\rho(t,\mathbf{x}) := \int_{\mathbb{R}^d} f\, d\mathbf{v}, \qquad \text{(mass density)}, \tag{1.24}$$

$$\rho\mathbf{u}(t,\mathbf{x}) := \int_{\mathbb{R}^d} \mathbf{v}\, f\, d\mathbf{v}, \qquad \text{(momentum density)}, \tag{1.25}$$

$$\mathcal{E}(t,\mathbf{x}) := \frac{1}{2}\int_{\mathbb{R}^d} \|\mathbf{v}\|^2 f\, d\mathbf{v}, \qquad \text{(energy density)}. \tag{1.26}$$

In addition, there are infinitely many moment equations that the Vlasov-Poisson equations satisfy. These are derived by multiplying the Vlasov equation (1.19) by increasing powers of $\mathbf{v}$, and integrating by parts. The first two evolution equations for these moments are prescribed by:

$$\rho_{,t} + \nabla\cdot(\rho\mathbf{u}) = 0, \tag{1.27}$$

$$(\rho\mathbf{u})_{,t} + \nabla\cdot\mathbb{E} = \rho\mathbf{E}; \quad \mathbb{E} := \int_{\mathbf{v}} \mathbf{v}\mathbf{v} f\, d\mathbf{v}. \tag{1.28}$$

Conservation of L^p-norms

Equations (1.49) and (1.50) are the special case of $p = 1$ and $p = 2$ for conservation of L^p norms. To derive these, we multiply the Vlasov equation (1.19) by pf^{p-1} and use the product rule. This yields:

$$\begin{aligned} & pf^{p-1}f_{,t} + \mathbf{v}\cdot\left(pf^{p-1}f_{,\mathbf{x}}\right) + \mathbf{E}\cdot\left(pf^{p-1}f_{,\mathbf{v}}\right) = 0, \\ \implies \quad & (f^p)_{,t} + \mathbf{v}\cdot(f^p)_{,\mathbf{x}} + \mathbf{E}\cdot(f^p)_{,\mathbf{v}} = 0, \\ \implies \quad & (f^p)_{,t} + \nabla_{\mathbf{x}}\cdot(\mathbf{v}f^p) + \nabla_{\mathbf{v}}\cdot(\mathbf{E}f^p) = 0. \end{aligned}$$

If we integrate this equation over $(\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times \mathbb{R}^d$, and assume that $f \to 0$ sufficiently fast as $\|\mathbf{x}\|, \|\mathbf{v}\| \to \infty$ and note that $f > 0$ for all $(t, \mathbf{x}, \mathbf{v})$, we arrive at the conclusion that the following quantity is constant in time for any $p \geq 1$:

$$\|f\|_{L^p} = \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left|f\right|^p d\mathbf{v}\, d\mathbf{x}. \tag{1.29}$$

Energy conservation

Energy conservation starts with considering the next moment after (1.28). Multiplying the Vlasov equation (1.19) by $\mathbf{vv}$ and integrating over all of $\mathbf{v} \in \mathbb{R}^d$ yields the energy tensor evolution equation:

$$\mathbb{E}_{,t} + \nabla \cdot \mathbb{F} = \rho \left(\mathbf{uE} + \mathbf{Eu}\right). \tag{1.30}$$

The scalar energy is defined as

$$\mathcal{E} := \frac{1}{2}\mathrm{tr}\left(\mathbb{E}\right) = \frac{1}{2}\left(\mathbb{E}^{11} + \mathbb{E}^{22} + \mathbb{E}^{33}\right) = \frac{1}{2}\int_{\mathbb{R}^d} \|\mathbf{v}\|^2 f\, d\mathbf{v}. \tag{1.31}$$

Therefore, the evolution equation for the scalar energy is obtained by taking the trace of the energy tensor evolution equation:

$$\mathcal{E}_{,t} + \nabla \cdot \vec{\mathcal{F}} - \rho \mathbf{u} \cdot \mathbf{E} = 0, \tag{1.32}$$

where

$$\mathcal{F}^i := \frac{1}{2}\sum_{k=1}^{d} \mathbb{F}^{ikk}. \tag{1.33}$$

Next we take the time derivative of the divergence constraint on the electric field and obtain:

$$\nabla \cdot \mathbf{E}_{,t} = \rho_{,t} = -\nabla \cdot (\rho \mathbf{u}), \tag{1.34}$$

$$\implies \mathbf{E}_{,t} = -\rho \mathbf{u} + \nabla \times \mathbf{C}, \tag{1.35}$$

$$\implies \mathbf{E} \cdot \mathbf{E}_{,t} = -\rho \mathbf{u} \cdot \mathbf{E} + \mathbf{E} \cdot \nabla \times \mathbf{C}, \tag{1.36}$$

$$\implies \left(\frac{1}{2} \|\mathbf{E}\|^2 \right)_{,t} - \mathbf{E} \cdot \nabla \times \mathbf{C} = -\rho \mathbf{u} \cdot \mathbf{E}, \tag{1.37}$$

$$\implies \left(\frac{1}{2} \|\mathbf{E}\|^2 \right)_{,t} + \nabla \cdot (\mathbf{E} \times \mathbf{C}) - \mathbf{C} \cdot \nabla \times \mathbf{E} = -\rho \mathbf{u} \cdot \mathbf{E}, \tag{1.38}$$

$$\implies \left(\frac{1}{2} \|\mathbf{E}\|^2 \right)_{,t} + \nabla \cdot (\mathbf{E} \times \mathbf{C}) = -\rho \mathbf{u} \cdot \mathbf{E}. \tag{1.39}$$

Adding this last result into the energy tensor evolution equation (1.32) results in the following conservation law:

$$\left(\mathcal{E} + \frac{1}{2} \|\mathbf{E}\|^2 \right)_{,t} + \nabla \cdot \left(\vec{\mathcal{F}} + \mathbf{E} \times \mathbf{C} \right) = 0. \tag{1.40}$$

From this we conclude that the following quantity, referred to as the *total energy*, is constant in time:

$$\int_{\mathbb{R}^d} \left(\mathcal{E} + \frac{1}{2} \|\mathbf{E}\|^2 \right) d\mathbf{x} = \frac{1}{2} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \|\mathbf{v}\|^2 f \, d\mathbf{v} \, d\mathbf{x} + \frac{1}{2} \int_{\mathbb{R}^d} \|\mathbf{E}\|^2 \, d\mathbf{x}. \tag{1.41}$$

Entropy conservation

The Vlasov-Poisson equations also satisfy an entropy conservation property. Consider the following identities:

$$(f \log f)_{,t} = f_{,t} + f_{,t} \log f = (1 + \log f) \, f_{,t}, \tag{1.42}$$

$$(f \log f)_{,\mathbf{x}} = f_{,\mathbf{x}} + f_{,\mathbf{x}} \log f = (1 + \log f) \, f_{,\mathbf{x}}, \tag{1.43}$$

$$(f \log f)_{,\mathbf{v}} = f_{,\mathbf{v}} + f_{,\mathbf{v}} \log f = (1 + \log f) \, f_{,\mathbf{v}}. \tag{1.44}$$

Multiplying the Vlasov equation by $(\log f + 1)$ and using the above identities yields

$$(\log f + 1) f_{,t} + \mathbf{v} \cdot \{(\log f + 1) f_{,\mathbf{x}}\} + \mathbf{E} \cdot \{(\log f + 1)\, f_{,\mathbf{v}}\} = 0, \tag{1.45}$$

$$\implies \quad (f \log f)_{,t} + \mathbf{v} \cdot (f \log f)_{,\mathbf{x}} + \mathbf{E} \cdot (f \log f)_{,\mathbf{v}} = 0, \tag{1.46}$$

$$\implies \quad (f \log f)_{,t} + \nabla_{\mathbf{x}} \cdot (\mathbf{v} f \log f) + \nabla_{\mathbf{v}} \cdot (\mathbf{E} f \log f) = 0. \tag{1.47}$$

If we integrate this equation over $(\mathbf{x}, \mathbf{v}) \in \mathbb{R}^d \times \mathbb{R}^d$ and assume that $f \log f \to 0$ sufficiently fast as $\|\mathbf{x}\|, \|\mathbf{v}\| \to \infty$, we arrive at the conclusion that the following quantity, referred to as the *total entropy*, is constant in time:

$$-\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} f\ \log(f)\, d\mathbf{v}\, d\mathbf{x}. \tag{1.48}$$

Summary

To summarize, we list the four quantities that will be used in diagnosing our proposed scheme, all of which should remain constant in time:

$$\|f\|_{L_1} := \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \big|f\big|\, d\mathbf{v}\, d\mathbf{x}, \tag{1.49}$$

$$\|f\|_{L_2} := \left(\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} f^2\, d\mathbf{v}\, d\mathbf{x} \right)^{\frac{1}{2}}, \tag{1.50}$$

$$\text{Total energy} := \frac{1}{2} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \|\mathbf{v}\|^2 f\, d\mathbf{v}\, d\mathbf{x} + \frac{1}{2} \int_{\mathbb{R}^d} \|\mathbf{E}\|^2\, d\mathbf{x}, \tag{1.51}$$

$$\text{Entropy} := -\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} f\ \log(f)\, d\mathbf{v}\, d\mathbf{x}. \tag{1.52}$$

1.2.5 Known Theoretical Results for Vlasov-Poisson

The Vlasov system was first suggested by Anatoly Vlasov in 1938. Despite its age, theoretical results are still an active area of research. Many modern results were proven in the 70's, 80's, and 90's. To date, the Vlasov system provides a rich source of research

problems. Indeed, Cédrik Villani [47] earned the prestigious 2010 Fields medal "for his proofs of nonlinear Landau damping and convergence to equilibrium for the Boltzmann equation." An excellent review of known long-time existence results for the Vlasov-Poisson system can be found in Chapter 4 of Glassey's text on kinetic theory [32]. We do not attempt to reproduce Glassey's summary here, but simply point out that there are several rigorous long-time existence results for Vlasov-Poisson. For example, Schaeffer [54] proved the existence of global smooth solutions for the Vlasov-Poisson system in 3D.

1.3 Numerical Methods for Vlasov-Poisson

The Vlasov system is faced with a number of numerical challenges, and hence there are a number of numerical schemes devised to overcome these challenges. In this section, we describe what hurdles need to be overcome and we summarize popular numerical schemes used to overcome these challenges.

1.3.1 Numerical Challenges

Below we briefly describe the three main challenges in the numerical solution of the Vlasov system.

High dimensionality

The Vlasov system is a nonlinear and nonlocal advection equation in six phase space dimensions ($\mathbf{x} \in \mathbb{R}^3$ and $\mathbf{v} \in \mathbb{R}^3$) and time – this is often referred to as $3 + 3 + 1$ dimensions. Even though the Vlasov equation is in many ways mathematically simpler

than fluid models, the fact that it lives in a space of twice the number of dimensions makes it computationally much more expensive to solve.

Conservation and positivity

In fluid models, conservation of mass, momentum, and energy are often relatively easy to guarantee in a numerical discretization, since each of these quantities is a dependent variable of the system. In Vlasov models it is generally more difficult to exactly maintain these quantities in the numerical discretization. Exact positivity of the probability density function is also not guaranteed by many standard discretizations of the Vlasov system; therefore, additional work in choosing the correct approximation spaces is often required.

Small time steps due to $\mathbf{v} \in \mathbb{R}^3$.

In the non-relativistic case, the advection velocity of the density function in phase space depends linearly on the components of the velocity vector $\mathbf{v} \in \mathbb{R}^3$ (see equation (1.2) in §1.2.2). Since it is in general possible to have "particles" in the Vlasov system that travel arbitrarily fast, there will be a severe time-step restriction, relative to the dynamics of interest, that arises from parts of the PDF associated with moderate-to-large velocities.

1.3.2 Numerical Methods

Several approaches have been introduced to try to solve some of these problems, including particle-in-cell methods, Lagrangian particle methods, and grid-based semi-Lagrangian methods. We briefly summarize each of these approaches below.

Particle-in-cell methods

Particle-in-cell (PIC) methods are ubiquitous in both astrophysical (e.g., [62]) and laboratory plasma (e.g., [10]) application problems. The basic approach is outlined in the celebrated textbooks of Birdsall and Langdon [9] and Hockney and Eastwood [39], both of which appeared in the mid-to-late 1980s. Modern improvements to these methods are still topics of current research (e.g., adaptive mesh refinement [62], very high-order variants [42, 41], etc.). The basic idea is that the distribution function is discretized into a set of macro-particles (Lagrangian representation), while the electromagnetic field is represented on a mesh (Eulerian representation). The main advantages of this approach are that positivity and mass conservation are essentially automatic, the small time step restriction is removed due to the fact that the particles are evolved in a Lagrangian framework, and the electromagnetic equations can be solved via standard mesh-based methods. The main disadvantages of this method are: (1) numerical errors are introduced due to the interpolations that must de done to exchange information between the particles and fields, (2) error control is non-trivial since particles may either cluster or generate rarefied regions during the evolution of the plasma, and (3) statistical noise from sampling scales as $\mathcal{O}(1/\sqrt{N})$, where N is the number of particles, which means that an incredibly large number of particles needs to be sampled in order to drive this error down.

Lagrangian particle methods

One possible alternative to the PIC methodology is to go to a completely Lagrangian framework – this removes the need to interpolate between the particles and fields. Such

approaches are commonplace in several application areas such as many body dynamics in astrophysics [4], vortex dynamics [45], as well as in plasma physics [15]. The key is that the potential (e.g., gravitational potential, stream-function, or electric potential) is calculated by integrating the point charges represented by the Lagrangian particles against a Green's function. Since the charges are point particles, evaluating this integral reduces to computing sums over the particles. Naive methods would need $\mathcal{O}(N^2)$ floating point operations to evaluate all of these sums, where N is the number of particles, but fast summation methods such as treecode methods [4, 45] and the fast multipole method [34] can be used to reduce this to $\mathcal{O}(N \log N)$. The main disadvantage of this approach is that it relies on having a Green's function, which for more complicated dynamics (i.e., full electromagnetism) may be difficult to obtain.

Semi-Lagrangian grid-based methods

Another alternative to PIC is to switch to a completely grid-based method. Such an approach allows for a variety of high-order spatial discretizations, and can be evolved forward in time via so-called *semi-Lagrangian* time-stepping. The basic idea is that the PDF sits initially on a grid; the PDF is then evolved forward in time using Lagrangian dynamics; and finally, the new PDF is projected back onto the original mesh. This gives many of the advantages of particle methods (i.e., no small time-step restrictions), but retains a nice grid structure for both the PDF and the fields, allowing extension to very high-order accuracy. There have been several contributions to this approach over the last few years. One of the first papers that developed a viable semi-Lagrangian method was put forward by Cheng and Knorr [12]. More recent activity on this approach includes the work of Parker and Hitchon [48], Sonnendrücker and his collaborators (see

for example [25, 30, 59, 22, 6, 24, 26, 5]), and Christlieb and Qiu [49].

1.4 Scope of This Work

The primary focus of this work is to develop a grid based, high-order, semi-Lagrangian alternative to traditional particle-in-cell (PIC) methods, which is the predominant approach in the plasma physics community for Vlasov simulations. Much like other recent work, e.g., [28, 43, 49], our high-order semi-Lagrangian method starts with the classical Cheng and Knorr [12] operator splitting method. We attain high-order in space by using a discontinuous Galerkin spatial discretization for the proposed method, which we review in Chapter 2. The heart of this thesis is located in Chapters 3 and 4, where we describe novel work on semi-Lagrangian and hybrid semi-Lagrangian DG methods for the Vlasov system. In particular, we describe a method that attains high-order accuracy in space and time, mass conservation, and positivity of the distribution function. In these Chapters we include a number of standard test cases for the Vlasov-Poisson system: the two-stream instability, bump on tail, and Landau damping. In addition, we present results for a plasma sheath problem with non-periodic boundary conditions.

In Chapter 5 we present a forward looking view towards developing a full (2+2)D Vlasov solver through a hybrid DG solver that is tested and developed in Chapter 4. Currently, all of our Vlasov results full are in (1+1)D, however, we demonstrate that the necessary tools are in place for extending this to (2+2)D. This extension will operate on unstructured grids in physical space, have high-order accuracy, utilize sub-cycling to alleviate strict CFL conditions, and retain positivity of the probability density function. We note that there are currently very few research groups working with grid based

Vlasov solvers in high dimensions, e.g., [25, 58, 2, 38]; therefore, one of the chief goals of the present work is a push towards that direction, retaining all the machinery developed in this thesis relating to high-order, large time-steps, mass conservation, and positivity-preservation.

In this thesis, we argue that our approach is a promising method that can produce very accurate results at relatively low computational expense. We demonstrate this through several examples for the (1+1)D case. We argue that our proposed method for the (2+2)D problem proves to be a promising avenue of research. Through the use of semi-Lagrangian time-stepping and unstructured grids, we hope that ultimately the methods presented in this work will be viewed as a bridge between particle and pure Eulerian methods, in the sense that we retain the ability to take large time-steps (semi-Lagrangian) and can handle complex geometries (unstructured meshes).

Chapter 2

Discontinuous Galerkin Methods

The entirety of this chapter is dedicated to a review of classical discontinuous Galerkin (DG) methods. The purpose is to provide the necessary background, as well as define notation used throughout this dissertation.

DG methods offer a high-order mechanism for solving partial differential equations (PDEs), and in particular, they excel at capturing rough data while maintaining a high-order of accuracy. Hyperbolic problems often require methods that are able to capture discontinuities (shocks) and are high-order elsewhere in order to maintain approximate solution fidelity. DG schemes as well as their finite volume and finite difference counterparts, WENO and ENO, are primarily reserved for working on hyperbolic problems because of their abilities to perform all of the above. However, they do tend to be more computationally expensive to run. The history of DG schemes dates back to 1973, where Reed & Hill [50] invented the DG method as a method for solving a neutron transport equation. Modern work solidified the theoretical background through a series of papers by Bernardo Cockburn, Chi-Wang Shu, and their collaborators [21, 19, 18, 17, 20].

The use of the word 'discontinuous' when used in conjunction with 'high-order' deserves some explanation. The term discontinuous in DG refers to the fact that the polynomial representations for the solution come from so-called *broken finite element*

spaces, which are defined later in (2.2) and (2.33). In these finite element spaces, the basis functions that are used for the representation (typically polynomials) are not forced to be continuous across interfaces (points in 1D, edges in 2D, faces in 3D, etc...); and therefore, the representation for the solution will have discontinuities. High-order means that when an exact solution has enough regularity, the representation will be high-order in L^p, with $1 \leq p \leq \infty$. In the case of $p = \infty$, we see that an M^{th}-order method will produce a solution that is pointwise accurate to $\mathcal{O}(h^M)$, where h refers to the mesh size. In particular, this means that the size of a jump in the representation of the solution across an interface has a magnitude $\mathcal{O}(h^M)$.

One key difference between DG and WENO/ENO is the use of a localized stencil in place of a broad stencil. Localized stencils are useful when trying to capture sharp gradients or discontinuities in a solution; e.g., see Zhou, Tie and Shu [69] for a comparison. One criticism of DG methods when compared to their counterparts is the large number of unknowns required per element for high-order, especially in higher dimensions. While WENO and ENO methods extend their stencils to attain high-order, DG, just as other finite element approaches, increases the number of basis functions, and hence unknowns, per grid cell. In particular, the number of unknowns for operating in d dimensions for an M^{th} order method grows as $\mathcal{O}(M^d)$. In dimension $d = 1$, this is simply M, and in $d = 2$, the minimal number of unknowns is $M(M+1)/2$. In the case of $d = 2$ and $M = 5$ there are 15 unknowns per element. In general, the exact formula for the minimal amount of moments is given by $\binom{M+d-1}{d}$.

We begin by describing DG methods for a 1D hyperbolic problem, and then we continue by describing a classical implementation of the DG method on a 2D grid, first on a Cartesian mesh, then on a triangle-based unstructured mesh.

2.1 1D Hyperbolic Conservation Laws

Consider a generic 1D hyperbolic balance law of the form

$$q_{,t} + f(q, x, t)_{,x} = \psi(q, x, t). \tag{2.1}$$

Here, $q : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^m$ is the unknown quantity, f is the flux function, and ψ is the source term. Equation (2.1) is hyperbolic if and only if the flux Jacobian,

$$A(q, x, t) := \frac{\partial f}{\partial q}(q, x, t)$$

is diagonalizable with real eigenvalues for all q, x and t in the domain of interest.

A DG solver starts by creating a grid on $[a, b]$ with m_x cells, each of whose width is given by $\Delta x = (b - a)/m_x$. We denote the i^{th} grid cell by $\mathcal{T}_i = [x_{i-1/2}, x_{i+1/2}]$, where the cell edges are given by $x_{i-1/2} = a + (i - 1)\Delta x$, for $i = 1, 2, \ldots, m_x + 1$, and the cell centers are given by $x_i = a + (i - 1/2)\Delta x$, for $i = 1, 2, \ldots, m_x$. For simplicity of exposition, we will restrict our attention to a uniform grid, however, DG methods can certainly be written to accommodate non-uniform, 1D grids.

On this grid we define the *broken* finite element space

$$W^h = \left\{ w^h \in L^\infty(\Omega) : \, w^h|_{\mathcal{T}} \in P^q, \, \forall \mathcal{T} \in \mathcal{T}_h \right\}, \tag{2.2}$$

where $h = \Delta x$. The above expression means that on each element $\mathcal{T}$, w^h will be a polynomial of degree at most q, and no continuity is assumed across element edges. Each element can be mapped to the canonical element $\xi \in [-1, 1]$ via the linear transformation:

$$x = x_i + \xi \frac{\Delta x}{2}. \tag{2.3}$$

For each element, we construct a set of basis functions that are orthonormal with respect to the following inner product:

$$\left\langle \varphi_{1\mathrm{D}}^{(\ell)},\, \varphi_{1\mathrm{D}}^{(k)} \right\rangle := \frac{1}{2}\int_{-1}^{1} \varphi_{1\mathrm{D}}^{(\ell)}(\xi)\varphi_{1\mathrm{D}}^{(k)}(\xi)\; d\xi = \delta_{\ell k},$$

where $\delta_{\ell k}$ is the Kronecker delta function. Starting with $\varphi_{1\mathrm{D}}^{(1)} = 1$, and proceeding via the Gram-Schmidt process, these define what are called the Legendre basis functions. Up to 5^{th} order, the Legendre basis functions are given by:

$$\varphi_{1\mathrm{D}}^{(\ell)} = \left\{ 1,\;\; \sqrt{3}\,\xi,\;\; \frac{\sqrt{5}}{2}\left(3\xi^2-1\right),\;\; \frac{\sqrt{7}}{2}(5\xi^3-3\xi),\;\; \frac{3}{8}\left(35\xi^4-30\xi^2+3\right) \right\}.$$

Of course other basis functions may be used; this is the 'Galerkin' part of DG.

We will look for approximate solutions of (2.1) that have the following form:

$$\tilde{q}^h(t,x)\Big|_{\mathcal{T}_{ij}} := q^h(t,\xi) = \sum_{k=1}^{M} Q_i^{(k)}(t)\,\varphi_{1\mathrm{D}}^{(k)}(\xi), \tag{2.4}$$

where M is the desired order of accuracy in space.

The Legendre coefficients of the initial conditions at $t = 0$ are determined from the L_2-projection of $q^h(0,x)$ onto the Legendre basis functions:

$$Q_{ij}^{(k)}(0) := \left\langle q^h(0,\xi),\, \varphi_{1\mathrm{D}}^{(k)}(\xi) \right\rangle. \tag{2.5}$$

In practice, these integrals can be evaluated to high-order using M standard Gaussian quadrature points.

In order to determine the evolution of the coefficients of the basis functions, we multiply (2.1) by a test function φ, and integrate by parts over grid cell $\mathcal{T}_i$:

$$\begin{aligned}\frac{\partial}{\partial t}\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} q(t,x)\varphi\; dx = {} & \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} f(t,x)\varphi_{,x}\; dx \\ & - \frac{1}{\Delta x}\left[f^{\downarrow}(t,x_{i+1/2})\varphi(x_{i+1/2}) - f^{\downarrow}(t,x_{i-1/2})\varphi(x_{i-1/2})\right] \\ & + \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \psi(q,x,t)\varphi(\xi(x))\, dx\end{aligned}$$

After setting $\varphi = \varphi^{(k)}$, this yields a large system of coupled differential equations:

$$\frac{d}{dt}\begin{bmatrix} Q_i^{(1)} \\ Q_i^{(2)} \\ Q_i^{(3)} \\ Q_i^{(4)} \\ Q_i^{(5)} \end{bmatrix} = \begin{bmatrix} 0 \\ N_i^{(2)} \\ N_i^{(3)} \\ N_i^{(4)} \\ N_i^{(5)} \end{bmatrix} - \frac{1}{\Delta x}\begin{bmatrix} \left(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}\right) \\ \sqrt{3}\left(F_{i+\frac{1}{2}} + F_{i-\frac{1}{2}}\right) \\ \sqrt{5}\left(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}\right) \\ \sqrt{7}\left(F_{i+\frac{1}{2}} + F_{i-\frac{1}{2}}\right) \\ 3\left(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}\right) \end{bmatrix} + \begin{bmatrix} \Psi_i^{(1)} \\ \Psi_i^{(2)} \\ \Psi_i^{(3)} \\ \Psi_i^{(4)} \\ \Psi_i^{(5)} \end{bmatrix}, \tag{2.6}$$

for each $i = 1, 2, \ldots, m_x$. Each $N_i^{(k)}$ represents the interior integral, and after a change of variables, can be evaluated on the canonical element via:

$$N_i^{(k)} = \frac{1}{\Delta x}\int_{-1}^{1} f\left(q\left(x_i + \xi\,\Delta x/2\right)\right)\varphi^{(k)}(\xi)_{,\xi}\,d\xi.$$

In terms of our Legendre polynomials, these are given by

$$N_i^{(1)} = 0, \tag{2.7}$$

$$N_i^{(2)} = \frac{\sqrt{3}}{\Delta x}\int_{-1}^{1} f\left(x_i + \xi\,\Delta x/2\right)\,d\xi, \tag{2.8}$$

$$N_i^{(3)} = \frac{3\sqrt{5}}{\Delta x}\int_{-1}^{1} \xi\, f\left(x_i + \xi\Delta x/2\right)\,d\xi, \tag{2.9}$$

$$N_i^{(4)} = \frac{3\sqrt{7}}{2\Delta x}\int_{-1}^{1} \left(5\,\xi^2 - 1\right)\, f\left(x_i + \xi\Delta x/2\right)\,d\xi, \tag{2.10}$$

$$N_i^{(5)} = \frac{15}{2\Delta x}\int_{-1}^{1} \left(7\,\xi^3 - 3\,\xi\right)\, f\left(x_i + \xi\Delta x/2\right)\,d\xi. \tag{2.11}$$

The source terms $\Psi_i^{(k)}$ are likewise given by

$$\Psi_i^{(k)} = \frac{1}{2}\int_{-1}^{1} \psi\left(x_i + \xi\,\Delta x/2\right)\varphi^{(k)}(\xi)\,d\xi. \tag{2.12}$$

Because $q(t, x_{i+1/2})$ is discontinuous at the cell boundary, $F_{i+1/2} = f^{\downarrow}(q(t, x_{i+1/2}))$ is not defined, and therefore requires some care. These numerical fluxes $F_{i+1/2}$ are obtained

by solving an approximate Riemann problem between the following states:

$$F_{i+\frac{1}{2}} = F(Q_\ell, Q_r): \quad \begin{cases} Q_r = Q_{i+1}^{(1)} - \sqrt{3}\, Q_{i+1}^{(2)} + \sqrt{5}\, Q_{i+1}^{(3)} - \sqrt{7}\, Q_i^{(4)} + 3\, Q_i^{(5)}, \\ Q_\ell = Q_i^{(1)} + \sqrt{3}\, Q_i^{(2)} + \sqrt{5}\, Q_i^{(3)} + \sqrt{7}\, Q_i^{(4)} + 3\, Q_i^{(5)}. \end{cases}$$

We note that the basis functions are always evaluated on the interior of the integral, and cross communication between grid cells happens via the solution to a Riemann problem. Usually, a local-Lax-Friedrichs or an HLLE-type numerical flux is used.

2.1.1 HLLE approximate Riemann solver

At each cell interface, x_i, the discontinuous Galerkin method requires a numerical flux: F_i. One approach for obtaining such a flux is to exactly solve the *generalized Riemann problem* between the solution in grid cell $\mathcal{T}_i$ and $\mathcal{T}_{i+1}$:

$$q_{,t} + f(q,x)_{,x} = 0 \quad \text{in} \quad -\infty < x < \infty, \quad \text{with} \tag{2.13}$$

$$q^h(0,x) = \begin{cases} Q_r & x > x_{i+1/2}, \\ Q_\ell & x < x_{i+1/2}, \end{cases} \tag{2.14}$$

where $Q_r = q^h|_{\mathcal{T}_{i+1}}$ and $Q_l = q^h|_{\mathcal{T}_i}$. This is in general far too complicated due to the spatial variations; and instead, one could approximate the spatially varying Q_r and Q_ℓ with constants:

$$Q_r = Q_{i+1}^{(1)} - \sqrt{3}\, Q_{i+1}^{(2)} + \sqrt{5}\, Q_{i+1}^{(3)} - \sqrt{7}\, Q_{i+1}^{(4)} + 3\, Q_{i+1}^{(5)}, \tag{2.15}$$

$$Q_\ell = Q_i^{(1)} + \sqrt{3}\, Q_i^{(2)} + \sqrt{5}\, Q_i^{(3)} + \sqrt{7}\, Q_i^{(4)} + 3\, Q_i^{(5)}, \tag{2.16}$$

thus arriving at what is usually referred to as the *Riemann problem*. Here Q_r and Q_ℓ are simply values of $q^h|_{\mathcal{T}_{i+1}}$ and $q^h|_{\mathcal{T}_i}$ at $x = x_{i+1/2}$, respectively. Although exactly solving the

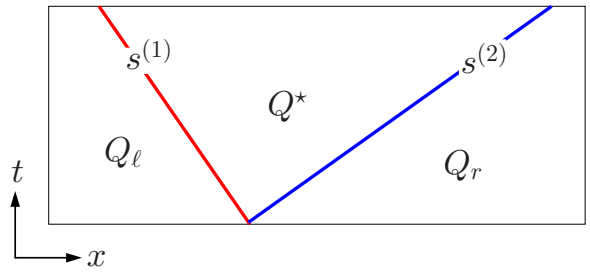


Figure 4: HLLE approximate Riemann solution. The two constant states, Q_ℓ and Q_r are connected to a single intermediate state $Q^\star$ by two shock waves moving at speeds $s^{(1)}$ and $s^{(2)}$.

Riemann problem is far simpler than exactly solving the generalized Riemann problem, for many applications this approach is still too computationally expensive.

An alternative to the exact Riemann solution is the approximate method of Harten, Lax, and van Leer [35], which was slightly modified by Einfeldt [29], and hence, has since been referred to in the literature as the HLLE approach. The idea is to approximate the Riemann solution by two shockwaves that separate a single constant state, $Q^\star$, from the constant states (2.15) and (2.16). We denote the speeds of the two shockwaves by $s^{(1)}$ and $s^{(2)}$ and by convention we take $s^{(1)} < s^{(2)}$. This scenario is depicted in Figure 4.

Conservation requires the following Rankine-Hugoniot condition be satisfied:

$$s^{(1)}\left(Q^\star - Q_\ell\right) + s^{(2)}\left(Q_r - Q^\star\right) = f(Q_r) - f(Q_\ell). \tag{2.17}$$

Solving this expression for the intermediate state $Q^\star$ yields

$$Q^\star = \frac{f(Q_r) - f(Q_\ell) + s^{(1)}Q_\ell - s^{(2)}Q_r}{s^{(1)} - s^{(2)}}. \tag{2.18}$$

The resulting flux at the interface separating Q_ℓ and Q_r is given by

$$F_i = \begin{cases} \frac{1}{2}\big(f(Q_\ell) + f(Q_r) + (s^{(1)} + s^{(2)})Q^\star \\ \qquad -s^{(1)}Q_\ell - s^{(2)}Q_r\big) & \text{if} \quad s^{(1)} < 0 \quad \text{and} \quad s^{(2)} > 0, \\ f\left(Q_\ell\right) & \text{if} \quad s^{(1)} \geq 0 \quad \text{and} \quad s^{(2)} > 0, \\ f\left(Q_r\right) & \text{if} \quad s^{(1)} < 0 \quad \text{and} \quad s^{(2)} \leq 0. \end{cases} \tag{2.19}$$

For stability, the shock speeds must to be chosen to enclose the shock structure of the exact Riemann solution. In practice one takes

$$s^{(1)} = \min\left\{\min_p\left\{\lambda^{(p)}(Q_\ell)\right\}, \min_p\left\{\lambda^{(p)}(\hat{Q})\right\}\right\} - \epsilon, \tag{2.20}$$

$$s^{(2)} = \max\left\{\max_p\left\{\lambda^{(p)}(Q_\ell)\right\}, \max_p\left\{\lambda^{(p)}(\hat{Q})\right\}\right\} + \epsilon, \tag{2.21}$$

where $\min_p\left\{\lambda^{(p)}\right\}$ is the minimum eigenvalue of the flux Jacobian, $\partial f/\partial q$, and $\hat{Q}$ is the Roe average of states Q_ℓ and Q_r [29]. In the above expression, ϵ is a small parameter that in practice can be taken to be $\epsilon = 10^{-10}$.

In the scalar case, both the HLLE and local-Lax-Friedrichs (LLF) Riemann solvers reduce to the upwind method. That is, at each interface where $f'(q) > 0$, we set $F(Q_\ell, Q_r) = f(Q_\ell)$, and when $f'(q) < 0$, we set $F(Q_\ell, Q_r) = f(Q_r)$. In this work, all equations we deal with are scalar equations, and hence an upwind method is sufficient for our purposes.

This completes a method of lines (MOL) discretization for our PDE. The only remaining part is to evolve the discrete coefficients through time. This is usually performed by explicit, high-order Runge-Kutta methods, resulting in a Runge-Kutta discontinuous Galerkin (RKDG) method.

2.1.2 High-Order Time Stepping

For explicit Runge-Kutta integrators, one needs to obey a Courant–Friedrichs–Lewy [23] (CFL) time step restriction when advancing the unknowns forward in time. For this 1D problem, the CFL number is defined as the dimensionless quantity,

$$\text{CFL} := \max_{1 \leq i \leq m_x+1} \left| \frac{s_{i-1/2} \Delta t}{\Delta x} \right|, \tag{2.22}$$

where $s_{i-1/2}$ refers to the largest eigenvalue (wave speed) of the flux Jacobian present at the interface located at $i - 1/2$. Popular time-integrators include recent, low storage total variation diminishing (TVD) methods [33, 44]. TVD and strong stability preserving (SSP) methods help prevent spurious oscillations from developing in the approximate solution. Roughly speaking, the maximum allowable CFL number scales as $1/(2M-1)$ for an M^{th} order method with a typical Runge-Kutta integrator. Time-stepping is most often handled via total-variation diminishing Runge-Kutta (TVD-RK) methods. To illustrate these methods, consider the initial value problem:

$$\frac{d}{dt} u = \mathcal{L}(u). \tag{2.23}$$

The first order TVD-RK method is simply the forward Euler method:

$$U^{n+1} = U^n + \Delta t \mathcal{L}(U^n). \tag{2.24}$$

The second order accurate version is

$$U^\star = U^n + \Delta t \mathcal{L}(U^n), \tag{2.25}$$

$$U^{n+1} = \frac{1}{2} U^n + \frac{1}{2} U^\star + \frac{1}{2} \Delta t \mathcal{L}(U^\star). \tag{2.26}$$

The third order accurate TVD integrator of Shu-Osher [57] is

$$U^\star = U^n + \Delta t\, \mathcal{L}(U^n), \tag{2.27}$$

$$U^{\star\star} = \frac{3}{4}U^n + \frac{1}{4}U^\star + \frac{1}{4}\Delta t\, \mathcal{L}(U^\star), \tag{2.28}$$

$$U^{n+1} = \frac{1}{3}U^n + \frac{2}{3}U^{\star\star} + \frac{2}{3}\Delta t\, \mathcal{L}(U^{\star\star}). \tag{2.29}$$

2.2 2D Problems on Cartesian Grids

In this section we briefly review the DG method for a general two-dimensional conservation law on a Cartesian mesh. This section will also serve as a continuation of our introduction to notation used throughout the remainder of this dissertation.

Consider a general 2D conservation law of the form:

$$q_{,t} + f(q,t,\mathbf{x})_{,x} + g(q,t,\mathbf{x})_{,y} = 0, \quad \text{in} \quad \mathbf{x} \in \Omega \subset \mathbb{R}^2, \tag{2.30}$$

with appropriate initial and boundary conditions. In this equation $q(t,\mathbf{x}) \in \mathbb{R}^m$ is the vector of conserved variables and $f(q,t,\mathbf{x}), g(q,t,\mathbf{x}) \in \mathbb{R}^m$ are the flux functions in the x and y-directions, respectively. We assume that equation (2.30) is hyperbolic, meaning that the family of $m \times m$ matrices defined by

$$A(q,t,\mathbf{x};\mathbf{n}) = \mathbf{n} \cdot \left(\frac{\partial f}{\partial q}, \frac{\partial g}{\partial q}\right)^T \tag{2.31}$$

are diagonalizable with real eigenvalues for all $\mathbf{x}$ and q in the domain of interest and for all $\|\mathbf{n}\| = 1$.

We construct a Cartesian grid over $\Omega = [a_x, b_x] \times [a_y, b_y]$, with uniform grid spacing Δx and Δy in each coordinate direction. Again, non-uniform meshes may certainly be considered, however we choose to proceed with a uniform description in order to avoid

obfuscating our description of the underlying method. A uniform grid has mesh elements centered at the coordinates

$$x_i = a_x + \left(i - \frac{1}{2}\right)\Delta x \quad \text{and} \quad y_j = a_y + \left(j - \frac{1}{2}\right)\Delta y, \tag{2.32}$$

with $1 \le i \le m_x$ and $1 \le j \le m_y$.

On this grid we define the *broken* finite element space

$$W^h = \left\{ w^h \in L^\infty(\Omega) : \, w^h|_{\mathcal{T}} \in P^q, \, \forall \mathcal{T} \in \mathcal{T}_h \right\}, \tag{2.33}$$

where W^h is shorthand notation for $W^{\Delta x, \Delta y}$. The above expression means that on each element $\mathcal{T}$, w^h will be a polynomial of degree at most q, and no continuity is assumed across element edges. Each element can be mapped to the canonical element $(\xi, \eta) \in [-1, 1] \times [-1, 1]$ via the linear transformation:

$$x = x_i + \xi \, \frac{\Delta x}{2}, \quad y = y_j + \eta \, \frac{\Delta y}{2}. \tag{2.34}$$

The normalized Legendre polynomials up to degree four on the canonical element can be written as

$$\begin{aligned} \varphi^{(\ell)} = \Bigg\{ & 1, \;\; \sqrt{3}\,\xi, \;\; \sqrt{3}\,\eta, \;\; 3\,\xi\eta, \;\; \frac{\sqrt{5}}{2}\left(3\xi^2 - 1\right), \;\; \frac{\sqrt{5}}{2}\left(3\eta^2 - 1\right), \\ & \frac{\sqrt{15}}{2}\eta\,(3\xi^2 - 1), \;\; \frac{\sqrt{15}}{2}\xi\,(3\eta^2 - 1), \;\; \frac{\sqrt{7}}{2}(5\xi^3 - 3\xi), \;\; \frac{\sqrt{7}}{2}(5\eta^3 - 3\eta), \\ & \frac{\sqrt{21}}{2}\eta\,(5\xi^3 - 3\xi), \;\; \frac{\sqrt{21}}{2}\xi\,(5\eta^3 - 3\eta), \;\; \frac{5}{4}(3\xi^2 - 1)(3\eta^2 - 1), \\ & \frac{105}{8}\xi^4 - \frac{45}{4}\xi^2 + \frac{9}{8}, \;\; \frac{105}{8}\eta^4 - \frac{45}{4}\eta^2 + \frac{9}{8} \Bigg\}. \end{aligned}$$

These basis functions are orthonormal with respect to the following inner product:

$$\left\langle \varphi^{(m)}, \, \varphi^{(n)} \right\rangle := \frac{1}{4} \int_{-1}^{1} \int_{-1}^{1} \varphi^{(m)}(\xi, \eta) \, \varphi^{(n)}(\xi, \eta) \, d\xi \, d\eta = \delta_{mn}. \tag{2.35}$$

We will look for approximate solutions of (2.30) that have the following form:

$$q^h(t,\xi,\eta)\Big|_{\mathcal{T}_{ij}} := \sum_{k=1}^{M(M+1)/2} Q_{ij}^{(k)}(t)\,\varphi^{(k)}(\xi,\eta), \tag{2.36}$$

where M is the desired order of accuracy in space. The Legendre coefficients of the initial conditions at $t = 0$ are determined from the L_2-projection of $q^h(0, x, y)$ onto the Legendre basis functions:

$$Q_{ij}^{(k)}(0) := \Big\langle q^h(0,\xi,\eta),\, \varphi^{(k)}(\xi,\eta) \Big\rangle. \tag{2.37}$$

In practice, these double integrals are evaluated using standard 2D Gaussian quadrature rules involving M^2 points. See appendix section B for explicit formulas.

In order to determine the Legendre coefficients for $t > 0$, we multiply conservation law (2.30) by the test function $\varphi^{(\ell)}$ and integrate over the grid cell $\mathcal{T}_{ij}$. After the appropriate integrations-by-part, we arrive at the following semi-discrete evolution equations:

$$\frac{d}{dt}\,Q_{ij}^{(\ell)} = \mathcal{L}_{ij}^{(\ell)}(Q,t) := N_{ij}^{(\ell)} - \frac{\Delta\mathcal{F}_{ij}^{(\ell)}}{\Delta x} - \frac{\Delta\mathcal{G}_{ij}^{(\ell)}}{\Delta y}, \tag{2.38}$$

where the interior integral is given by

$$N_{ij}^{(\ell)} = \frac{1}{2}\int_{-1}^{1}\int_{-1}^{1}\left[\frac{1}{\Delta x}\,\varphi_{,\xi}^{(\ell)}\,f(q^h,t,\mathbf{x}) + \frac{1}{\Delta y}\,\varphi_{,\eta}^{(\ell)}\,g(q^h,t,\mathbf{x})\right]\,d\xi\,d\eta, \tag{2.39}$$

and the boundary terms are given by,

$$\Delta\mathcal{F}_{ij}^{(\ell)} = \left[\frac{1}{2}\int_{-1}^{1}\varphi^{(\ell)}\,f(q^h,t,\mathbf{x})\,d\eta\right]_{\xi=-1}^{\xi=1}, \tag{2.40}$$

$$\Delta\mathcal{G}_{ij}^{(\ell)} = \left[\frac{1}{2}\int_{-1}^{1}\varphi^{(\ell)}\,g(q^h,t,\mathbf{x})\,d\xi\right]_{\eta=-1}^{\eta=1}. \tag{2.41}$$

The integrals in (2.39) can be numerically approximated via standard 2D Gaussian quadrature rules involving $(M-1)^2$ points. The integrals in (2.40) and (2.41) can be

approximated with standard 1D Gauss quadrature rules involving M points. For each of these 1D quadrature points, one needs to solve a 1D Riemman problem, where the left and right states are evaluated by sampling q on the left and right hand side of the integral. Test functions are always evaluated on the interior of the mesh element. Equation 2.38 is again evolved through a method of lines (MOL) formulation via a high-order integrator.

2.3 2D Problems on Unstructured Grids

We also briefly describe how to solve a hyperbolic balance law (2.30) on a polygonal domain Ω with boundary $\partial\Omega$. Let $\mathcal{T}^h$ be a mesh with triangular elements Ω, where h is the longest edge in $\mathcal{T}^h$. Consider an element $\mathcal{T}_i \in \mathcal{T}^h$ with nodes $(x_k,\, y_k)$ for $k = 1, 2, 3$ centered at

$$\bar{x}_i = \frac{1}{3}\left(x_1 + x_2 + x_3\right) \qquad \text{and} \qquad \bar{y}_i = \frac{1}{3}\left(y_1 + y_2 + y_3\right) \tag{2.42}$$

with area

$$|\mathcal{T}_i| := \frac{1}{2}\Big[(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)\Big]. \tag{2.43}$$

Note that we assume that the nodes on each triangle are numbered in a counter-clockwise fashion so that $|\mathcal{T}_i| > 0$. We map this to the canonical element $\mathcal{T}_c$ centered at $(\xi = 0,\, \eta = 0)$ with nodes:

$$(\xi_k,\, \eta_k) = \left\{ \left(-\frac{1}{3},\, -\frac{1}{3}\right), \left(\frac{2}{3},\, -\frac{1}{3}\right), \left(-\frac{1}{3},\, \frac{2}{3}\right) \right\}, \tag{2.44}$$

via the following linear transformation:

$$x\left(\xi, \eta\right) = \bar{x}_i + \xi\left(x_2 - x_1\right) + \eta\left(x_3 - x_1\right), \tag{2.45}$$

$$y\left(\xi, \eta\right) = \bar{y}_i + \xi\left(y_2 - y_1\right) + \eta\left(y_3 - y_1\right). \tag{2.46}$$

Let $\mu^{(k)}(\xi,\eta)$ be the set of monomials defined on the canonical element $\mathcal{T}_c$. For example, up to degree one these monomials are

$$\mu^{(k)}(\xi,\eta) = \{1,\ \ \xi,\ \ \eta\}\,. \tag{2.47}$$

The monomials $\mu^{(k)}(\xi,\eta)$ are converted to an orthonormal basis on $\mathcal{T}_c$ via the Gram-Schmidt process. The result is the basis functions:

$$\varphi^{(\ell)}(\xi,\eta) := \sum_{k=1}^{\ell} \Theta_{\ell k}\ \mu^{(k)}(\xi,\eta), \tag{2.48}$$

where, again in the example of linear polynomials,

$$\Theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{18} & 0 \\ 0 & \sqrt{6} & \sqrt{24} \end{bmatrix}. \tag{2.49}$$

These basis functions are orthonormal with respect to the following inner product:

$$\left\langle \varphi^{(m)},\, \varphi^{(n)} \right\rangle := 2\int_{-\frac{1}{3}}^{\frac{2}{3}} \int_{-\frac{1}{3}}^{\frac{1}{3}-\xi} \varphi^{(m)}(\xi,\eta)\,\varphi^{(n)}(\xi,\eta)\,d\eta\,d\xi = \delta_{mn}. \tag{2.50}$$

Once we have established the basis functions, we proceed as in the Cartesian case. We look for approximate solutions of (2.30) that have the following form:

$$q^h\Big(t, x\,(\xi,\eta)\,, y\,(\xi,\eta)\Big)\Big|_{\mathcal{T}_i} := \sum_{k=1}^{M(M+1)/2} Q_i^{(k)}(t)\,\varphi^{(k)}(\xi,\eta), \tag{2.51}$$

where M is the desired order of accuracy in space. After multiplication by test functions and integrations-by-part, we end up with a semi-discrete system of the form:

$$\frac{d}{dt}\,Q_i^{(\ell)} = N_i^{(\ell)} - \sum_{e=1}^{3} F_{e\,i}^{(\ell)}, \tag{2.52}$$

where $N_i^{(\ell)}$ is again the contribution from the element interior and $F_{e\,i}^{(\ell)}$ are the numerical flux contributions from the element boundary.

Chapter 3

Semi-Lagrangian Methods for 1+1 Vlasov-Poisson

In this chapter, we describe a semi-Lagrangian discontinuous Galerkin (SLDG) method for the (1+1)D Vlasov-Poisson system. Our novel SLDG method simultaneously accomplishes all of the following:

1. Unconditionally stable;
2. Mass conservative;
3. Positivity-preserving;
4. 4^{th} order accurate in time;
5. 5^{th} order accurate in space.

Unconditional stability is attained by turning to semi-Lagrangian methods. High-order spatial accuracy is accomplished by appealing to a DG representation for the solution. High-order time accuracy is accomplished through high-order split methods. Mass conservation is essentially automatic because DG methods are a type of finite volume method. In addition, a recent positivity preserving limiter [66] is modified to accommodate our method.

3.1 Introduction

Our method uses, as a starting point, the method developed by Cheng and Knorr for solving the Vlasov equations. Cheng and Knorr [12] developed a second order accurate scheme for Vlasov Poisson via Strang operator splitting [60]. Operator split methods are described in §3.3.3, and the scheme developed by Cheng and Knorr is summarized in Algorithm 3.1. Our result uses Cheng and Knorr's method as its starting point. We add in a high-order spatial representation via a DG framework, where each split direction is described in §3.3. We also demonstrate that it is possible to achieve high-order accuracy via high-order splitting methods, which are described in §3.3.3. These high-order split methods necessitate a high-order method for expanding the electric field, which is described in §3.4.2. In addition, we adapt a high-order positivity preserving limiter, presented in section §3.3.4, to fit our method.

3.2 Cheng and Knorr Splitting

Cheng and Knorr realized that if we momentarily freeze the electric field in time, the Vlasov equation (1.19) can be viewed as an advection equation of the following form:

$$f_{,t} + \mathbf{a}(\mathbf{v}) \cdot f_{,\mathbf{x}} + \mathbf{b}(\mathbf{x}) \cdot f_{,\mathbf{v}} = 0. \tag{3.1}$$

This equation can be handled very efficiently if split into the following two sub-problems:

$$\text{Problem } \mathcal{A}: \quad f_{,t} + \mathbf{a}(\mathbf{v}) \cdot f_{,\mathbf{x}} = 0,$$

$$\text{Problem } \mathcal{B}: \quad f_{,t} + \mathbf{b}(\mathbf{x}) \cdot f_{,\mathbf{v}} = 0.$$

The key benefit of this splitting is that each operator is now a constant coefficient advection equation (i.e., the transverse coordinate acts only as a parameter), each of

which can be handled very simply with a variety of spatial discretization and semi-Lagrangian time-stepping. The down side of this approach, of course, is the introduction of splitting errors.

It is worth pointing out that the electric field computed in Step 2, $\mathbf{E}^{n+\frac{1}{2}}$, is second order accurate in time, even though it is computed after advection in the $\mathbf{x}$ variables only. This fact is often left out of papers that use this splitting scheme, which we now prove.

Algorithm 3.1 Cheng and Knorr [12] operator split algorithm.

1. $\frac{1}{2}\Delta t$ step on $f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} = 0.$
2. Solve $-\nabla^2\phi = \rho^{n+\frac{1}{2}} - \rho_0,$ and compute $\mathbf{E}^{n+\frac{1}{2}} = -\nabla\phi.$
3. Δt step on $f_{,t} + \mathbf{E}^{n+\frac{1}{2}} \cdot f_{,\mathbf{v}} = 0.$
4. $\frac{1}{2}\Delta t$ step on $f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} = 0.$

Claim. *Assuming that the current solution at time $t = t^n$ is known exactly, and that each step in Algorithm 3.1 is carried out exactly in space, velocity, and time, the density computed in Step 2 is second order accurate in time:*

$$\rho^{n+\frac{1}{2}} = \rho\left(t^n + \frac{\Delta t}{2}, \mathbf{x}\right) + \mathcal{O}\left(\Delta t^2\right).$$

This also implies that the electric field in Step 2 is second order accurate in time:

$$\mathbf{E}^{n+\frac{1}{2}} = \mathbf{E}\left(t^n + \frac{\Delta t}{2}, \mathbf{x}\right) + \mathcal{O}\left(\Delta t^2\right).$$

Proof. By assumption the PDF after the first step satisfies the following relationship:

$$\tilde{f}\left(\mathbf{x}, \mathbf{v}\right) := f\left(t^n, \mathbf{x} - \frac{\Delta t}{2}\mathbf{v}, \mathbf{v}\right).$$

We integrate this relationship in velocity to compute the density at time $t^n + \frac{\Delta t}{2}$:

$$\begin{aligned}\rho^{n+\frac{1}{2}} &:= \int_{\mathbf{v}} \tilde{f}\left(\mathbf{x},\mathbf{v}\right) d\mathbf{v} = \int_{\mathbf{v}} f\left(t^n, \mathbf{x} - \frac{\Delta t}{2}\mathbf{v}, \mathbf{v}\right) d\mathbf{v} \\ &= \int_{\mathbf{v}} f\left(t^n,\mathbf{x},\mathbf{v}\right) d\mathbf{v} - \frac{\Delta t}{2}\nabla_{\mathbf{x}} \cdot \left\{ \int_{\mathbf{v}} \mathbf{v} f\left(t^n,\mathbf{x},\mathbf{v}\right) d\mathbf{v} \right\} + \mathcal{O}(\Delta t^2) \\ &= \rho^n - \frac{\Delta t}{2}\nabla_{\mathbf{x}} \cdot \left(\rho^n \mathbf{u}^n\right) + \mathcal{O}(\Delta t^2).\end{aligned}$$

Finally, we use the fact that

$$\rho^n_{,t} = -\nabla_{\mathbf{x}} \left(\rho^n \mathbf{u}^n\right),$$

in order to assert that

$$\rho^{n+\frac{1}{2}} = \rho^n + \frac{\Delta t}{2}\rho^n_{,t} + \mathcal{O}(\Delta t^2) = \rho\left(t + \frac{\Delta t}{2}, \mathbf{x}\right) + \mathcal{O}(\Delta t^2),$$

which proves the claim. □

3.3 SLDG Schemes for the Advection Equation

At the heart of a semi-Lagrangian solver for the Vlasov-Poisson system lies a method of solving a variable coefficient advection equation of the form,

$$f_{,t} + a(v) f_{,x} + b(t,x) f_{,v} = 0. \tag{3.2}$$

The correct building blocks for a full solver include a 1D constant coefficient solver, a quasi-1D solver, splitting methods to glue the problems together, and finally, a method for accommodating time dependence. Beyond that, Poisson solvers need to be developed, and a method for evaluating the field, $b(t,x)$ needs to be added. But first, a simpler problem lies in creating a reliable solver for the 1D constant coefficient problem, to which we presently turn.

3.3.1 1D Constant Coefficient Problem

The 1D constant coefficient advection equation is given by

$$f_{,t} + u f_{,x} = 0; \quad f(0,x) = f_0(x). \tag{3.3}$$

The domain which we solve this problem on is $(t,x) \in (\mathbb{R}^+, \mathbb{R})$. The initial condition is prescribed by $f(0,x) = f_0(x)$, which is a known function. The exact, analytic solution to this problem is simply $f(t,x) = f_0(x-ut)$. For simplicity of exposition, we assume that $u > 0$; the extension to the case $u < 0$ is straightforward.

A simple, high-order accurate, and unconditionally stable algorithm to update this solution can be developed based on the following two steps:

1. Exactly advect the initial condition over a time step Δt:

$$f\left(t + \Delta t, x\right) = f\left(t, x - u\Delta t\right)$$

2. Project this solution back onto the mesh $\mathcal{T}_i$.

This process is illustrated in Figure 5. Given a starting time t^n and final time $t^{n+1} = t^n + \Delta t$, the unknowns are defined through:

$$\begin{aligned} F_i^{(\ell)}\left(t^{n+1}\right) &= \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \varphi_{\mathrm{1D}}^{(\ell)}(\xi) f(t^{n+1}, x)\; dx \\ &= \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \varphi_{\mathrm{1D}}^{(\ell)}(\xi) f(t^{n}, x - u\Delta t)\; dx. \end{aligned} \tag{3.4}$$

In order to evaluate (3.4), we split the integral up into two parts. The numerical update is then defined by:

$$\begin{aligned} F_i^{(\ell)} = &\frac{1}{2}\sum_{k=1}^{M} F_{i-1-j}^{(k)}(t^n) \int_{-1}^{-1+2\nu} \varphi_{\mathrm{1D}}^{(k)}(\xi + 2 - 2\nu)\, \varphi_{\mathrm{1D}}^{(\ell)}(\xi)\, d\xi \\ &+ \frac{1}{2}\sum_{k=1}^{M} F_{i-j}^{(k)}(t^n) \int_{-1+2\nu}^{1} \varphi_{\mathrm{1D}}^{(k)}(\xi - 2\nu)\, \varphi_{\mathrm{1D}}^{(\ell)}(\xi)\, d\xi, \end{aligned} \tag{3.5}$$

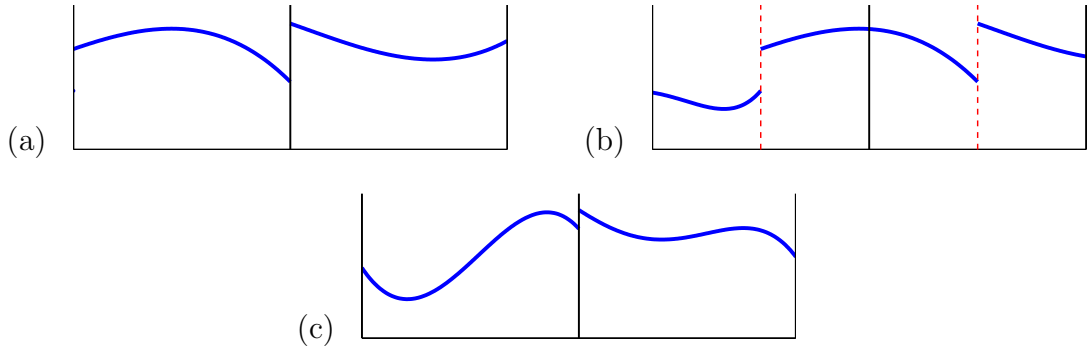


Figure 5: Illustration of the shift + project method for solving the constant coefficient advection equation in 1D as described in §3.3.1. Panel (a) shows piecewise polynomial initial data; Panel (b) shows the initial data shifted by some amount (i.e., the exact evolution of the initial data); and finally, Panel (c) shows the solution after it has been re-projected back onto the original piecewise polynomial basis.

where

$$j := \left\lfloor \frac{u\Delta t}{\Delta x} \right\rfloor \qquad \text{and} \qquad \nu := \frac{u\Delta t}{\Delta x} - j. \tag{3.6}$$

Here $\lfloor \cdot \rfloor$ denotes the *floor* operation[1] and $0 \leq \nu < 1$. By construction, update (3.5) is *unconditionally stable* independent of the polynomial order of the spatial discretization.

The integrals in equation (3.5) can be evaluated exactly. For example, in the case of piecewise constants and $j = 0$, (3.5) is nothing more than the first-order upwind scheme:

$$F_i^{(1),n+1} = F_i^{(1),n} - \nu \left(F_i^{(1),n} - F_{i-1}^{(1),n} \right). \tag{3.7}$$

In the case of piecewise linear polynomials and $j = 0$, the scheme can be written as

[1]This function takes a real input and rounds down to the largest integer that is smaller than or equal to the input.

follows:

$$\begin{aligned} F_i^{(1),n+1} = F_i^{(1),n} - \nu \left(\left[F_i^{(1),n} + \sqrt{3}\, F_i^{(2),n} \right] - \left[F_{i-1}^{(1),n} + \sqrt{3}\, F_{i-1}^{(2),n} \right] \right) \\ + \sqrt{3}\, \nu^2 \left(F_i^{(2),n} - F_{i-1}^{(2),n} \right), \end{aligned} \tag{3.8}$$

$$\begin{aligned} F_i^{(2),n+1} = F_i^{(2),n} + \sqrt{3}\, \nu \left(\left[F_i^{(1),n} - \sqrt{3}\, F_i^{(2),n} \right] - \left[F_{i-1}^{(1),n} + \sqrt{3}\, F_{i-1}^{(2),n} \right] \right) \\ - \sqrt{3}\, \nu^2 \left(F_i^{(1),n} - F_{i-1}^{(1),n} - 2\sqrt{3}\, F_{i-1}^{(2),n} \right) + 2\nu^3 \left(F_i^{(2),n} - F_{i-1}^{(2),n} \right). \end{aligned} \tag{3.9}$$

In order to compute the integrals presented in (3.5), one needs to know $u, \Delta x$ and Δt, because these determine how far information has shifted, as well as the location of the discontinuity, ν. In the semi-Lagrangian literature, one often refers to a method as either a forward, or a backward semi-Lagrangian method. We prefer to think of this as both a forward, as well as a backward method, in the following sense: discontinuities are propagated forwards in time, and function values are retraced backwards in time. In order to formulate a quadrature rule which will integrate (3.5) exactly, the following two steps need to be enacted:

1. *Forward:* Push the discontinuities in the solution forward in time, in order to determine where the discontinuities lie for the projection step lie. After doing this step, one may lay down a list of quadrature points. For an M^{th} order method, this requires M points on the left half, ξ^L, and M on the right half, ξ^R with appropriate weights, ω^L and ω^R.

2. *Backward:* After the quadrature points are known, solution values at each point $f(t^{n+1}, \xi^{L,R})$ can be determined by tracing characteristics backwards in time.

This process is illustrated in Figure 6.

We now describe exactly how the integrals presented in (3.5) can be exactly evaluated using Gaussian quadrature. For ease of presentation, we will restrict ourselves to

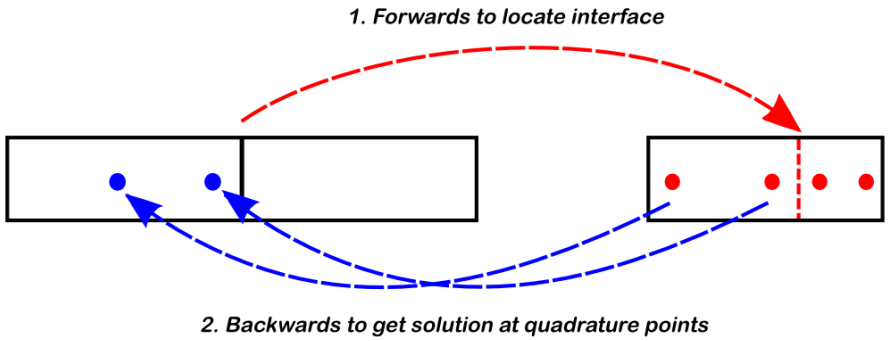


Figure 6: Illustration of the forward and backward nature of the proposed semi-Lagrangian scheme. First, the cell edges are propagated *forward* from their initial time to their final time. Once these locations are known, Gauss-Legendre quadrature points are placed between the old cell edges and the new cell edges. In order to find solution values at these Gauss-Legendre points, we trace *backwards* along the characteristics to the initial time.

describing the computation of:

$$I_l := \int_{-1}^{-1+2\nu} \varphi(\xi) f(t^{n+1}, x(\xi))\, d\xi \quad \text{and} \quad I_r := \int_{-1+2\nu}^{1} \varphi(\xi) f(t^{n+1}, x(\xi))\, d\xi. \tag{3.10}$$

Let $\{\omega_1, \omega_2, \ldots, \omega_M\}$ denote M 1D quadrature weights, together with their associated quadrature points, $\{\xi_1, \xi_2, \ldots, \xi_M\} \subset [-1, 1]$. Both the left interval $[-1, -1+2\nu]$ and right intervals, $[-1+2\nu, 1]$ can be transformed into the interval [-1,1] by defining left and right quadrature points as:

$$\xi_m^L = \nu\xi_m + (-1+\nu); \quad \xi_m^R = (1-\nu)\,\xi_m + \nu \tag{3.11}$$

together with their associated quadrature points:

$$\omega_m^L = \nu\omega_m; \quad \omega_m^R = (1-\nu)\,\omega_m. \tag{3.12}$$

We note that in the case of $\nu = 0$, the left hand integral vanishes, and all left hand quadrature points are at $\xi = -1$. In the case where $\nu = 1$, i.e. a CFL number of 1,

the right hand integral vanishes, and each right hand quadrature point gets mapped to $\xi = 1$. The single update formula, with exact integration becomes,

$$F_i^{(k),n+1} = \sum_{m=1}^{M} \omega_m^L \varphi^{(k)}(\xi_m^L) f(t^{n+1}, x(\xi_m^L)) + \omega_m^R \varphi(\xi_m^R) f(t^{n+1}, x(\xi_m^R)). \tag{3.13}$$

Function values are evaluated by tracing quadrature points back in time:

$$f(t^{n+1}, x(\xi_m^R)) = f(t^n, x(\xi_m^R) - u\Delta t) = \tilde{f}_{i-j}^h(t^n, \xi_m^R - 2\nu), \tag{3.14}$$

$$f(t^{n+1}, x(\xi_m^L)) = f(t^n, x(\xi_m^L) - u\Delta t) = \tilde{f}_{i-j-1}^h(t^n, \xi_m^L + 1(1-\nu)). \tag{3.15}$$

Here, we are using the notation $\tilde{f}^h(t, \xi) := f^h(t, x)|_{\mathcal{T}_i}$. Each of these values can be computed by evaluating the basis functions with appropriate weights:

$$\tilde{f}_i^h(t^n, \xi) = \sum_{k=1}^{M} F_i^{(k),n} \varphi^{(k)}(\xi). \tag{3.16}$$

The index j and the CFL number ν refer to the index shift and how far information has traveled; these two quantities have already been defined in (3.6). In terms of each canonical grid cell, the location of the discontinuity for the **new** cell is $\xi = -1 + 2\nu$. When we plug in $\xi^L = -1 + 2\nu$, the location of the **old** ξ is then $\xi = +1$, which is expected because we are supposed to be evaluating f at the right hand endpoint of a grid cell. When we plug in $\xi^R = -1 + 2\nu$, the location of the **old** ξ is $\xi = -1$ which is also expected, because we're supposed to be evaluating f at the left half of a grid cell.

Time Dependent Velocities

We will shortly describe how this extends to a 2D problem, but first we would like to describe how one can hand a time dependent velocity field. The case where the velocity field, $u = u(t)$ also depends on time can also be solved to high-order, using essentially

the same scheme. The key to high-order for this solver rests on being able to accurately and efficiently retrace quadrature points forward and backwards in time. For example, the exact solution to

$$f_{,t} + u(t) f_{,x} = 0, \quad f(0,x) = f_0(x), \tag{3.17}$$

is no longer $f(t,x) = f_0(x - u(t)t)$, but rather $f(t,x) = f_0\left(x - \int_0^t u(s)\ ds\right)$. In the case where $u(t) = u$ is constant, this reproduces the exact solution from earlier. In practice, code written for the constant coefficient case can be recycled, provided the correct definitions are placed. If we set $\bar{u} := \frac{1}{\Delta t}\int_0^{\Delta t} u(s)\ ds$, and take a single time step using code written to handle (3.3), with $\bar{u}$ in place of u, then this code will exactly trace quadrature points backwards and forwards in time to their correct location. Of course, we have delayed describing exactly how the velocity integral, $\int u(s)\ ds$ is computed, but one may choose to apply high-order quadrature if intermediate velocity values can be computed, or in the case of a known function, this may be evaluated exactly. We now turn towards describing how this method can be utilized to build a high-order, 2D advection solver.

3.3.2 Quasi 1D Advection Equation

Extension of the exact integration formulas from the 1D problem to a 2D advection equation is not necessarily straightforward with a DG spatial representation, and one of the novel ideas presented by Rossmanith and Seal [53] is the spatial discretization for the semi-Lagrangian problem. One difficulty encountered with a DG representation that is different than that of Christlieb and Qiu [49] is that the basis functions also depend on the transverse direction.

The next building block is a simple quasi-1D solver. This will create the basis for a fully working 2D solver. Here, we define our 'quasi-1D' problem as:

$$f_{,t} + u(v) f_{,x} = 0; \quad f(0, x, v) = f_0(x, v), \tag{3.18}$$

and the domain we solve this on is $(t, x, v) \in \mathbb{R}^+ \times [a, b] \times \mathbb{R}$. In a DG representation, the basis functions live on grid elements which depend on the variable being advected, x, as well as the transverse variable, v, which defines the velocity field. As time is advanced, each rectangular grid element will undergo shearing. This stands in stark contrast to the 1D problem, where grid cells experienced rigid translation. In the case of $u(v) = v$, this is a linear shear, but for a more generic velocity field, this shear can take on a much more complicated structure. We present our complete algorithm in Table 3.2, to which we presently direct the reader. Observe that this algorithm accommodates generic shearing for the velocity fields.

We now prove that the quasi-1D update (Algorithm 3.2) is mass conservative.

Theorem 1. *The update for the quasi-1D problem is numerically mass conservative.*

Proof. Let $\mathcal{T}_{ij} = [x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{j+1/2}]$ denote a 2D mesh element, and let $C_i = [x_{i-1/2}, x_{i+1/2}]$ denote the corresponding 1D grid cell. Suppose $\{v_1, v_2, ...v_M\}$ and $\{\omega_1, \omega_2, ...\omega_M\}$ denote the 1D quadrature weights for evolving each row in the update. For a fixed time t, the total mass in the j^{th} row is given by

$$\int_{v_{j-1/2}}^{v_{j+1/2}} \int_{-\infty}^{\infty} f(t, x, v)\ dx\ dv = \sum_i \iint_{\mathcal{T}_{ij}} f(t, x, v)\ dx\ dv = \Delta x \Delta v \sum_i F_{ij}^{(1)}(t). \tag{3.22}$$

The last equality follows by orthogonality of the basis functions. After dividing this by

Algorithm 3.2 Proposed method for solving quasi-1D problem (3.18).

0. Initial Projection: Start with the Galerkin representation of the solution. When restricted to a single cell $\mathcal{T}_{ij}$, this is is simply

$$f^h(t,x,v)|_{\mathcal{T}_{ij}} = \sum_{k=1}^{M(M+1)/2} F_{ij}^{(k)} \varphi^{(k)}(\xi,\eta),$$

where the coefficients $F_{ij}^{(k)}$ are given by the projection provided in (2.37).

1. Convert to 1D Problems: For each row j, consider M lines defined by M quadrature points $\{v_{j1}, v_{j2}, \ldots, v_{jM}\}$. For each of these lines, project 2D Legendre moments onto 1D moments via

$$F_{\mathrm{1D},ijk}^{(l)}(t^n) := \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \varphi_{1D}^{(l)}(\xi(x))\, f^h(t^n, x, v_{jk})\, dx. \tag{3.19}$$

Each 1D integral can be evaluated exactly using M quadrature points because f^h is a polynomial of degree at most $M-1$ when restricted to a single line, and $\phi_{1D}^{(k)}$ is a polynomial of degree at most $M-1$. (C.f. Figure 7 for a visual illustration for $M=2$).

2. Evolve 1D Problems: For each line, $k = 1, 2, \ldots, M$, take a step of length Δt on $f_{,t} + u(v_{jk}) f_{,x} = 0$, using the 1D method and produce $\tilde{f}_{i,jk}^*$ whose coefficients are given by $F_{\mathrm{1D},ijk}^{(l)}(t^{n+1})$. In the case of VP, $u(v) = v$, however this algorithm is intentionally written to accommodate generic shearing. Note that there are a total of $m_v \cdot M$ such lines to advance, because each row j produces a total of M lines to advance.

3. Integrate 1D Problems: Integrate the 1D coefficients up to 2D coefficients. This is accomplished through a tensor product of M^2 quadrature points:

$$\begin{aligned} F_{ij}^{(\ell)}(t^{n+1}) &= \sum_{k=1}^{M} \frac{\omega_k}{2} \int_{-1}^{1} \varphi^{(\ell)}(\xi,\eta_k) \tilde{f}_{i,jk}^*(\xi)\, d\xi & (3.20) \\ &= \sum_{l=1}^{M} \sum_{k,m=1}^{M} \frac{\omega_k \omega_m}{4} F_{\mathrm{1D},ijk}^{(l)}(t^{n+1}) \varphi^{(\ell)}(\xi_m,\eta_k) \phi_{\mathrm{1D}}^{(l)}(\xi_m) & (3.21) \end{aligned}$$

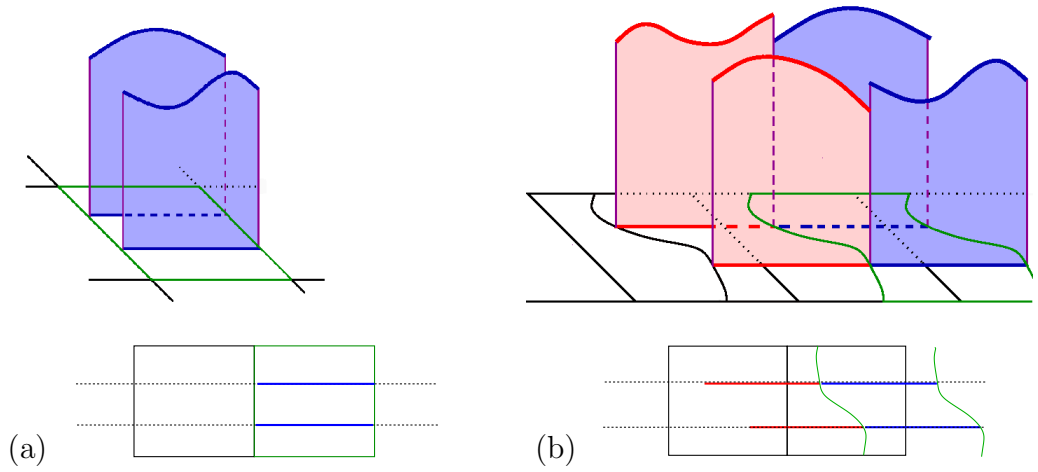


Figure 7: Illustration of the shift + project method for solving the quasi-1D variable coefficient advection equation in 2D as described in §3.3.2 . Panel (a) shows a depiction of initial conditions using $M = 2$ slices of the solution. Each slice is represented via 1D polynomial moments. The top picture represents each slice, and the bottom picture represents a top down view of the solution. Panel (b) shows the initial data shifted by some amount (i.e., the exact evolution of the initial data); note that each equation receives a different velocity speed, and hence the edge of each cell is distorted by some amount. We note that this is a cartoon depiction of a shift where the CFL number is less than one, but there is no reason that information can't shift farther than that. C.f. Figure 5 for the analogous 1D pictures; here we have exactly 2 of these problems to solve. The next step in the algorithm is the projection step, back onto 1D basis function followed by finally integrating the 1D problems back up to the 2D basis functions.

$\Delta x \Delta v$, total mass after a single update is then

$$\begin{aligned}
\sum_i F_{ij}^{(1)}(t^{n+1}) &= \sum_i \frac{1}{\Delta x \Delta v} \iint_{\mathcal{T}_{ij}} f^h(t^{n+1}, x, v)\ dx\ dv && (3.23)\\
&= \sum_{i,m} \frac{\omega_m}{\Delta x} \int_{C_i} f^h(t^{n+1}, x, v_m)\ dx && (3.24)\\
&= \sum_{i,m} \frac{\omega_m}{\Delta x} \int_{C_i} f^h(t^{n}, x, v)\ dx && (3.25)\\
&= \sum_i \frac{1}{\Delta x \Delta v} \iint_{\mathcal{T}_{ij}} f^h(t^{n}, x, v)\ dx\ dv && (3.26)\\
&= \sum_i F_{ij}^{(1),n}. && (3.27)
\end{aligned}$$

Line (3.24) follows because integrating $\varphi(x, v_m)$ along a fixed row specified by y_m is exact when using enough quadrature points. Line (3.25) follows because the 1D method with a 1D update conserves mass. Lines (3.26) and (3.27) follow for the same reasons the first two lines are true. That is, the method exactly integrates the sum of each row, and the basis functions are orthogonal. If we sum (3.23) over all rows j, we obtain numerical mass conservation, that is,

$$\iint_{x,v\in\mathbb{R}} f^h(t^{n+1}, x, v)\ dx\ dv = \iint_{x,v\in\mathbb{R}} f^h(t^{n}, x, v)\ dx\ dv. \tag{3.28}$$

□

Now that we have a method for updating a quasi-1D problem, we can now solve two equations:

$$f_{,t} + a(v) f_{,x} = 0; \quad f_{,t} + b(x) f_{,v} = 0. \tag{3.29}$$

Obviously, the roles of x and v need to be swapped in algorithm 3.2, and the projection step onto 1D problems is replaced by swapping the roles of ξ and η. Splitting methods allow us to glue these two problems together.

3.3.3 Operator Split Methods

One of the goals of this work is to demonstrate that *high-order* splitting methods can be applied to the Vlasov-Poisson system. Before presenting high-order split methods, we first provide motivation for where these methods come from by reviewing simple, low order split methods.

Consider a time-dependent problem where the right-hand side is written as the sum of two differential operators $\mathcal{A}$ and $\mathcal{B}$:

$$q_{,t} = \mathcal{A}(q) + \mathcal{B}(q). \tag{3.30}$$

We note that the advection equation, (3.2) is one such example, with $\mathcal{A}(q) = -a(v)f_{,x}$ and $\mathcal{B}(q) = -b(t,x)f_{,v}$. In order to analyze errors incurred in various splitting methods, it will be useful to define the *commutator* of two operators as $[\mathcal{A},\mathcal{B}] = \mathcal{A}\mathcal{B} - \mathcal{B}\mathcal{A}$ which is zero if and only if the operators commute.

Low Order Operator Splitting

The most basic splitting methods is that of Lie-Trotter [61], which is a two stage method, and is first order accurate:

$$\begin{aligned} &\textbf{Stage 1:} \quad \Delta t \quad \text{step on} \quad q_{,t} = \mathcal{A}\left(q\right), \\ &\textbf{Stage 2:} \quad \Delta t \quad \text{step on} \quad q_{,t} = \mathcal{B}(q). \end{aligned}$$

In the case where $[\mathcal{A},\mathcal{B}] = 0$, then these operators commute and this method will produce the exact solution, provided that each stage is integrated exactly. However, when the operators do not commute, a splitting error is introduced.

The idea behind high-order splitting can be illustrated through analyzing where the low order split methods come from. In the case where $\mathcal{A}$ and $\mathcal{B}$ are linear operators, then the exact solution to (3.30) is simply $q(t) = e^{(\mathcal{A}+\mathcal{B})t}q(0)$. Here, the exponential operator is defined through its Taylor expansion:

$$e^{\mathcal{A}t} := \sum_{n=0}^{\infty} \frac{\mathcal{A}^n t^n}{n!}; \qquad \mathcal{A}^n = \underbrace{\mathcal{A} \circ \mathcal{A} \circ \cdots \circ \mathcal{A}}_{n\text{-times}}.$$

When one compares the series for $e^{(\mathcal{A}+\mathcal{B})t}$ to that of the product of $e^{\mathcal{A}t}$ and $e^{\mathcal{B}t}$, we see that

$$e^{(\mathcal{A}+\mathcal{B})\Delta t} = I + (\mathcal{A}+\mathcal{B})\,\Delta t + (\mathcal{A}+\mathcal{B})^2\,\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right) \tag{3.31}$$

$$= I + (\mathcal{A}+\mathcal{B})\,\Delta t + \left(\mathcal{A}^2 + \mathcal{A}\mathcal{B} + \mathcal{B}\mathcal{A} + \mathcal{B}^2\right)\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right). \tag{3.32}$$

Likewise, the product of $e^{\mathcal{A}\Delta t}$ and $e^{\mathcal{B}\Delta t}$ becomes:

$$e^{\mathcal{A}\Delta t}e^{\mathcal{B}\Delta t} = \left(I + \mathcal{A}\Delta t + \mathcal{A}^2\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right)\right)\left(I + \mathcal{B}\Delta t + \mathcal{B}^2\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right)\right) \tag{3.33}$$

$$= I + (\mathcal{A}+\mathcal{B})\,\Delta t + \left(\mathcal{A}^2 + 2\mathcal{A}\mathcal{B} + \mathcal{B}^2\right)\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right). \tag{3.34}$$

If we subtract these two, we see that

$$e^{\mathcal{A}\Delta t}e^{\mathcal{B}\Delta t} - e^{(\mathcal{A}+\mathcal{B})\Delta t} = (\mathcal{A}\mathcal{B} - \mathcal{B}\mathcal{A})\,\frac{\Delta t^2}{2!} + \mathcal{O}\left(\Delta t^3\right). \tag{3.35}$$

By combining products of the two exponentials, we were able to knock out the Δt term, and end up with a first-order accurate method. The error incurred after a single time step is dominated by $[\mathcal{A},\mathcal{B}]\Delta t^2/2$. In order to cancel the $\mathcal{B}\mathcal{A}$ term, one needs to have a $\mathcal{B}$ on the left side of an $\mathcal{A}$. Through clever manipulation of time steps $\{s_1, s_2, \ldots, s_n\}$, it's easy to envision constructing high-order split methods of the form:

$$e^{(\mathcal{A}+\mathcal{B})\Delta t} = e^{\mathcal{A}s_1\Delta t}e^{\mathcal{B}s_2\Delta t}e^{\mathcal{A}s_3\Delta t}\cdots e^{\mathcal{A}s_n\Delta t} + \mathcal{O}(\Delta t^{M+1}). \tag{3.36}$$

One such method is known as Strang splitting [60], with $s_1 = 1/2$, $s_2 = 1$ and $s_3 = 1/2$. This method is a second order and requires three stages:

$$\begin{aligned}
&\textbf{Stage 1:} \quad \frac{\Delta t}{2} \quad \text{step on} \quad q_{,t} = \mathcal{A}\left(q\right), \\
&\textbf{Stage 2:} \quad \Delta t \quad \text{step on} \quad q_{,t} = \mathcal{B}(q), \\
&\textbf{Stage 3:} \quad \frac{\Delta t}{2} \quad \text{step on} \quad q_{,t} = \mathcal{A}\left(q\right).
\end{aligned}$$

High-order Operator Splitting

A fourth-order accurate operator splitting technique for such systems was developed by Forest and Ruth [31] and by Yoshida [63, 64]. If we define the following two constants:

$$\gamma_1 = \frac{1}{2 - 2^{1/3}} \approx 1.351207191959658, \tag{3.37}$$

$$\gamma_2 = -\frac{2^{1/3}}{2 - 2^{1/3}} \approx -1.702414383919315, \tag{3.38}$$

then the fourth-order splitting approach of [31, 63, 64] can be written as a composition of the following seven stages:

$$\begin{aligned}
&\textbf{Stage 1:} \quad \frac{\gamma_1 \Delta t}{2} \approx 0.6756\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{A}\left(q\right), \\
&\textbf{Stage 2:} \quad \gamma_1 \Delta t \approx 1.3512\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{B}(q), \\
&\textbf{Stage 3:} \quad \frac{(\gamma_1 + \gamma_2)\Delta t}{2} \approx -0.1756\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{A}(q), \\
&\textbf{Stage 4:} \quad \gamma_2 \Delta t \approx -1.7024\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{B}(q), \\
&\textbf{Stage 5:} \quad \frac{(\gamma_1 + \gamma_2)\Delta t}{2} \approx -0.1756\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{A}(q), \\
&\textbf{Stage 6:} \quad \gamma_1 \Delta t \approx 1.3512\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{B}(q), \\
&\textbf{Stage 7:} \quad \frac{\gamma_1 \Delta t}{2} \approx 0.6756\,\Delta t \quad \text{step on} \quad q_{,t} = \mathcal{A}\left(q\right).
\end{aligned}$$

$s_1,\ s_{13}$	$\approx$	0.0829844064174052
$s_2,\ s_{12}$	$\approx$	0.245298957184271
$s_3,\ s_{11}$	$\approx$	0.396309801498368
$s_4,\ s_{10}$	$\approx$	0.604872665711080
$s_5,\ s_9$	$\approx$	-0.0390563049223486
$s_6,\ s_8$	$\approx$	-0.35017162289535098
s_7	$\approx$	0.11952419401315084

Table 1: Coefficients for a high-order Runge-Kutta-Nyström operator split method. Each s_i refers to a multiple of Δt from equation (3.36). The odd s's are the stages required for operator $\mathcal{A}$, and the even s's are the stages required for operator $\mathcal{B}$. Note that there are more stages here, but the negative time steps are smaller than those of the Yoshida split method. We note that stages 6, 7 and 8 are explicitly defined by $s_6 = \frac{1}{2} - (s_2 + s_4)$ and $s_7 = 1 - 2\,(s_1 + s_3 + s_5)$.

We note that this splitting approach requires some steps larger than Δt: **Step 2** and **Step 6**; as well as backward steps: **Step 3**, **Step 4**, and **Step 5**.

Other variants of high-order splitting methods exist; often they involve smaller time-steps on sub-stages at the expense of adding more stages. One such method attributed to Blanes and Moan [11], and recently investigated by Crouseilles [27] for the Vlasov-Poisson equation, is a 13 stage method whose coefficients for equation (3.36) are presented in Table 1. By and large, these high order split methods have seen little attention from the hyperbolic PDE community because of the negative time steps involved. In fact, there are provably no methods with order larger than 2 which contain no negative time steps [56]. This can be seen by observing the order conditions necessary for constructing high-order split methods.

3.3.4 Positivity-preserving limiter

In general, high-order methods do not guarantee that the approximate solution remains positive, even though the exact solution is positive. The culprit for this violation is the higher-order moments. That is, even if $F_{ij}^{(1)} \geq 0$, the second moment $F_{ij}^{(2)}$ can allow the solution to reach negative values. After evolution, these negative values can poison the cell average, and one may observe negative values. For our purposes, we attain positivity in the mean via two steps:

1. Each advection-projection step maintains positivity in the mean, provided that the solution is positive at a finite set of select points; and

2. A mechanism needs to be in place which guarantees that the solution is positive at these points. In our application, this mechanism may be applied as a 'pre-processing' step, and we are also guaranteed that this doesn't destroy order of accuracy.

The following theorem guarantees that the solution retains positivity in the mean, and it also tells us exactly which collection of points need to be positive.

Theorem 2 (Positivity in the mean)**.** *Let M denote the spatial order of accuracy and let*

$$K := \left\lceil \frac{M}{2} \right\rceil, \tag{3.39}$$

where $\lceil \cdot \rceil$ denotes the ceiling operation[2]*. Let $f^h(t^n, x, v)$ be a function defined on the broken finite element space* (2.33) *with $q = M - 1$, and let*

$$\tilde{f}_{ij}^h(t^n, \xi, \eta) := f^h(t^n, x, v)\Big|_{\mathcal{T}_{ij}}, \tag{3.40}$$

[2] *This function takes a real input and rounds up to the smallest integer that is larger than or equal to the input.*

where $(\xi, \eta) \in [-1, 1] \times [-1, 1]$ *are the variables on the canonical element. Assume that* $\tilde{f}_{ij}^{h}(t^n, \xi, \eta)$ *is non-negative at all of the following* $2MK$ *points:*

$$(\xi, \eta) = \left(\xi_{\ell jk}^{L}, \eta_k\right), \qquad \textit{where} \quad \xi_{\ell jk}^{L} := \nu_{jk}\,(1 - s_\ell) + s_\ell, \tag{3.41}$$

$$(\xi, \eta) = \left(\xi_{\ell jk}^{R}, \eta_k\right), \qquad \textit{where} \quad \xi_{\ell jk}^{R} := \nu_{jk}\,(1 + s_\ell) - 1, \tag{3.42}$$

$\forall k = 1, \ldots, M$ *and* $\forall\, \ell = 1, \ldots, K$. *In the above expression,* ν_{jk} *is given by the fractional CFL number for each 1D line. Specifically,*

$$I_{jk} := \left\lfloor \frac{u(v_{jk})\Delta t}{\Delta x} \right\rfloor \qquad \textit{and} \qquad \nu_{jk} := \frac{u(v_{jk})\Delta t}{\Delta x} - I_{jk}. \tag{3.43}$$

Each s_ℓ *denotes the* ℓ^{th} *quadrature point in the standard 1D Gauss-Legendre rule with* K *points, and* η_k *denotes the* k^{th} *quadrature point in the standard 1D Gauss-Legendre rule with* M *points.*

If one time-step in one coordinate direction is taken using the semi-Lagrangian scheme as described in Algorithm 3.2 with $f^h(t^n, x, v)$ *as the initial condition, then the approximate solution at the end of this time-step will have a non-negative average in every element (independent of the time step* Δt*):*

$$F_{ij}^{(1),new} \geq 0, \quad \forall\, \mathcal{T}_{ij} \in \Omega^h.$$

Proof. Using the notation of Algorithm 3.2, the update for the mean-value in element $\mathcal{T}_{ij}$ can be written as

$$F_{ij}^{(1),\mathrm{new}}(t^{n+1}) \;=\; \sum_{k=1}^{M} \frac{\omega_k}{2} \int_{-1}^{1} \tilde{f}_{i,jk}^{*}(\xi, \eta_k)\, d\xi \tag{3.44}$$

Because each $\omega_m \geq 0$, it suffices to show that for all i, and each equation v_{jk}, we have

$$\int_{-1}^{1} \tilde{f}_{i,jk}^{*}(\xi)\, d\xi \geq 0. \tag{3.45}$$

We define the left and right polynomials by

$$P^R_{ijk}(\xi) := \tilde{f}^h_{i-1-I_{jk}\,j}(t^n, \xi + 2 - 2\nu_{jk}, \eta_k) \quad \text{for} \quad \xi \in [-1, -1 + 2\nu_{jk}]\,,$$
$$P^L_{ijk}(\xi) := \tilde{f}^h_{i-I_{jk}\,j}(t^n, \xi - 2\nu_{jk}, \eta_k) \quad \text{for} \quad \xi \in [-1 + 2\nu_{jk}, 1]\,.$$

Since $P^L_{ijk}(\xi)$ and $P^R_{ijk}(\xi)$ are polynomials of degree at most $M-1$, we can exactly evaluate integrals of the above polynomials exactly via Gauss-Legendre quadrature rules using K points (where K is defined in (3.39)). The update for the 1D problem on the k^{th} equation for the j^{th} row simply involves evaluating the above polynomials over specified quadrature points. That is,

$$\int_{-1}^{1} \tilde{f}^*_{i,jk} := \int_{-1}^{-1+2\nu_{jk}} P^L_{ijk}(\xi)\; d\xi + \int_{-1+2\nu_{jk}}^{1} P^R_{ijk}(\xi)\; d\xi.$$

Since $P^L_{ijk}(\xi)$ and $P^R_{ijk}(\xi)$ are polynomials of degree at most $M-1$, we can exactly evaluate each of the above integrals via Gauss-Legendre quadrature rules using K points (where K is defined in (3.39)):

$$F^{(1),\text{new}}_{ij} = \frac{1}{2}\sum_{k=1}^{M} \omega_k \left\{ \sum_{\ell=1}^{K} \varpi_\ell P^R_{ijk}\left(\xi^R_{\ell jk}\right) + \sum_{\ell=1}^{K} \varpi_\ell P^L_{ijk}\left(\xi^L_{\ell jk}\right) \right\},$$

where the ϖ_ℓ's are the standard quadrature weights for Gauss-Legendre quadrature with K points.

To conclude our proof, we note that since all of the quadrature weights in the above expression for $F^{(1),\text{new}}_{ij}$ are strictly positive, and therefore we obtain positivity in the mean,

$$F^{(1),\text{new}}_{ij} \geq 0,$$

if $\tilde{f}^h_{ij}(t^n, \xi, \eta)$ is non-negative at all of the $2MK$ points defined in (3.41)–(3.42). $\square$

One of the key assumptions in the above proof of positivity in the mean is the fact that solution prior to a time-step must be positive at all of points defined in (3.41)–(3.42). We show in this subsection how to limit the solution, including the initial condition, so that we achieve positivity at all of these points. The key piece of technology necessary for achieving this positivity is a modification of the limiter of Zhang and Shu [66]. This limiter is simple to implement and is completely local to each element.

The solution on some element $\mathcal{T}$ can be written as

$$f^h(\xi,\eta) := \sum_{\ell=1}^{M(M+1)/2} F^{(\ell)}\,\varphi^{(\ell)}(\xi,\eta), \tag{3.46}$$

where M is the desired order of accuracy in space. We assume that the element average is non-negative: $F^{(1)} \geq 0$. We sample this solution on a set of *test points*:

$$(\xi_i,\,\eta_i) \in [-1,1]\times[-1,1] \qquad \text{for} \qquad i = 1,2,\ldots,P, \tag{3.47}$$

and define:

$$m := \min_{i=1,\ldots,P} f^h(\xi_i,\,\eta_i). \tag{3.48}$$

Note that $m \in (-\infty, F^{(1)}]$.

The *limited solution* is defined as follows:

$$\tilde{f}^h(\xi,\eta) := F^{(1)} + \theta\cdot\left(f^h(\xi,\eta) - F^{(1)}\right) \tag{3.49}$$

where

$$\theta = \min\left\{1,\,\frac{F^{(1)}}{F^{(1)}-m}\right\}. \tag{3.50}$$

Because $\varphi^{(1)} = 1$, the limited solution in (3.49) has an identical cell average to the non-limited solution, regardless of θ. If write $f^h(\xi,\eta)$ through its Legendre expansion, we see:

$$\tilde{f}^h(\xi,\eta) = F^{(1)} + \theta\cdot\left(\sum_{\ell=2}^{M(M+1)/2} F^{(\ell)}\varphi^{(\ell)}(\xi,\eta)\right). \tag{3.51}$$

Note that $0 \leq \theta \leq 1$ and that

$$\theta = \begin{cases} 1 & \text{if} \quad 0 \leq m \leq F^{(1)}, \\ \in [0,1) & \text{if} \quad m < 0. \end{cases} \tag{3.52}$$

This means that if the solution is already non-negative at each of the test points, then this limiter does not alter the solution. On the other hand, if the solution on the element is negative at any of the test points, then the high-order corrections, which deviate from being positive by at most $\mathcal{O}(\Delta x^{M+1})$, are *damped* until the solution is again non-negative. We are guaranteed that as $\theta \to 0$, which is only the case for extremely course grids and very inaccurate solutions, the solution will eventually become non-negative on the entire element since $F^{(1)} \geq 0$. In the case where $f^h \geq 0$ over the entire element, $\theta = 1$ and hence the limiter does absolutely nothing. One incredibly nice property of this limiter is the fact that it doesn't destroy order of accuracy.

We now state this theorem, attributed to Zhang & Shu [66, 67].

Theorem 3. *Suppose f^h is an approximate solution to f comprised of polynomials of degree $M-1$. The positivity preserving limiter expressed in equations* (3.48) *through* (3.50) *does not affect order of accuracy. That is, provided $\|f^h - f\| = \mathcal{O}\left(h^M\right)$, then $\|\tilde{f}^h - f\| = \mathcal{O}\left(h^M\right)$, where f is the exact solution. Moreover, $f^h(x_i, v_j) \geq 0$ at each point (x_i, v_j) that is sampled. Here, for simplicity, we choose to use the L^∞ norm, $\|g\| := \max_{(x,v)\in\mathcal{T}_{ij}} |g(x,v)|$.*

Proof. Here we present an outline the proof, which consists of three stages. For complete details, see Lemma 2.3 in Zhang & Shu [67].

1. The computed value of θ is larger than the 'ideal' value which uses the *exact*

minimum of the approximate solution:

$$m_{\text{ex}} := \min_{(x,v)\in\mathcal{T}_{ij}} f^h(x,v); \qquad \theta_{\text{id}} = \min\left\{1,\ \frac{F^{(1)}}{F^{(1)} - m_{\text{ex}}}\right\}.$$

The ideal value θ_{id} changes the solution more than the computed value of θ, and therefore it suffices to prove the theorem in the case where we have access to the exact minimum of the solution.

2. The amount that the approximate solution deviates from positivity is at most $\mathcal{O}(h^M)$. This is because the approximate solution is high order accurate, and the exact solution is non-negative. Applying the ideal value θ_{id} lifts the minimum value up by the exact amount it needs to move, and no more. In fact, out of all points that get moved up, this point moves the farthest.

3. The amount that any select point gets moved down is bounded by a constant, depending only on the choice of the polynomial representation space, times the largest amount that points get moved up.

□

3.4 Semi-Lagrangian Vlasov Poisson

Now that we have a full method for solving split problems, as well as a method for gluing the solutions together via high-order splitting, we essentially have a full Vlasov solver. The one necessary ingredient missing for a complete numerical scheme is a description of a Poisson solver, and in addition, in order to accommodate high-order splitting, we need a method of evaluating the electric field at intermediate time values. These two ingredients are described in this section.

3.4.1 1D Poisson solver

We describe in this section how to efficiently solve the Poisson equation in Step 2 of the operator splitting approach shown in Algorithm 3.1. Consider first the 1D Poisson equation on $x \in [a, b]$ with mixed boundary conditions:

$$-\phi_{,x,x} = \rho(x) - \rho_0, \quad -\phi_{,x}(a) = \gamma, \quad \phi(b) = \beta. \tag{3.53}$$

We apply to the Poisson equation the so-called local discontinuous Galerkin method (LDG) (see [1, 37] for two reviews of various approaches for solving Poisson equations via the DG method), and rewrite it as a system of two equations:

$$E_{,x} = \rho(x) - \rho_0, \tag{3.54}$$

$$-\phi_{,x} = E(x). \tag{3.55}$$

We expand $\phi(x)$, $E(x)$, and $\rho(x)$ on each element as follows:

$$\Big\{\phi^h(x),\, E^h(x),\, \rho^h(x) - \rho_0\Big\}\Big|_{\mathcal{T}_i} = \sum_{k=1}^{M} \Big\{\Phi_i^{(k)},\, \mathbb{E}_i^{(k)},\, \mathbb{P}_i^{(k)}\Big\}\, \varphi_{1\mathrm{D}}^{(k)}(\xi), \tag{3.56}$$

where M is the desired order of accuracy.

We multiply (3.54) and (3.55) each by $\varphi_{1\mathrm{D}}^{(\ell)}(\xi)$ and integrate from $\xi = -1$ to $\xi = 1$:

$$\frac{1}{\Delta x}\Big[\varphi_{1\mathrm{D}}^{(\ell)}\, E^h\Big]_{-1}^{1} - \frac{1}{\Delta x}\int_{-1}^{1} \varphi_{1\mathrm{D},\xi}^{(\ell)}\, E^h\, d\xi = \mathbb{P}_i^{(\ell)}, \tag{3.57}$$

$$-\frac{1}{\Delta x}\Big[\varphi_{1\mathrm{D}}^{(\ell)}\, \phi^h\Big]_{-1}^{1} + \frac{1}{\Delta x}\int_{-1}^{1} \varphi_{1\mathrm{D},\xi}^{(\ell)}\, \phi^h\, d\xi = \mathbb{E}_i^{(\ell)}. \tag{3.58}$$

Next, we apply the following *one-sided* rules in order to evaluate ϕ^h and E^h at the grid

interfaces:

$$\phi^h(-1) := \sum_{k=1}^{M} \varphi_{1\mathrm{D}}^{(k)}(-1)\,\Phi_i^{(k)} = \sum_{k=1}^{M}(-1)^{k+1}\sqrt{2k-1}\;\Phi_i^{(k)}, \tag{3.59}$$

$$\phi^h(1) := \sum_{k=1}^{M} \varphi_{1\mathrm{D}}^{(k)}(-1)\,\Phi_{i+1}^{(k)} = \sum_{k=1}^{M}(-1)^{k+1}\sqrt{2k-1}\;\Phi_{i+1}^{(k)}, \tag{3.60}$$

$$E^h(-1) := \sum_{k=1}^{M} \varphi_{1\mathrm{D}}^{(k)}(1)\,\mathbb{E}_{i-1}^{(k)} = \sum_{k=1}^{M}\sqrt{2k-1}\;\mathbb{E}_{i-1}^{(k)}, \tag{3.61}$$

$$E^h(1) := \sum_{k=1}^{M} \varphi_{1\mathrm{D}}^{(k)}(1)\,\mathbb{E}_{i}^{(k)} = \sum_{k=1}^{M}\sqrt{2k-1}\;\mathbb{E}_{i}^{(k)}. \tag{3.62}$$

Using these definitions, (3.57) and (3.58) can be rewritten as follows:

$$\sum_{k=1}^{M}\sqrt{2k-1}\sqrt{2\ell-1}\,\left(\mathbb{E}_i^{(k)} + (-1)^{\ell}\,\mathbb{E}_{i-1}^{(k)}\right) - S_{\ell k}\,\mathbb{E}_i^{(k)} = \Delta x\,\mathbb{P}_i^{(\ell)}, \tag{3.63}$$

$$-\sum_{k=1}^{M}(-1)^{k+1}\sqrt{2k-1}\sqrt{2\ell-1}\,\left(\Phi_{i+1}^{(k)} + (-1)^{\ell}\,\Phi_i^{(k)}\right) + S_{\ell k}\,\Phi_i^{(k)} = \Delta x\,\mathbb{E}_i^{(\ell)}, \tag{3.64}$$

where S is an $M \times M$ matrix with entries given by

$$S_{\ell k} = \int_{-1}^{1} \varphi_{1\mathrm{D},\xi}^{(\ell)}\,\varphi_{1\mathrm{D}}^{(k)}\,d\xi. \tag{3.65}$$

Note that the boundary conditions in (3.53) imply that

$$\mathbb{E}^h(a) = \gamma \quad \Longrightarrow \quad \sum_{k=1}^{M}\sqrt{2k-1}\,\mathbb{E}_0^{(k)} = \gamma, \tag{3.66}$$

$$\phi^h(b) = \beta \quad \Longrightarrow \quad \sum_{k=1}^{M}(-1)^{k+1}\sqrt{2k-1}\,\Phi_{m_x+1}^{(k)} = \beta, \tag{3.67}$$

where m_x is the number of grid elements.

Putting everything together, (3.63) and (3.64) can be written in matrix form:

$$\frac{1}{\Delta x}\begin{bmatrix} A & & & & \\ B & A & & & \\ & B & A & & \\ & & \ddots & \ddots & \\ & & & B & A \end{bmatrix}\begin{bmatrix} \vec{\mathbb{E}}_1 \\ \vec{\mathbb{E}}_2 \\ \vec{\mathbb{E}}_3 \\ \vdots \\ \vec{\mathbb{E}}_{m_x} \end{bmatrix} = \begin{bmatrix} \vec{\mathbb{P}}_1 - (-1)^\ell \sqrt{2\ell-1}\,\gamma\,(\Delta x)^{-1} \\ \vec{\mathbb{P}}_2 \\ \vec{\mathbb{P}}_3 \\ \vdots \\ \vec{\mathbb{P}}_{m_x} \end{bmatrix}, \tag{3.68}$$

$$\frac{1}{\Delta x}\begin{bmatrix} C & D & & & \\ & C & D & & \\ & & C & \ddots & \\ & & & \ddots & D \\ & & & & C \end{bmatrix}\begin{bmatrix} \vec{\Phi}_1 \\ \vec{\Phi}_2 \\ \vec{\Phi}_3 \\ \vdots \\ \vec{\Phi}_{m_x} \end{bmatrix} = \begin{bmatrix} \vec{\mathbb{E}}_1 \\ \vec{\mathbb{E}}_2 \\ \vec{\mathbb{E}}_3 \\ \vdots \\ \vec{\mathbb{E}}_{m_x} + \sqrt{2\ell-1}\,\beta\ (\Delta x)^{-1} \end{bmatrix}, \tag{3.69}$$

where, for example,

$$\vec{\mathbb{E}}_i = \left(\mathbb{E}_i^{(1)}, \ldots, \mathbb{E}_i^{(M)}\right)^T, \tag{3.70}$$

and A, B, C, and D are $M \times M$ matrices with entries given by:

$$A_{\ell k} = \sqrt{2k-1}\sqrt{2\ell-1} - S_{\ell k}, \tag{3.71}$$

$$B_{\ell k} = (-1)^\ell \sqrt{2k-1}\sqrt{2\ell-1}, \tag{3.72}$$

$$C_{\ell k} = (-1)^{k+\ell} \sqrt{2k-1}\sqrt{2\ell-1} + S_{\ell k}, \tag{3.73}$$

$$D_{\ell k} = (-1)^k \sqrt{2k-1}\sqrt{2\ell-1}. \tag{3.74}$$

The advantage of this formulation is that equations (3.68) and (3.69) are already in lower and upper triangular forms, respectively, and therefore can be easily be solved. The matrices A and C can be easily inverted once at the beginning of the calculation.

Dirichlet boundary conditions

The above method can easily be adapted to handle Dirichlet boundary conditions:

$$\phi(a) = \alpha, \quad \phi(b) = \beta, \tag{3.75}$$

by noting that these boundary conditions are equivalent to the mixed BCs in (3.53) if we carefully choose the parameter γ in (3.53). It can be shown that the correct choice for γ is given by

$$\begin{aligned} \gamma &= \frac{\beta - \alpha}{a - b} + \frac{1}{a - b} \int_a^b (s - b) \Big(\rho(s) - \rho_0 \Big) \, ds \\ &= \frac{\beta - \alpha}{a - b} + \frac{\Delta x}{a - b} \sum_{i=1}^{m_x} (x_i - b) \, \mathbb{P}_i^{(1)} + \frac{\Delta x^2}{2\sqrt{3}\,(a - b)} \sum_{i=1}^{m_x} \mathbb{P}_i^{(2)}, \end{aligned} \tag{3.76}$$

where m_x is the number of grid elements.

Periodic boundary conditions

Periodic boundary conditions can also be readily handled:

$$\phi(a) = \phi(b), \quad E(a) = E(b), \tag{3.77}$$

by again noting that we need to carefully choose β and γ in (3.53). It can be shown that the correct choice for γ is given by

$$\gamma = \frac{1}{a - b} \int_a^b s \Big(\rho(s) - \rho_0 \Big) \, ds = \frac{\Delta x}{a - b} \sum_{i=1}^{m_x} x_i \mathbb{P}_i^{(1)} + \frac{\Delta x^2}{2\sqrt{3}\,(a - b)} \sum_{i=1}^{m_x} \mathbb{P}_i^{(2)}, \tag{3.78}$$

and β is arbitrary. Without loss of generality we simply take $\beta = 0$. By solving (3.53) with γ given by (3.78) and with $\beta = 0$, we obtain a solution $\phi(x)$ with the property that

$$\phi(a) = \phi(b) = 0. \tag{3.79}$$

3.4.2 High-order Electric Field

One difficulty with raising the temporal order of accuracy from two to four is the time-dependence of the electric field. In other words, the Cheng and Knorr [12] method does not completely reduce the Vlasov-Poisson to two constant coefficient problems, since the electric field remains time-dependent. In the case of Strang splitting, it turned out that one could easily generate a second order accurate representation of the electric field at the half time step, $t^n + \frac{1}{2}\Delta t$, as required in Step 3 of Algorithm 3.1, simply by carrying out Steps 1 and 2 of Algorithm 3.1. Additional attention must be paid in order to obtain temporally fourth-order accurate representations of the electric field. We describe in this section some important implementation details needed to achieve this. The resulting method will be summarized in Algorithm 3.3, but first some preliminary details.

In order to avoid having to use the electric field at different points in time (i.e., a multi-step method, which we discovered has some stability issues), we construct the fourth-order Taylor polynomial centered at $t = t^n$:

$$\bar{\mathbf{E}}(t, \mathbf{x}) := \mathbf{E}^n + (t - t^n)\, \mathbf{E}^n_{,t} + \frac{1}{2}\,(t - t^n)^2\, \mathbf{E}^n_{,t,t} + \frac{1}{6}\,(t - t^n)^3\, \mathbf{E}^n_{,t,t,t}. \tag{3.80}$$

In this section, we first describe a method for computing time derivatives of the electric field in (1+1)D, and then we describe how to extend this to multi-D.

High-Order (1+1)D Electric Field

The starting point for constructing high-order values for the electric field is to compute the electric field at time level t^n through the Poisson equation:

$$-\phi^n_{,x,x} = E^n_{,x} = \rho^n - \rho_0. \tag{3.81}$$

The first time derivative is readily computed by:

$$E_{,t,x} = \rho_{,t,x} = -\left(\rho u\right)_{,x} \quad \Longrightarrow \quad E^n_{,t} = -\left(\rho u\right)^n + J_0(t). \tag{3.82}$$

In order to compute further time derivatives of E, we appeal to the first few moment equations of the Vlasov-Poisson equation (1.19) which were presented in §1.2.4. The (1+1)D version of those are:

$$\rho_{,t} + \left(\rho u\right)_{,x} = 0, \tag{3.83}$$

$$\left(\rho u\right)_{,t} + \mathcal{E}_{,x} = \rho E, \tag{3.84}$$

$$\mathcal{E}_{,t} + \mathcal{F}_{,x} = 2\rho u E, \tag{3.85}$$

where

$$\rho := \int_v f\, dv, \quad \rho u := \int_v v f\, dv, \quad \mathcal{E} := \int_v v^2 f\, dv, \quad \text{and} \quad \mathcal{F} := \int_v v^3 f\, dv.$$

Using equation (3.82) and the above moment evolution equations, we can compute the second and third time derivatives of the electric potential entirely in terms of spatial derivatives:

$$E_{,t} = -\rho u + J_0, \tag{3.86}$$

$$E_{,t,t} = \mathcal{E}_{,x} - \rho\, E + \dot{J}_0, \tag{3.87}$$

$$E_{,t,t,t} = 2\left(\rho u E\right)_{,x} - \mathcal{F}_{,x,x} + E\left(\rho u\right)_{,x} + \rho^2 u - \rho J_0 + \ddot{J}_0, \tag{3.88}$$

where the integration constant, $J_0(t)$, satisfies the following properties:

$$J_0 = \frac{1}{L}\int_0^L \rho u\, dx, \tag{3.89}$$

$$\dot{J}_0 = \frac{1}{L}\int_0^L \rho E\, dx, \tag{3.90}$$

$$\ddot{J}_0 = J_0\left(\frac{1}{L}\int_0^L \rho\, dx - \rho_0\right). \tag{3.91}$$

When we use periodic boundary conditions for ϕ, that is $\phi(t,0) = \phi(t,L)$, then J_0 is constant for all time. To see this, multiply (3.81) by E:

$$EE_{,x} = \rho E - \rho_0 E \implies \rho E = \left(\frac{1}{2} \left(E^2 \right) - \rho_0 \phi \right)_{,x}. \tag{3.92}$$

Integrating (3.92) over the entire domain gives

$$\dot{J}_0 = \frac{1}{L} \int_0^L \rho E \; dx = 0, \tag{3.93}$$

because E^2 and ϕ are both periodic. Therefore, in equations (3.86) – (3.88), we need only numerically compute this constant at the start of the problem.

It is clear from these expressions that in order to compute $E_{,t,t}$ and $E_{,t,t,t}$, we need to be able to compute first and second derivatives in space. One approach for doing this in the discontinuous Galerkin framework is to multiply by a test function and then integrate-by-parts. However, this approach will in general lead to a loss of accuracy. Instead, the approach taken in this work is to apply central finite differences that work directly on the Legendre coefficients of the function that needs to be differentiated.

Consider the L_2-projection of the function $f(x)$, where $f : \mathbb{R} \to \mathbb{R}$, onto the space of piecewise polynomials of degree four on a uniform mesh of elements, $\mathcal{T}_i$, that each have width Δx:

$$f^h \Big|_{\mathcal{T}_i} = \sum_{\ell=1}^{5} F_i^{(\ell)} \, \varphi_{1\mathrm{D}}^{(\ell)}(\xi). \tag{3.94}$$

Therefore, f^h represents the finite dimensional approximation of $f(x)$. We can approximate the first and second derivatives of $f(x)$ to $\mathcal{O}\left(\Delta x^5\right)$ accuracy by computing appropriate central finite differences of the Legendre coefficients $F^{(\ell)}$. If we let $D_x f^h$ and $D_{xx} f^h$ represent the finite dimensional approximations of $f'(x)$ and $f''(x)$, respectively,

then the central finite difference formulas on the Legendre coefficients are

$$\begin{bmatrix} D_x F_i^{(1)} \\ D_x F_i^{(2)} \\ D_x F_i^{(3)} \\ D_x F_i^{(4)} \\ D_x F_i^{(5)} \end{bmatrix} = \frac{1}{2\Delta x} \begin{bmatrix} \Delta_1 F_i^{(1)} - 2\sqrt{5}\,\Delta_1 F_i^{(3)} + 78\,\Delta_1 F_i^{(5)} \\ \Delta_1 F_i^{(2)} - \frac{10}{3}\sqrt{3}\sqrt{7}\,\Delta_1 F_i^{(4)} \\ \Delta_1 F_i^{(3)} - 14\sqrt{5}\,\Delta_1 F_i^{(5)} \\ \Delta_1 F_i^{(4)} \\ \Delta_1 F_i^{(5)} \end{bmatrix}, \tag{3.95}$$

$$\begin{bmatrix} D_{xx} F_i^{(1)} \\ D_{xx} F_i^{(2)} \\ D_{xx} F_i^{(3)} \\ D_{xx} F_i^{(4)} \\ D_{xx} F_i^{(5)} \end{bmatrix} = \frac{1}{\Delta x^2} \begin{bmatrix} \Delta_2 F_i^{(1)} - \sqrt{5}\,\Delta_2 F_i^{(3)} + 11\,\Delta_2 F_i^{(5)} \\ \Delta_2 F_i^{(2)} - \frac{5}{3}\sqrt{3}\sqrt{7}\,\Delta_2 F_i^{(4)} \\ \Delta_2 F_i^{(3)} - 7\sqrt{5}\,\Delta_2 F_i^{(5)} \\ \Delta_2 F_i^{(4)} \\ \Delta_2 F_i^{(5)} \end{bmatrix}, \tag{3.96}$$

where

$$\Delta_1 F_i^{(k)} := F_{i+1}^{(k)} - F_{i-1}^{(k)}, \tag{3.97}$$

$$\Delta_2 F_i^{(k)} := F_{i+1}^{(k)} - 2F_i^{(k)} + F_{i-1}^{(k)}. \tag{3.98}$$

To the best of our knowledge, this is the first time such formulas have been written down in the context of discontinuous Galerkin methods. In Table 2 we verify the order of accuracy by computing the first and second derivatives of $f(x) = e^{\sin(2\pi x)}$. The errors in this table are computed using the relative L_2 errors defined by equation (A.10) with $M = 5$ and varying Δx. See A.1 for more details.

Mesh	$f'(x)$ **error**	$\log_2$**(Ratio)**	$f''(x)$ **error**	$\log_2$**(Ratio)**
25	1.747×10^{-4}	–	8.292×10^{-5}	–
50	5.543×10^{-6}	4.98	2.672×10^{-6}	4.96
100	1.738×10^{-7}	5.00	8.413×10^{-8}	4.99
200	5.437×10^{-9}	5.00	2.634×10^{-9}	5.00
400	1.699×10^{-10}	5.00	8.364×10^{-11}	4.98

Table 2: Relative L_2-norm errors for computing $f'(x)$ and $f''(x)$ using the Legendre coefficient finite difference formulas (3.95) and (3.96), respectively. The example shown here is for $f(x) = e^{\sin(2\pi x)}$ on a uniform mesh on $0 \leq x \leq 1$. Periodic boundary conditions are imposed to compute the derivative in the first and last elements: $F_0^{(k)} = F_M^{(k)}$ and $F_{M+1}^{(k)} = F_1^{(k)}$ for each $k = 1, 2, 3, 4, 5$.

Extensions to Multi-D

The starting point for the multi-D extension is identical to the previous section, where we compute the electric field value $\mathbf{E}^n$ with the Poisson equation:

$$\nabla \cdot \mathbf{E}^n = -\nabla^2 \phi^n = \rho^n - \rho_0. \tag{3.99}$$

In this case, the first time derivative of the electric field is proportional to the momentum, up to the curl of a vector field,

$$\nabla \cdot \mathbf{E}_{,t} = \rho_{,t} = -\nabla \cdot (\rho \mathbf{u}) \quad \Longrightarrow \quad \mathbf{E}_{,t}^n = -(\rho \mathbf{u})^n + \nabla \times \mathbf{C}(t, \mathbf{x}). \tag{3.100}$$

In order to avoid navigating the integration constant, we suggest working with the electric potential, ϕ in order to compute time derivatives of $\mathbf{E}$:

$$\mathbf{E}_{,t} = -\nabla \phi_{,t}, \quad \mathbf{E}_{,t,t} = -\nabla \phi_{,t,t}, \quad \mathbf{E}_{,t,t,t} = -\nabla \phi_{,t,t,t}. \tag{3.101}$$

Once we have a method for computing $\phi_{,t}, \phi_{,t,t}, \phi_{,t,t,t}$, (up to an arbitrary constant), we can run three additional Poisson solves to compute the three required time derivatives

of $\mathbf{E}$. In order to compute time derivatives of ϕ, we again need the evolution equations for the first few moments of the Vlasov-Poisson equations. Here, we collect equations (1.27), (1.28) and (1.30):

$$\begin{aligned}
\rho_{,t} + \nabla \cdot (\rho \mathbf{u}) &= 0, \\
(\rho \mathbf{u})_{,t} + \nabla \cdot \mathbb{E} &= \rho \mathbf{E}, \\
\mathbb{E}_{,t} + \nabla \cdot \mathbb{F} &= \rho \left(\mathbf{u}\mathbf{E} + \mathbf{E}\mathbf{u} \right).
\end{aligned}$$

This time, the moments are multi-D tensors:

$$\rho := \int_{\mathbf{v}} f \, d\mathbf{v}, \quad \rho \mathbf{u} := \int_{\mathbf{v}} \mathbf{v} f \, d\mathbf{v}, \quad \mathbb{E} := \int_{\mathbf{v}} \mathbf{v}\mathbf{v} f \, d\mathbf{v}, \quad \text{and} \quad \mathbb{F} := \int_{\mathbf{v}} \mathbf{v}\mathbf{v}\mathbf{v} f \, d\mathbf{v}.$$

Using equation (3.99) and the above moment evolution equations, we can compute the second and third time derivatives of the electric potential entirely in terms of spatial derivatives:

$$-\nabla^2 \left(\phi_{,t} \right) = -\nabla \cdot (\rho \mathbf{u}), \tag{3.102}$$

$$-\nabla^2 \left(\phi_{,tt} \right) = \nabla \cdot \nabla \cdot \mathbb{E} - \nabla \cdot (\rho \mathbf{E}), \tag{3.103}$$

$$\begin{aligned}
-\nabla^2 \left(\phi_{,ttt} \right) = &-\nabla \cdot \nabla \cdot \nabla \cdot \mathbb{F} + \nabla \cdot \nabla \cdot (\rho \mathbf{u}\mathbf{E} + \rho \mathbf{E}\mathbf{u}) \\
&+ \nabla \cdot (\mathbf{E} \nabla \cdot (\rho \mathbf{u})) - \nabla \cdot (\rho \mathbf{E}_{,t}).
\end{aligned} \tag{3.104}$$

Again, one needs to be able to compute first, second and this time third derivatives in order to attain a high-order electric field. The exact details for how to accomplish this will be the subject of future work.

Time-Dependent Advection with the Electric Field

Once we have constructed an approximation to the time-dependent electric field, we are faced with an advection equation with time-dependent coefficients:

$$f_{,t} + \bar{\mathbf{E}}(t, \mathbf{x}) \cdot f_{,\mathbf{v}} = 0. \tag{3.105}$$

This equation can be readily solved to high-order via the method of characteristics. The key step in this approach is the evolution of the characteristics $\mathbf{v}$ as function of time:

$$\frac{d\mathbf{v}}{dt} = \bar{\mathbf{E}}(t, \mathbf{x}) \quad \Longrightarrow \quad \mathbf{v}(t^n + \Delta t) = \mathbf{v}(t^n) + \int_{t^n}^{t^n + \Delta t} \bar{\mathbf{E}}(t, \mathbf{x})\, dt, \tag{3.106}$$

$$\mathbf{v}(t^n + \Delta t) = \mathbf{v}(t^n) + \Delta t\, \mathbf{E}^n + \frac{\Delta t^2}{2}\, \mathbf{E}^n_{,t} + \frac{\Delta t^3}{6}\, \mathbf{E}^n_{,t,t} + \frac{\Delta t^4}{24}\, \mathbf{E}^n_{,t,t,t}. \tag{3.107}$$

In other words, the semi-Lagrangian DG method as outlined in §3.3 remains largely unaltered by the fact that the electric is time dependent. The only difference is that interfaces and quadrature points are transported by equation (3.107) instead of the simpler version of this equation when $\bar{\mathbf{E}}$ is constant in time. In order to recycle existing code, suppose a routine,

$$\text{StepAdvec}(f,\, \mathbf{b}(\mathbf{x}),\, \Delta t)$$

has been written which takes a single step of length Δt on:

$$f_{,t} + \mathbf{b}(\mathbf{x}) \cdot f_{,\mathbf{v}} = 0.$$

Suppose we wish to take a time step over the interval $[t_1, t_2]$, and the time step length is $h = t_2 - t_1$. If we define $\mathbf{E}^*(\mathbf{x}) := \frac{1}{h} \int_{t_1}^{t_2} \bar{\mathbf{E}}(s, \mathbf{x})\, ds$, then running

$$\text{StepAdvec}(f,\, \mathbf{E}^*(\mathbf{x}),\, h)$$

will accommodate the exact translation and integration characteristics, assuming we integrate $\bar{\mathbf{E}}(t, \mathbf{x})$ exactly, which we can since it is a polynomial. This is identical to

what was presented in 3.3.1 for time dependant equations in 1D. We further note that this will naturally accommodate negative time steps, because in this case a negative h simply reversed the direction in which characteristics are traced, which is exactly what it is supposed to do.

The complete fourth-order splitting method for the Vlasov-Poisson system is summarized in Algorithm 3.3. We note that there, we describe a multi-D version of this problem, but in this work, we are able to use equations (3.86) – (3.88) in place of the Poisson solves listed in Step 2 of the algorithm.

Positivity Limiter

The limiter described in §3.3.4 is applied during the following steps:

1. During each of the stages labelled $3 - 9$ in Algorithm 3.3, we apply the positivity limiter with the test points given by (3.41)–(3.42). For Algorithm 3.1, we apply the limiter during stages 1, 3 and 4. In this case the number of points sampled per element is $P = 2MK$, where M is the desired order of accuracy in space and K is defined by (3.39). As proved in Theorem 2, this guarantees that in each stage the approximate solution remains positive in the mean.

2. After all of the stages of Algorithm 3.3 have been carried out, we apply the positivity limiter one more time to the solution, this time with the test points taken as the $P = M^2$ Gauss-Legendre quadrature points on $[-1, 1] \times [-1, 1]$. This final limiting provides some additional positivity enforcement and allows us to compute a variety of integrals of the form (A.1) with function values that are non-negative. This is particularly useful in computing the L_1-norm (A.3), the total energy (A.5),

and the entropy (A.6).

3.5 Numerical examples

Now that we have all of the basic pieces are in place for a full Vlasov Poisson solver (i.e., a semi-Lagrangian solver for each split-piece §3.3, and a Poisson solver §3.4.1), we apply the full force of all the methods described in §3.3 and §3.4 to a variety of numerical test cases. We begin in §3.5.1 by considering a linear advection equation with a velocity field that produces solid body rotation. This example is primarily used to show the benefits of switching from second to fourth-order operator splitting strategies. In §3.5.2 we verify the order of accuracy of the method on a forced Vlasov-Poisson equation to which we know the exact solution. In the subsequent three subsection we consider three standard problems for the Vlasov-Poisson system: §3.5.3 the two-stream instability, §3.5.4 weak Landau damping, and §3.5.5 strong Landau damping. In addition, we present a plasma sheath problem in section §3.5.6 which demonstrates how to accommodate non-periodic boundary conditions. Unless otherwise stated, all simulations in this Chapter are 5^{th} order in space and with the positivity-preserving limiters as described in §3.3.4 turned on.

3.5.1 Linear advection

We first consider a linear advection under a divergence-free velocity field:

$$q_{,t} + \mathbf{u} \cdot \nabla q = 0. \tag{3.108}$$

Algorithm 3.3 Fourth-order operator split algorithm for Vlasov-Poisson.

1. Solve $-\nabla^2\phi = \rho^n - \rho_0$ and compute $\mathbf{E}^n = -\nabla\phi$.

2. Compute

$$\begin{aligned}
\mathbf{E}^n_{,t} &= -\nabla\phi^n_{,t} \;; & -\nabla^2\left(\phi^n_{,t}\right) &= -\nabla\cdot(\rho\mathbf{u})^n\,,\\
\mathbf{E}^n_{,t,t} &= -\nabla\phi^n_{,t,t} \;; & -\nabla^2\left(\phi_{,t,t}\right) &= \nabla\cdot\nabla\cdot\mathbb{E}^n - \nabla\cdot(\rho\mathbf{E})^n\,,\\
\mathbf{E}^n_{,t,t,t} &= -\nabla\phi^n_{,t,t,t}; & -\nabla^2\left(\phi_{,t,t,t}\right) &= -\nabla\cdot\nabla\cdot\nabla\cdot\mathbb{F}^n + \nabla\cdot\nabla\cdot(\rho\mathbf{u}\mathbf{E}+\rho\mathbf{E}\mathbf{u})^n\\
& & &\quad + \nabla\cdot(\mathbf{E}^n\nabla\cdot(\rho\mathbf{u})^n) - \nabla\cdot(\rho\mathbf{E}_{,t})^n\,,
\end{aligned}$$

   where the spatial derivatives are computed via (3.95) and (3.96).
   Then construct

$$\bar{\mathbf{E}}(t,\mathbf{x}) := \mathbf{E}^n + (t-t^n)\,\mathbf{E}^n_{,t} + \frac{1}{2}(t-t^n)^2\,\mathbf{E}^n_{,tt} + \frac{1}{6}(t-t^n)^3\,\mathbf{E}^n_{,ttt}.$$

   For the 1D method, we use equations (3.86)–(3.88) in place of computing time derivatives of the potential.

3. $\frac{\gamma_1}{2}\Delta t$ step on $f_{,t} + \mathbf{v}\cdot f_{,\mathbf{x}} = 0$.

4. $\gamma_1\Delta t$ step on $f_{,t} + \bar{\mathbf{E}}\,(t,\mathbf{x})\cdot f_{,\mathbf{v}} = 0, \quad t \in t^n + \Big[0,\, \gamma_1\Delta t\Big]$.

5. $\frac{(\gamma_1+\gamma_2)}{2}\Delta t$ step on $f_{,t} + \mathbf{v}\cdot f_{,\mathbf{x}} = 0$.

6. $\gamma_2\Delta t$ step on $f_{,t} + \bar{\mathbf{E}}\,(t,\mathbf{x})\cdot f_{,\mathbf{v}} = 0, \quad t \in t^n + \Big[\gamma_1\Delta t,\, (\gamma_1+\gamma_2)\Delta t\Big]$.

7. $\frac{(\gamma_1+\gamma_2)}{2}\Delta t$ step on $f_{,t} + \mathbf{v}\cdot f_{,\mathbf{x}} = 0$.

8. $\gamma_1\Delta t$ step on $f_{,t} + \bar{\mathbf{E}}\,(t,\mathbf{x})\cdot f_{,\mathbf{v}} = 0, \quad t \in t^n + \Big[(\gamma_1+\gamma_2)\Delta t,\, (2\gamma_1+\gamma_2)\Delta t\Big]$.

9. $\frac{\gamma_1}{2}\Delta t$ step on $f_{,t} + \mathbf{v}\cdot f_{,\mathbf{x}} = 0$.

We take the computational domain to be $[0,1] \times [0,1]$ and the velocity field to be solid body rotation around $(0.5, 0.5)$:

$$\mathbf{u} = (u(y), v(x)) = (\pi (2y - 1), \pi (1 - 2x))^T . \tag{3.109}$$

The initial condition is taken to be a smooth, compactly supported bump that is centered at $(x_0, y_0) = (0.4, 0.5)$:

$$q(0, x, y) = \begin{cases} \cos^6 \left(\frac{5\pi}{3} r\right) & \text{if} \quad r \leq 0.3, \\ 0 & \text{otherwise,} \end{cases} \tag{3.110}$$

where

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2}. \tag{3.111}$$

This problem, just as the Vlasov-Poisson system, is solved via operator splitting on the two operators:

$$\text{Problem } \mathcal{A}: \quad q_{,t} + u(y)\, q_{,x} = 0, \tag{3.112}$$

$$\text{Problem } \mathcal{B}: \quad q_{,t} + v(x)\, q_{,y} = 0. \tag{3.113}$$

We run the initial condition out to time $t = 1$, at which point it should return to its initial state. The errors are computed using the relative L_2 errors defined by equation (A.14) with $M = 5$ and varying $\Delta x = \Delta y$. See A.2 for more details. Convergence studies with Strang and the fourth-order operator splitting results are shown in Table 3.

3.5.2 A forced problem: verifying order of accuracy

Next we consider an example of a forced Vlasov-Poisson equation. By construction of the source term, we can compare our approximate solution with the exact solution. The

Mesh	**SL2 Error**	$\log_2$**(Ratio)**	**SL4 Error**	$\log_2$**(Ratio)**
10^2	3.215×10^{-1}	–	5.679×10^{-1}	–
20^2	7.185×10^{-2}	2.16	3.113×10^{-2}	4.19
40^2	1.578×10^{-2}	2.19	1.691×10^{-3}	4.20
80^2	3.923×10^{-3}	2.01	1.010×10^{-4}	4.07
160^2	9.890×10^{-4}	1.99	6.220×10^{-6}	4.02
320^2	2.454×10^{-4}	2.01	3.843×10^{-7}	4.02
640^2	6.136×10^{-5}	2.00	2.390×10^{-8}	4.01

Table 3: Convergence study for the linear advection equation. Shown are the relative errors computed via (A.14) at time $t = 1$. All calculations were done with 5^{th} order accuracy in space using the positivity-preserving limiters and a CFL number of 5.00, where CFL $:= \Delta t \max\{\max_y |u(y)|/\Delta x, \max_x |v(x)|/\Delta y\}$. SL2 refers to the Strang split semi-Lagrangian scheme and SL4 to the fourth-order split semi-Lagrangian scheme.

forced Vlasov-Poisson system is

$$f_{,t} + v f_{,x} + E f_{,v} = \psi(t, x, v), \tag{3.114}$$

$$E_{,x} = \int_{-\infty}^{\infty} f(t, x, v)\, dv - \sqrt{\pi}, \tag{3.115}$$

on $(t, x, v) \in [0, \infty) \times [-\pi, \pi] \times (-\infty, \infty)$ with periodic boundary conditions in x. We take the following source term:

$$\begin{aligned}\psi(t, x, v) = \frac{1}{2}\sin(2x - 2\pi t) e^{-\frac{1}{4}(4v-1)^2} \Big\{ &\left(2\sqrt{\pi} + 1\right)\left(4v - 2\sqrt{\pi}\right) \\ &- \sqrt{\pi}\,(4v - 1)\cos(2x - 2\pi t)\Big\}.\end{aligned} \tag{3.116}$$

The exact solution in this case is

$$f(t, x, v) = \left\{2 - \cos\left(2x - 2\pi t\right)\right\} e^{-\frac{1}{4}(4v-1)^2}, \tag{3.117}$$

$$E(t, x) = -\frac{\sqrt{\pi}}{4}\,\sin\left(2x - 2\pi t\right). \tag{3.118}$$

The numerical scheme for this forced problem is the same as the one described in §3.3 with two minor modifications. First, the two operators in the operator split scheme are

$$\text{Problem } \mathcal{A}\text{:} \quad f_{,t} + v\, f_{,x} = \psi(t,x,v), \tag{3.119}$$

$$\text{Problem } \mathcal{B}\text{:} \quad f_{,t} + E(t,x)\, f_{,v} = 0, \tag{3.120}$$

which means that Problem $\mathcal{A}$ is slightly modified from the unforced Vlasov-Poisson system. The modified $\mathcal{A}$ still has the same characteristics as the case with no source term; the only difference is that the solution is no longer constant along the characteristics. In order to advance f forward under the influence of operator $\mathcal{A}$, we use the method of characteristics and obtain the following solution:

$$f(t^{n+1},x,v) = f(t^n, x - v\Delta t, v) + \int_{t^n}^{t^{n+1}} \psi(s, x + v(s - t^{n+1}), v)\, ds. \tag{3.121}$$

The time integral in the above expression can be easily exactly evaluated; we omit the details here.

The second modification comes from the fact that with a non-zero source term, it is no longer true that $E_{,t} = -\rho u$. This means that the time interpolation described in §3.3 must be slightly modified. Instead of using $E_{,t} = -\rho u$, (3.87), and (3.88), we make use of the following modified formulas:

$$E_{,t}(t,x) = -\rho u + C_1, \tag{3.122}$$

$$E_{,tt}(t,x) = \mathbb{E}_{,x} - \rho E + C_2, \tag{3.123}$$

$$E_{,ttt}(t,x) = (2\rho u E)_{,x} - \mathbb{F}_{,x,x} + E\,(\rho u)_{,x} + \rho^2 u + C_3, \tag{3.124}$$

where

$$C_1 := \frac{\sqrt{\pi}}{4} + \frac{\sqrt{\pi}}{8}(4\pi - 1)\cos(2x - 2\pi t), \tag{3.125}$$

$$C_2 := \frac{3\sqrt{\pi} + 4\pi - 16\sqrt{\pi^5}}{16}\sin(-2x + 2\pi t) + \frac{\pi}{16}\sin(4x - 4\pi t), \tag{3.126}$$

$$C_3 := -\frac{\pi}{4} + \frac{7\sqrt{\pi} + 16\pi - 64\sqrt{\pi^7}}{32}\cos(2x - 2\pi t) - \frac{3\pi}{16}\cos(4x - 4\pi t). \tag{3.127}$$

In the above expression we used the shorthand notation:

$$x := x^1, \quad E := E^1, \quad u := u^1, \quad \mathbb{E} := \mathbb{E}^{11}, \quad \text{and} \quad \mathbb{F} := \mathbb{F}^{111}.$$

We run the initial condition out to time $t = 1$, at which point it should return to its initial state. Convergence studies on various grids on the domain $(x, v) \in [-\pi, \pi] \times [-\pi, \pi]$ with Strang and the fourth-order operator splitting results are shown in Table 4. We compute the errors in an identical manner to the linear test problem presented in the previous section using the relative L_2 errors defined by equation (A.14) with $M = 5$. See A.2 for more details.

3.5.3 Two-stream instability

The two-stream instability problem has become a standard benchmark to test numerical Vlasov solvers, and has been used as such by several authors (e.g., [36, 3, 49, 30, 15, 12]). We use the following initial distribution function

$$f(t = 0, x, v) = \frac{v^2}{\sqrt{8\pi}}\left(2 - \cos\left(\frac{x}{2}\right)\right)e^{-\frac{v^2}{2}}, \tag{3.128}$$

and solve on the domain $(x, v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. Results for time $t = 45$ are presented in Figure 8 for various mesh sizes. In Figure 9 we present cross-sections of the solution taken at $x = 0$ for the same mesh sizes.

Mesh	**SL2 Error**	$\log_2$**(Ratio)**	**SL4 Error**	$\log_2$**(Ratio)**
10^2	5.210×10^{-1}	–	9.493×10^{-1}	–
20^2	1.433×10^{-1}	1.86	2.715×10^{-1}	1.81
40^2	1.640×10^{-2}	3.13	1.652×10^{-2}	4.04
80^2	3.438×10^{-3}	2.26	7.079×10^{-4}	4.55
160^2	8.333×10^{-4}	2.04	3.434×10^{-5}	4.37
320^2	2.068×10^{-4}	2.01	1.962×10^{-6}	4.13
640^2	5.161×10^{-5}	2.00	1.203×10^{-7}	4.03
1280^2	1.290×10^{-5}	2.00	7.509×10^{-9}	4.00

Table 4: Convergence study for the forced Vlasov-Poisson equation, comparing two operator split methods. All calculations presented here are 5^{th} order in space and were run with a CFL number of 5. Shown are the relative errors computed via (A.14) at time $t = 1$. SL2 refers to the Strang split semi-Lagrangian scheme and SL4 to the fourth-order split semi-Lagrangian scheme. Since the positivity-preserving limiters as described in §3.3.4 don't guarantee positivity in the mean in the presence of a source term, we have turned them off for this convergence study only.

Figures 8 and 9 clearly demonstrate the ability of the discontinuous Galerkin methodology to approximate very rough data, something that is more difficult with methods that act over larger stencils. The results shown in these figures indicate far more structure than what is shown in other recent work, including results from the WENO method [49, 3] and an explicit DG method that uses a piecewise constant representation of the distribution function, f, and a piecewise quadratic representation of the electric potential ϕ [36].

In Figure 10 we demonstrate the effects of adding the positivity-preserving limiter. We see that even without limiting, the base scheme already does a reasonable job of not producing large negative values in the distribution function. With the positivity-preserving limiters we are able to remove these small positivity violations. In Figure 12 we plot four quantities that are exactly conserved by the continuous Vlasov-Poisson equation, but only approximately conserved in our numerical discretization: L_1-norm of f (1.49), L_2-norm of f (1.50), total energy (1.51), and total entropy (1.52). In particular, we use the numerical approximations to (1.49)–(1.52) as given by equations (A.3)–(A.6) in §A. We note that it is difficult to obtain accurate values for the total entropy (1.52), because there are many values where f becomes very small.

3.5.4 Weak Landau damping

Landau damping has been extensively studied both numerically [12, 36, 68] and analytically (e.g., the work of Mouhot and Villani [46, 47]). Just as the two-stream instability problem, Landau damping has become a favorite standard test case. It is particularly useful since the linear decay rates of the L_2-norm of the electric field are well-known.

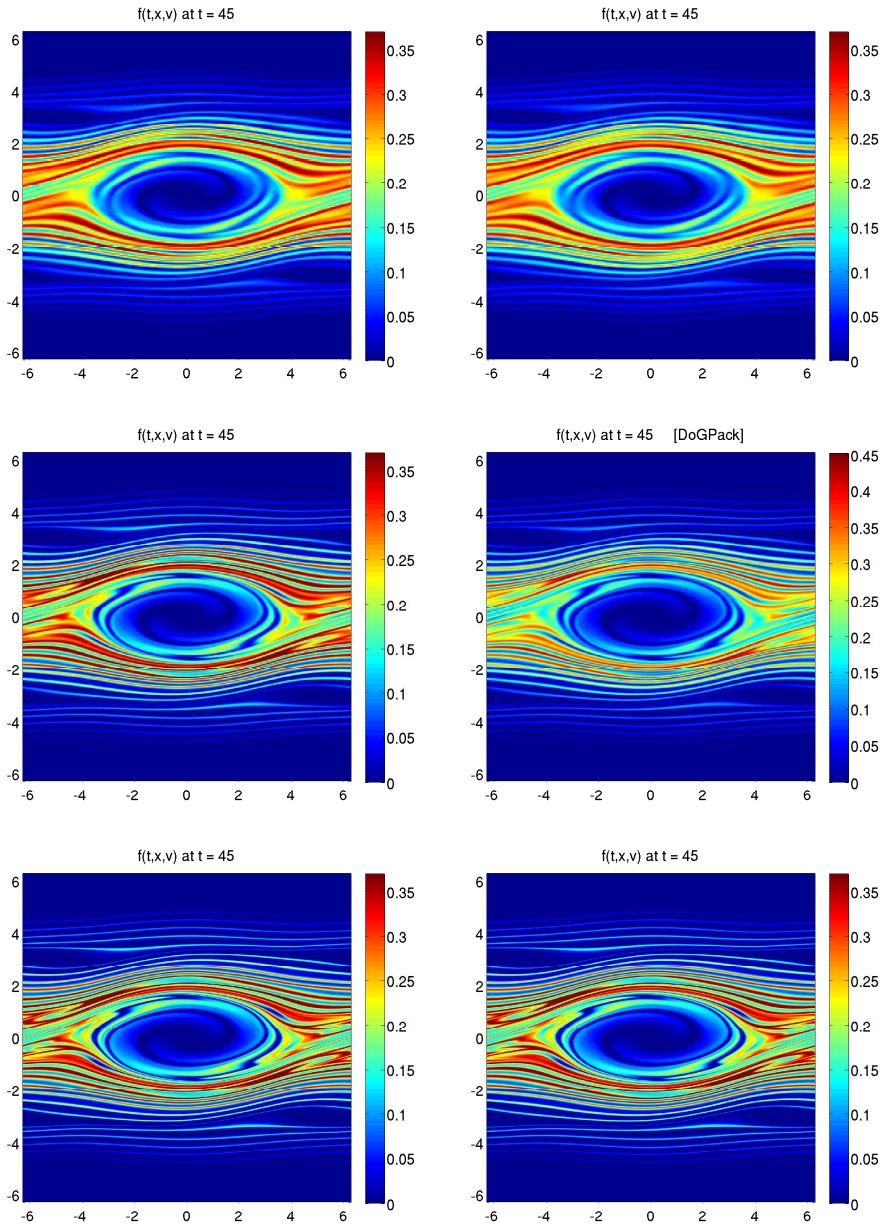


Figure 8: The two-stream instability problem. The panels in the left-hand column are results using the 2^{nd} order Strang splitting method. The panels in the right-hand column are results using the 4^{th} order splitting method. All simulations are 5^{th} order in space. The mesh sizes for the first, second and third rows are $(m_x, m_v) = (65, 65)$, $(m_x, m_v) = (129, 129)$, and $(m_x, m_v) = (255, 255)$, respectively. All solutions use the positivity-preserving algorithm. The above figures were produced by plotting the numerical solution at each of the 5×5 Gaussian quadrature points in each mesh element.

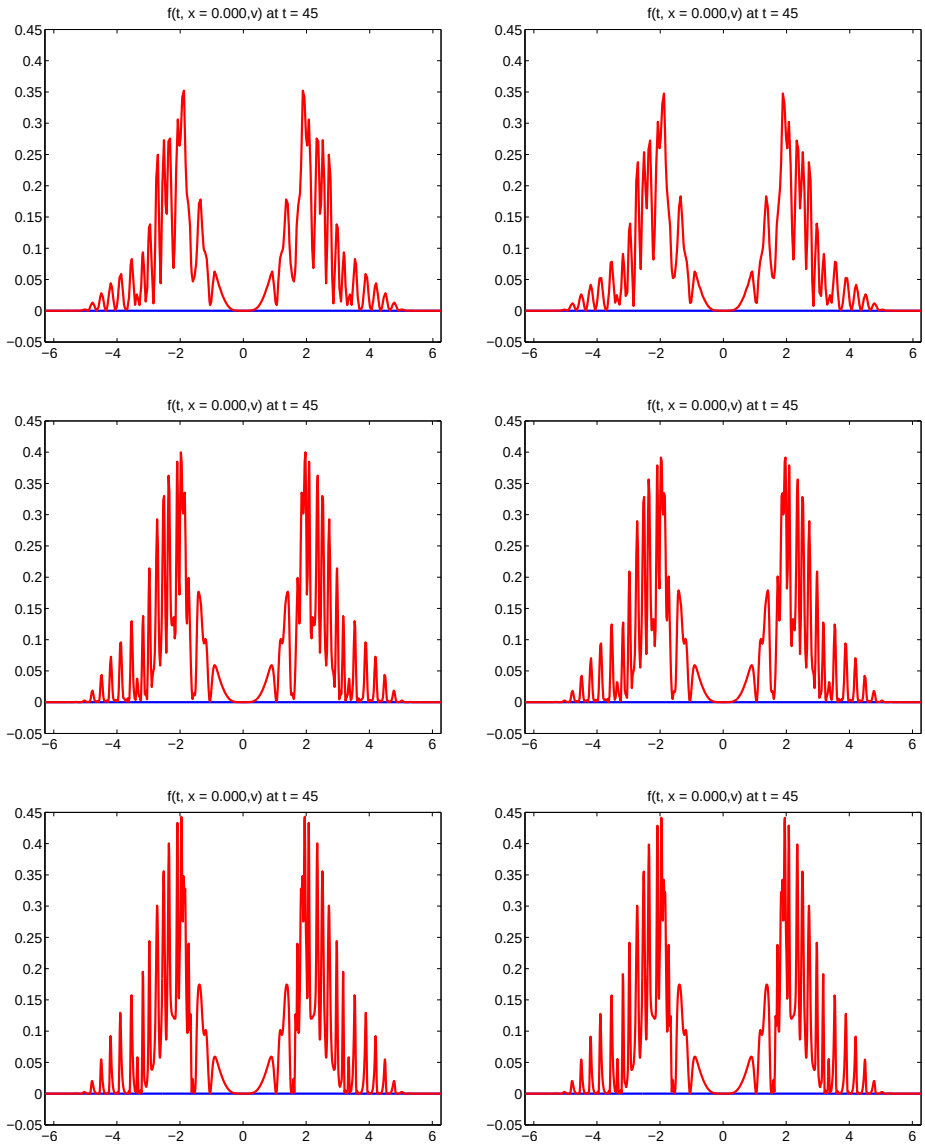


Figure 9: The two-stream instability problem. The panels in the left-hand column are results using the 2^{nd} order Strang splitting method. The panels in the right-hand column are results using the 4^{th} order splitting method. All simulations are 5^{th} order accurate in space. The mesh sizes for the first, second and third rows are $(m_x, m_v) = (65, 65)$, $(m_x, m_v) = (129, 129)$, and $(m_x, m_v) = (255, 255)$, respectively. All solutions use the positivity-preserving algorithm. Each figure above represents the numerical solution at $x = 0$ and use 5 Gaussian quadrature points for each cell in the v-coordinate. The above solutions were computed with an odd number of elements in each coordinate direction in order to easily obtain a slice of the solution at $x = 0$.

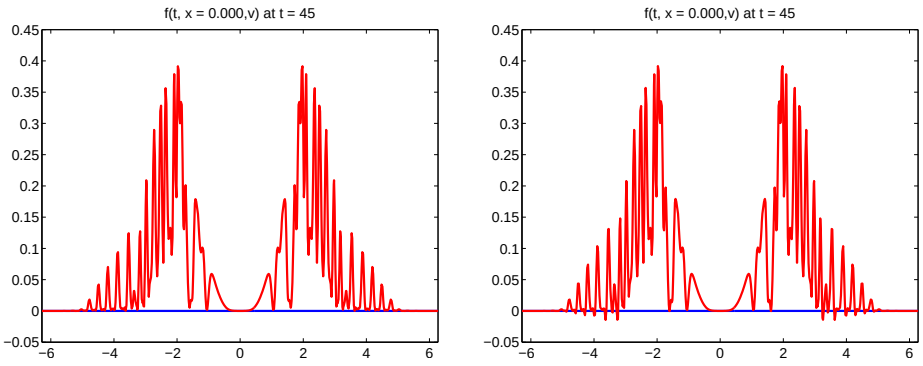


Figure 10: The two-stream instability problem. These panels demonstrate the effect of the positivity preserving-limiter. The result on the left is the 4^{th} order splitting algorithm with limiters turned on, and the result on the right hand side is the same algorithm with the limiters turned off. Both pictures represent a slice of the solution at $x = 0$ and final time $t = 45$. Both results represent a mesh of size $(m_x, m_v) = (129, 129)$ and are 5^{th} order accurate in space; an odd number of grid elements are used in order to easily obtain function values. We further note that $\min_i f(x_i, v_i) = -2.020 \times 10^{-2}$ for the solution without the limiter and $\min_i f(x_i, v_i) = 7.000 \times 10^{-12}$ for the limited solution, where the minimum is taken over all 5×5 Gaussian quadrature points over every mesh element.

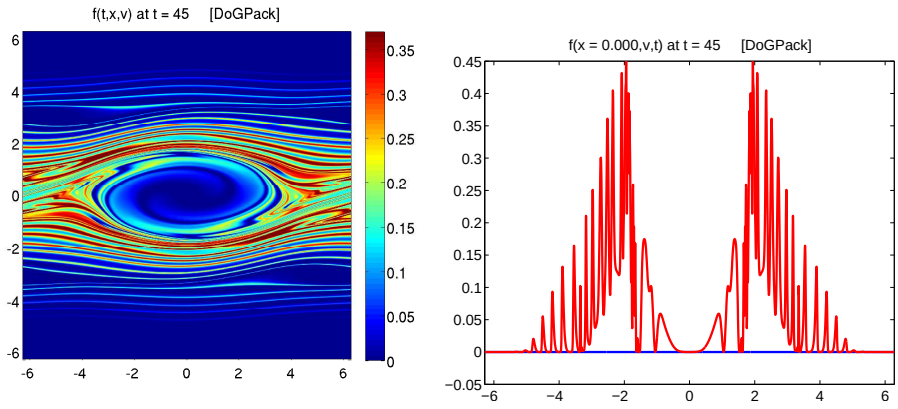


Figure 11: The two-stream instability problem. This figure represents a fine grid of size $(m_x, m_v) = (513, 513)$. This simulation is 5^{th} order in space, 2^{nd} order in time, and uses the positivity-preserving algorithm. Left: phase-space plot. Right: vertical slice of the solution at $x = 0.0$. An odd number of elements were used in order to easily obtain a solution at the center. The above figures were produced by plotting the numerical solution at each of the 5×5 Gaussian quadrature points in each mesh element.

We use the following initial distribution function

$$f(t = 0, x, v) = \Big(1 + \alpha \cos(kx)\Big) f_M(v); \qquad f_M(v) = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}}, \tag{3.129}$$

with $\alpha = 0.01$ and $k = 0.5$ on the domain $(x, v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. Because α is a small parameter, we expect to see results that closely agree with the linear theory, where the electric field decays exponentially. In Figure 13 we present this decay provided by

$$\log\left(\|E(t,\cdot)\|_{L_2}\right) := \log \sqrt{\int_{-2\pi}^{2\pi} |E(t,x)|^2 \; dx}$$

versus time for two different mesh sizes. Our computed decay rate matches the linear decay rate, $\gamma = -0.1533$. In Figure 14 we again plot the deviations of several quantities that are conserved by the continuous Vlasov-Poisson system from their initial values: $\|f\|_{L_1}, \|f\|_{L_2}$, total energy, and entropy. We again use the numerical approximation to these norms given by (A.3)–(A.6) in §A. Our results are comparable to what is reported

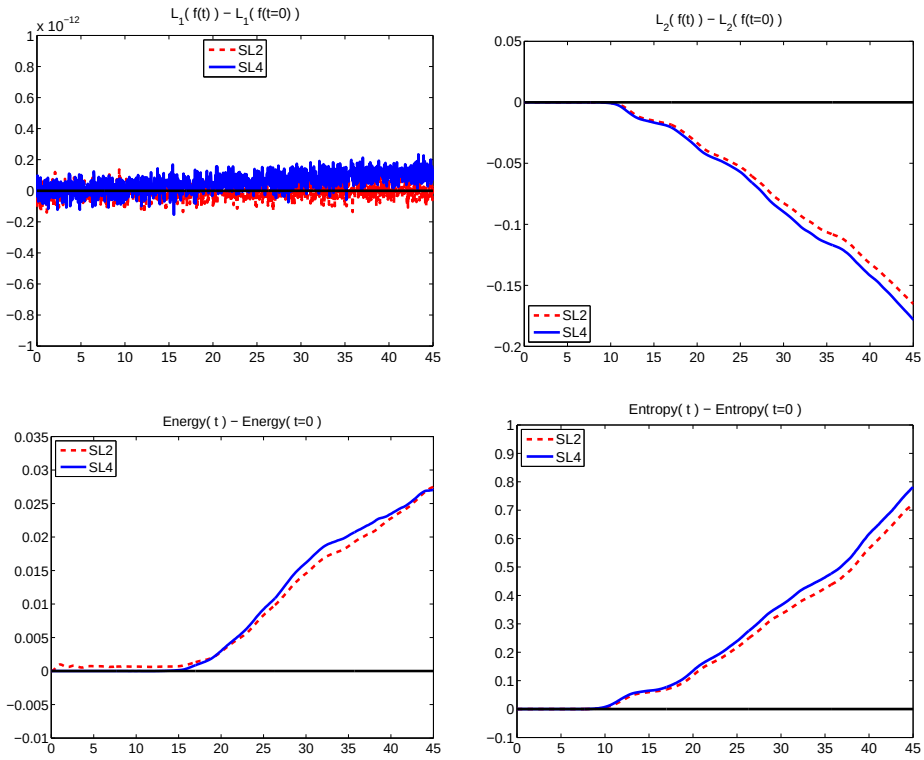


Figure 12: The two-stream instability problem. Shown in these panels are the deviations of the L_1 norm of f (top-left), L_2 norm of f (top-right), total energy (bottom-left), and total entropy (bottom-right) from their initial values. All simulations use a constant CFL number of 2.0, and are obtained from a mesh of size $(m_x, m_v) = (129, 129)$. The domain for this problem is $(x, v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. We note that at time $t = 0$, to within six digits of accuracy, the computed value for the L_1 and L_2 norms are, $\|f\|_{L_1} = 12.5664$, $\|f\|_{L_2} = 2.99102$, and the initial values for the total energy and total entropy are total energy$(t = 0) = 21.9911$ and total entropy$(t = 0) = 20.4161$. Each simulation is 5^{th} order accurate in space and is positivity preserving.

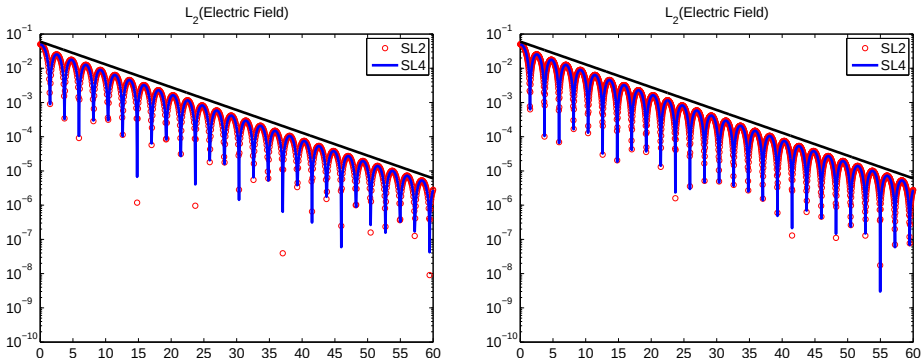


Figure 13: The weak Landau damping problem. Shown in the panels are semi-log plots of the L_2-norm of the electric field. Simulations use a constant CFL number of 2.0 and are 5^{th} order accurate in space. All simulations use the positivity-preserving limiter. The figure on the left represents a mesh of size $(m_x, m_v) = (64, 128)$ and the result on the right was represents a mesh of size $(m_x, m_v) = (128, 256)$, both on the domain $(x, v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. Both simulations match the theoretical decay rate of $\gamma = -0.1533$, and to demonstrate this we plot the line defined by $y = 0.06e^{\gamma t}$. One should note that the discrepancy in the two plots is due to the fact that twice as many time points in the second plot as the first one.

for example by Qiu and Christlieb [49].

3.5.5 Strong Landau damping

The initial condition is again given by (3.129), this time with $\alpha = 0.5$ and $k = 0.5$ on the domain $(x, v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. The time evolution of the distribution function is shown in the panels of Figure 15. These images are comparable to what is shown in Qiu and Christlieb [49], but we are again able to capture more fine scale structure with the discontinuous Galerkin approach.

A semi-log plot of the L_2-norm of the electric field is provided in Figure 16, and decay rates are computed by sampling the solution at data points. We find that the initial linear decay rate is approximately $\gamma_1 \approx -0.292$ which is close to the value of

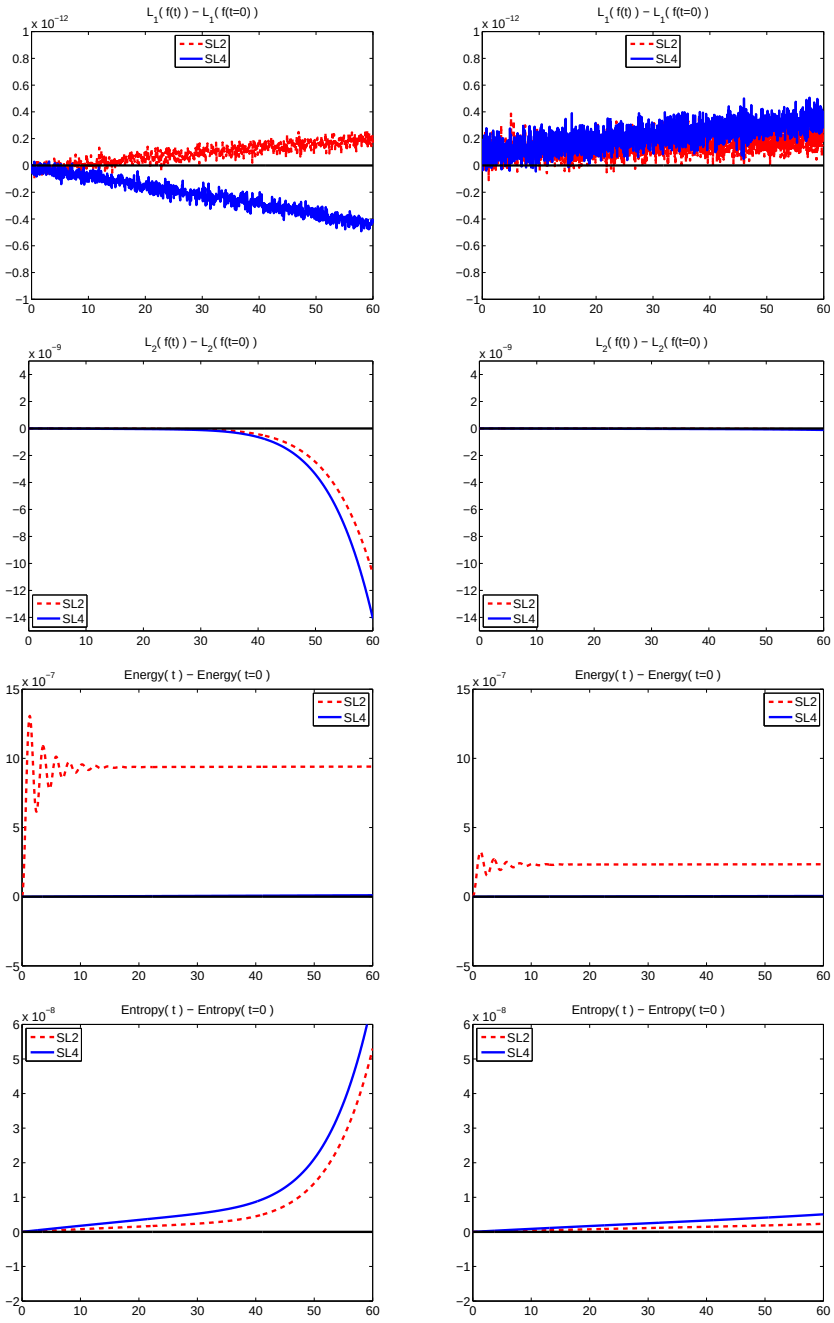


Figure 14: The weak Landau damping problem. Shown in the panels are the deviations of various quantities that are conserved by the exact equations from their initial values. All simulations use a constant CFL number of 2.0, are 5^{th} order accurate in space, and are positivity preserving. The mesh size for the left hand column is $(m_x, m_v) = (64, 128)$ and the mesh size for the right hand column is $(m_x, m_v) = (128, 256)$. We note that the largest deviation for total energy for the 4^{th} order algorithm is on the order of 10^{-10}. The initial conditions at time $t = 0$ were computed numerically for each run. For the coarser solution, to within eight digits of accuracy, the computed value for the L_1 and L_2 norms are, $\|f\|_{L_1} = 12.692034$, and $\|f\|_{L_2} = 3.616160$. The initial values for the total energy and total entropy are $\text{Energy}(t = 0) = 6.346017$, and $\text{Entropy}(t = 0) = 17.882927$.

-0.243 computed by Zaki et al. [65], closer still to the value of -0.281 computed by Cheng and Knorr [12], but much larger than the value of -0.126 computed by Heath et. al [36]. In this same figure we also estimate the growth rate due to particle trapping and find it to be approximately $\gamma_2 = 0.0815$; this number also differs from the value reported by Heath et al. [36]: $\gamma_2 = 0.0324$. The initial linear decay was computed by taking the maximum of the first two peaks located at $t \approx 2.45$ and $t \approx 4.54$. For the particle trapping growth regime, we sampled the maximum of the solution at the two peaks located at $t \approx 2.33$ and $t \approx 2.84$. We postulate that the difference between our computed growth rates and those of Heath et al. [36] stems from the fact that we are using piecewise quartic polynomials to represent the distribution function, while they are using only piecewise constants. This issue should be further investigated.

In Figure 17 we again plot the deviations of several quantities that are conserved by the continuous Vlasov-Poisson system from their initial values: $\|f\|_{L_1}, \|f\|_{L_2}$, total energy, and entropy. In particular, we use the numerical approximations to (1.49)–(1.52) as given by equations (A.3)–(A.6) in §A. Our results are comparable to what is reported for example by Qiu and Christlieb [49].

Finally, we note that in both our weak and strong Landau damping simulations we made $V_{\max} = 2\pi$ instead of the more commonly used value of $V_{\max} = 5$. The reason for this is that we noticed that between roughly $v = 5$ and $v = 6$ the distribution function $f(t, x, v)$ still had a non-negligible amplitude on the order of about 10^{-6}; the precise behavior of strong and weak Landau damping in this region is shown in Figure 18. Therefore, truncating at $V_{\max} = 5$ caused additional errors when tracking various conserved quantities; we found improvements in these errors when taking $V_{\max} = 2\pi$.

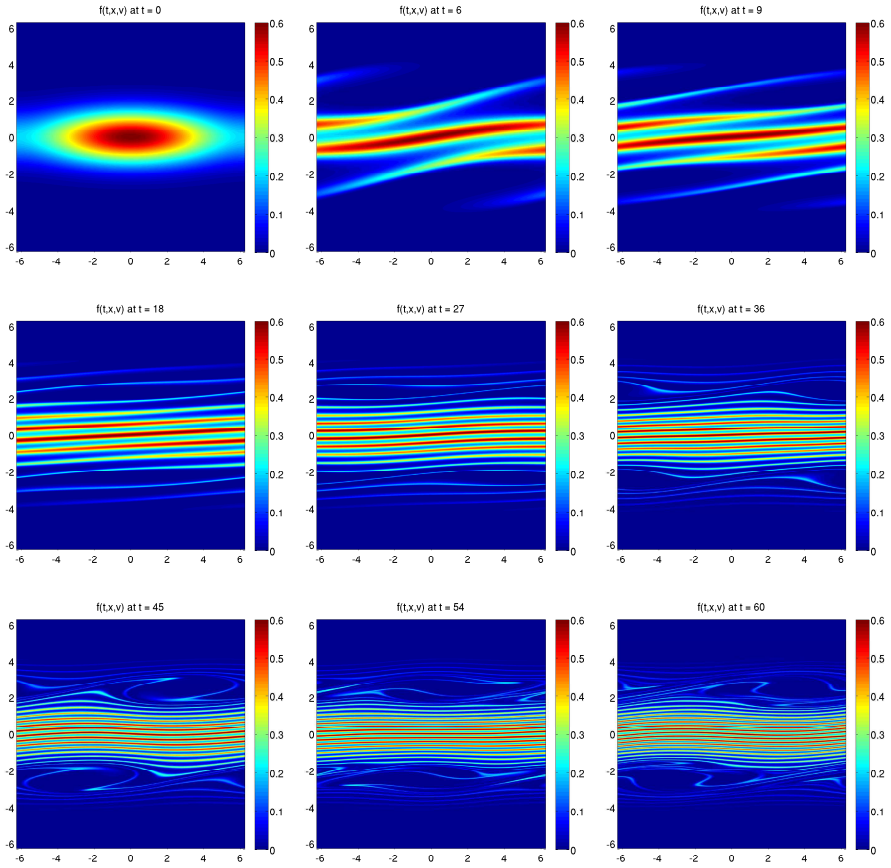


Figure 15: The strong Landau damping problem. Shown in the panels are the distribution function at various points in time. This simulation was run with a constant CFL number of 2.0 on a mesh of size $(m_x, m_v) = (128, 256)$ using 5^{th} order accuracy in space and the positivity-preserving limiters. It is clear from these plots that the high-order discontinuous Galerkin method is able to capture much of the fine-scale structure for the solution.

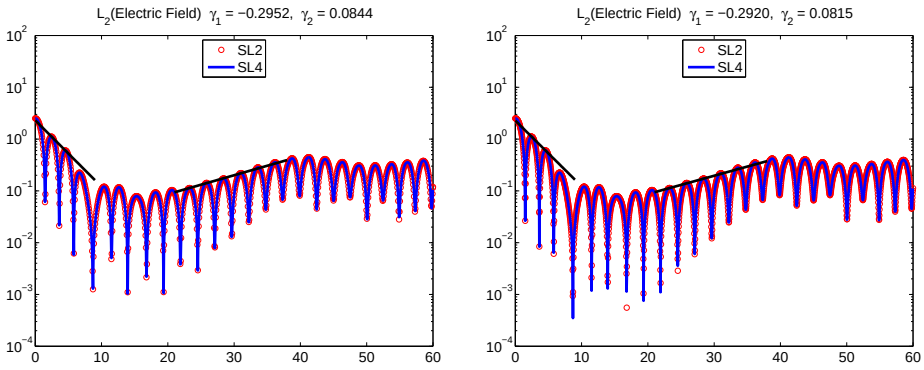


Figure 16: The strong Landau damping problem. Shown in these panels are semi-log plots of the L_2 norm of the electric field with two different resolutions; the mesh size for the the figure on the left is $(m_x, m_v) = (64, 128)$ and the figure on the right is $(m_x, m_v) = (128, 256)$. Both simulations use the positivity preserving limiter, are 5^{th} order accuracy in space and use a constant CFL number of 2.0. In each panel, γ_1 refers to the slope of the initial decay, and γ_2 refers to the growth rate between times $t = 20$ and $t = 40$.

3.5.6 Plasma Sheath

So far, all of the presented examples have periodic boundary conditions in x. However, the SLDG method can easily be adapted to accommodate non-periodic boundary conditions. One classical 1D example that is not periodic in x is the so-called plasma sheath problem. The distribution function in dimensional units, $\tilde{f}\left(\tilde{t}, \tilde{x}, \tilde{v}\right)$, represents the PDF for electrons inside the region: $0 \leq \tilde{x} \leq L$. We will assume that no electrons can enter this domain, meaning that we take zero-inflow boundary condition:

$$\tilde{f}(\tilde{t}, \tilde{x} = 0, \tilde{v} \geq 0) = \tilde{f}(\tilde{t}, \tilde{x} = L, \tilde{v} \leq 0) = 0. \tag{3.130}$$

The electric potential is fixed to a constant value at $x = 0$ and $x = L$, which, without loss of generality, can be taken to be zero: $\tilde{\phi}(\tilde{t}, \tilde{x} = 0) = \tilde{\phi}(\tilde{t}, \tilde{x} = L) = 0$.

The initial condition, written, at least for the moment, in dimensional form, is a

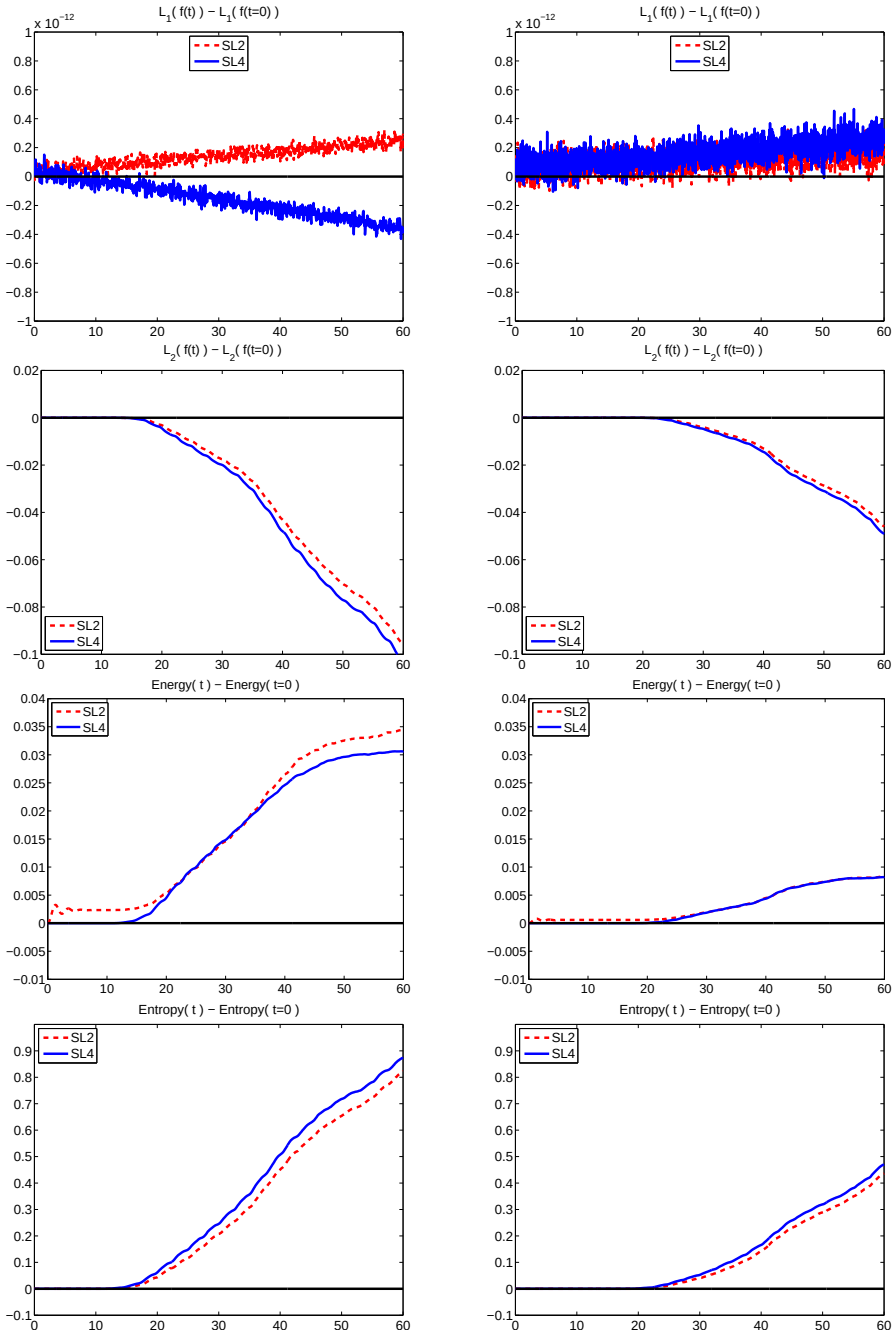


Figure 17: The strong Landau damping problem. Simulation results for the L_1 norm (first row), L_2 norm (second row), energy (third row), and entropy (bottom row) for strong Landau damping. All simulations use a constant CFL number of 2.0 and are 5^{th} order accurate in space. The mesh size for the left column is $(m_x, m_v) = (64, 128)$ and the mesh size for the right column is $(m_x, m_v) = (128, 256)$. The initial conditions at time $t = 0$ were computed numerically for each run. For the coarser solution, to within six digits of accuracy, the computed value for the L_1 and L_2 norms are, $\|f\|_{L_1} = 12.5664$, and $\|f\|_{L_2} = 3.98802$. The initial values for the total energy and total entropy are $\text{Energy}(t = 0) = 9.42478$, and $\text{Entropy}(t = 0) = 17.0186$.

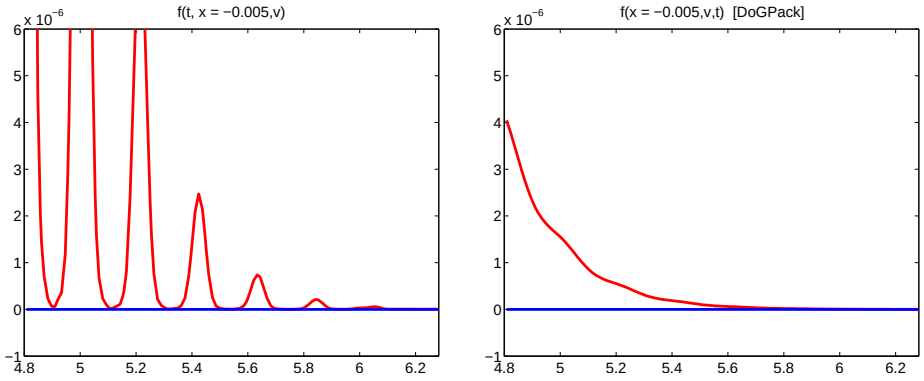


Figure 18: A comparison of vertical slices for strong landau damping (left) and weak Landau damping (right) at time $t = 60$. The simulations for each panel use a grid resolution of $(m_x, m_v) = (128, 256)$, each are 5^{th} order accuracy in space, and each use the positivity-preserving limiter. The CFL number for each simulation is 2.0. We note that the solution is non-zero for $|v| > 5$ in both cases, although there is much more activity in the case of strong Landau damping. The plots suggests that the commonly used maximum velocity of $|v| = 5$ should be increased in order to get better accuracy in conservation of the quantities (A.3)–(A.6).

stationary Maxwellian distribution with mass density $\tilde{\rho}_0$ (units of meters^{-1}) and temperature $\tilde{\theta}_0$ (units of Kelvin):

$$\tilde{f}(0, \tilde{x}, \tilde{v}) = \frac{\tilde{\rho}_0}{\sqrt{2\pi m k \tilde{\theta}_0}} \exp\left(-\frac{m\tilde{v}^2}{2k\tilde{\theta}_0}\right). \tag{3.131}$$

In order to non-dimensionalize this initial condition, we introduce the following scaling parameter:

$$\tilde{\theta} = \Theta_0 \, \theta. \tag{3.132}$$

After substituting this, along with the dimensional variables from §1.2.2, into the initial

condition, we arrive at the following:

$$\begin{aligned} f(0,x,v) &= \left(\frac{qL}{\sqrt{\epsilon_0 mN}}\right) \frac{mN\rho_0}{\sqrt{2\pi mk\Theta_0\theta_0}} \exp\left(\frac{-mL^2v^2}{2T^2k\Theta_0\theta_0}\right) \\ &= \sqrt{\frac{Nq^2L^2}{\epsilon_0 k\Theta_0}} \frac{\rho_0}{\sqrt{2\pi\theta_0}} \exp\left(-\left(\frac{mL^2}{T^2k\Theta_0}\right)\frac{v^2}{2\theta_0}\right) \\ &= \sqrt{\frac{Nq^2L^2}{\epsilon_0 k\Theta_0}} \frac{\rho_0}{\sqrt{2\pi\theta_0}} \exp\left(-\left(\frac{Nq^2L^2}{\epsilon_0 k\Theta_0}\right)\frac{v^2}{2\theta_0}\right). \end{aligned} \tag{3.133}$$

We choose Θ_0 to make the constants in the parenthesis equal to one:

$$\frac{Nq^2L^2}{\epsilon_0 k\Theta_0} = 1 \quad \Longrightarrow \quad \Theta_0 = \frac{Nq^2L^2}{\epsilon_0 k}. \tag{3.134}$$

This yields a Maxwellian distribution that has been non-dimensionalized:

$$f(0,x,v) = \frac{\rho_0}{\sqrt{2\pi\theta_0}} \exp\left(-\frac{v^2}{2\theta_0}\right). \tag{3.135}$$

The two free parameters are given by

$$\rho_0 = \left(\frac{1}{mN}\right)\tilde{\rho}_0, \quad \theta_0 = \left(\frac{\epsilon_0 k}{Nq^2L^2}\right)\tilde{\theta}_0,$$

Recall that in SI units, the Boltzmann constant, k, the electron mass, m, the electron charge, q, and the permittivity of free space, ϵ_0, are given by

$$\begin{aligned} k &= 1.38064881 \times 10^{-23}\, \frac{\mathrm{kg\,m^2}}{\mathrm{K\ sec^2}}, \\ m &= 9.10938188 \times 10^{-31}\, \mathrm{kg}, \\ q &= 1.60217646 \times 10^{-19}\, \mathrm{C}, \\ \epsilon_0 &= 8.85418782 \times 10^{-12}\, \frac{\mathrm{sec^2\,C^2}}{\mathrm{kg\,m}}. \end{aligned}$$

For the simulation results presented here, we set

$$N = 10^{13}\,\mathrm{m}^{-1},$$

$$L = 0.1\,\mathrm{m},$$

$$\tilde{\rho}_0 = mN = 9.10938188 \times 10^{-18}\,\frac{\mathrm{kg}}{\mathrm{m}},$$

$$\tilde{\theta}_0 = 1\,\frac{\mathrm{eV}}{k} = 1.160451897 \times 10^{4}\,\mathrm{K}.$$

The net results for all of the scaling parameters then becomes:

$$F = 5.605424055 \times 10^{5}\,\frac{\mathrm{sec}}{\mathrm{m}^2},$$

$$T = 5.605424055 \times 10^{-9}\ \mathrm{sec},$$

$$E_0 = 1.809512620 \times 10^{4}\,\frac{\mathrm{Volts}}{\mathrm{m}},$$

$$\Phi_0 = 1.809512620 \times 10^{3}\,\mathrm{Volts},$$

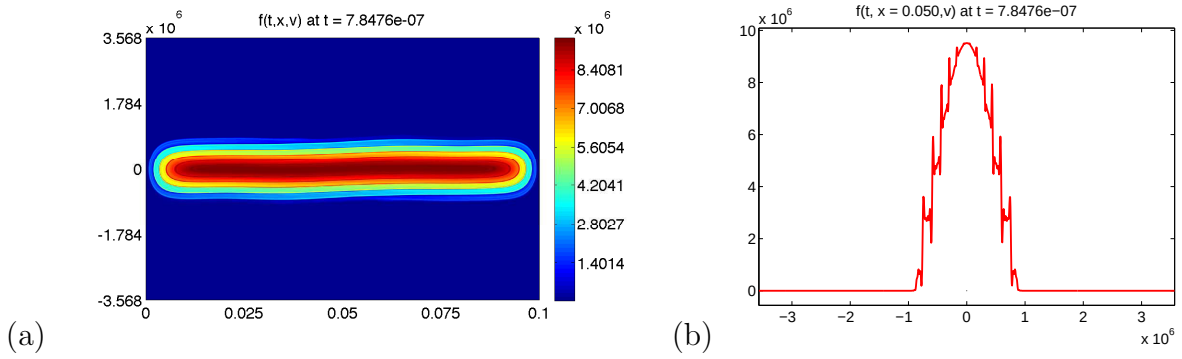
$$\Theta_0 = 2.099852260 \times 10^{7}\,\mathrm{K},$$

$$\rho_0 = 1,$$

$$\theta_0 = 5.526350206 \times 10^{-4}.$$

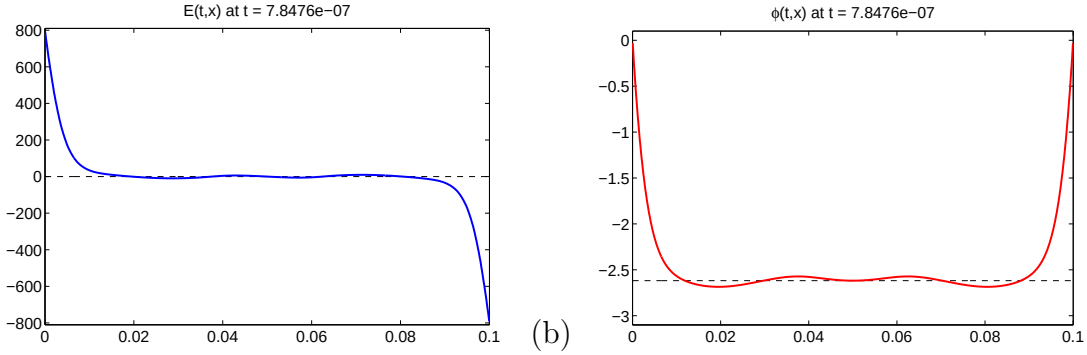
We note that our setup is identical to that presented in Christlieb et al. [16].

We solve the above initial and boundary conditions on a 256×256 mesh on $(x, v) \in [0, 1] \times [-0.2, 0.2]$ out to $t = 140$. The solution at the final time is shown in Figures 19 and 20. In Figure 19 we show the full distribution function in phase space at the final time in the simulation, as well as a slice through the data at $x = 0.5$ (i.e., the midpoint in the simulation domain). In Figure 20 we show the electric field and electric potential at the final time. In all plots the quantities are reported in dimensional form. Finally, we note that our results are consistent with other results in the literature, such as those reported in Christlieb et al. [16].



(a) (b)

Figure 19: Plasma sheath problem. Panel (a) shows a plot of the distribution function in phase space. Panel (b) shows a vertical slice of the distribution function through $x = 0.5$ (i.e., $\tilde{x} = 0.05\,\mathrm{m}$). Note that we are plotting the dimensional quantities.



(a) (b)

Figure 20: Plasma sheath problem. Panel (a) shows the electric field. Panel (b) shows the electric potential. Note that we are plotting the dimensional quantities.

Chapter 4

Hybrid Semi-Lagrangian Methods for 1+1 Vlasov-Poisson

In this chapter we describe a hybrid semi-Lagrangian discontinuous Galerkin (HSLDG) approach for solving the $(1+1)$D Vlasov-Poisson equation on Cartesian meshes. This method is not as efficient as the full semi-Lagrangian method described in Chapter 3, but the purpose of implementing and running this method here is that it creates the correct building blocks for constructing a (2+2)D Vlasov solver that will operate on unstructured grids in physical space. We describe the full details of this extension to unstructured grids in Chapter 5.

4.1 Extensions of SLDG to Higher Dimensions

The full SLDG scheme from Chapter 3 can certainly be extended to higher dimensions. The simplest extension would come from dimensional splitting together with the operator split technology discussed in §3.3.3. However, the SLDG method in its current form would be restricted to operating on Cartesian grids. Extensions of SLDG methods to unstructured grids in 2D has proven to be somewhat difficult, where the primary issue is that exact integration of characteristics is difficult to come by, yet is necessary to obtain

stability.

We point out that Restelli, Bonaventura, and Sacco [51] have demonstrated that it is certainly possible to run SLDG methods on unstructured grids, however their method as described is unstable for structured grids. Qiu and Shu [43] demonstrated how to adapt their method and make it stable for structured grids, however, their method of extending 1D problems to 2D problems is identical to this work, and hence is currently limited to working on Cartesian grids. Besse and Sönnendrucker [7] present a backwards semi-Lagrangian scheme for unstructured grids, but they do not use a DG representation for their solution, and we believe there is mileage to gain from switching to DG.

Our goal is to create a provably stable method that can handle unstructured physical grids with high order accuracy, and is positivity preserving. In this chapter, we lay the foundations for a hybrid, semi-Lagrangian discontinuous Galerkin (HSLDG) method which will accomplish all of the above. The correct starting point for demonstrating its efficacy is a (1+1)D Vlasov problem, and that is the subject of this chapter.

4.2 Hybrid semi-Lagrangian Scheme

Our hybrid method is a blend of classical Runge-Kutta discontinuous Galerkin (RKDG) methods (see Chapter 2) together with the semi-Lagrangian discontinuous Galerkin (SLDG) methods presented in Chapter 3. The operator splitting techniques presented in §3.3.3 allow us to easily treat the space and velocity problems via different numerical methods. The essential idea of the hybrid method is the following: use semi-Lagrangian methods for advection over the velocity coordinates, and use Runge-Kutta time stepping for advection over physical coordinates.

Resorting to explicit Runge-Kutta time stepping wakens unfortunate CFL limitations, and in practice, the results of this hybrid method are nowhere near as efficient as a full semi-Lagrangian method for structured grids. However, we reiterate that a simple, (1+1)D problem provides the necessary framework for extensions to higher dimensions, and in particular, to unstructured grids. In order to mitigate a globally restrictive CFL condition, we utilize sub-cycling for the RKDG part of the problem which allows us to take much larger time steps than would normally be allowed. The details of the proposed sub-cycling method are presented in §4.2.2.

4.2.1 Description of Hybrid Method

Identical to the methods presented in the previous chapter, our hybrid method for the Vlasov-Poisson system utilizes the exact same operator split technology (c.f. §3.3.3). To remind the reader, we note that each stage in the split VP solver requires solving two sub-problems:

$$\text{Problem } \mathcal{A}\text{:} \quad f_{,t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f = 0, \tag{4.1}$$

$$\text{Problem } \mathcal{B}\text{:} \quad f_{,t} + \mathbf{E}(t, \mathbf{x}) \cdot \nabla_{\mathbf{v}} f = 0. \tag{4.2}$$

The full SLDG solver presented in Chapter 3 uses semi-Lagrangian time stepping for *both* sub-problems. The hybrid method proposes to replace the solver used for "Problem $\mathcal{A}$" with a classical RKDG solver in lieu of semi-Lagrangian time stepping. The operator split technology for Strang splitting as well as Yoshida splitting remains intact, but each spot where we advect over the spatial coordinates, we choose to apply an RKDG method.

For clarity, in Table 4.2.1 we repeat the 2^{nd} order Cheng and Knorr [12] split method, but this time we specifically indicate which method is applied to each sub-problem.

Likewise, the fourth order split hybrid method is identical to that presented in Algorithm 3.3, but this time sub-problems are treated differently, that is, we replace the semi-Lagrangian solver used in stages 3, 5, 7 and 9 with an RKDG solver. The high order Lax-Wendroff expansion of the electric field from section §3.4.2 remains in place. Again, we note that for either split method, high-order or second-order, we only require a single Poisson solve per time step.

Algorithm 4.1 A 2^{nd} order hybrid operator split algorithm.

1. $\frac{1}{2}\Delta t$ RKDG step on $f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} = 0$.
2. Solve $-\nabla^2 \phi = \rho^{n+\frac{1}{2}} - \rho_0$, and compute $\mathbf{E}^{n+\frac{1}{2}}(t, \mathbf{x})$.
3. Δt SLDG step on $f_{,t} + \mathbf{E}^{n+\frac{1}{2}} \cdot f_{,\mathbf{v}} = 0$.
4. $\frac{1}{2}\Delta t$ RKDG step on $f_{,t} + \mathbf{v} \cdot f_{,\mathbf{x}} = 0$.

In order to complete the description of the method, we still need to address how to run a RKDG solver for a quasi-1D problem (3.18). Here, we closely follow algorithm 3.2, by treating the quasi-1D advection equation as a set of many 1D problems. In Algorithm 4.2 we present the proposed hybrid method for advancing $f_{,t} + u(v) f_{,x} = 0$ forward in time using RKDG technology. This algorithm is identical to that presented in Algorithm 3.2, with on major exception: Step 2 of the quasi-1D solver is replaced with a sub-cycled RKDG update.

In both algorithms, Step 1 produces a list of $m_v \cdot M$ problems, where m_v is the number of grid elements in the v direction, and M denotes the order of accuracy. The equation indexed by j and k is defined by the 1D advection equation:

$$f_{,t} + v_{jk} f_{,x} = 0.$$

An RKDG solver produces new moments, $F^{(\ell)}_{1\mathrm{D},ijk}(t^{n+1})$ at time level t^{n+1} using initial conditions prescribed by $F^{(\ell)}_{1\mathrm{D},ijk}(t^n)$. This completes the description of the hybrid method for solving each quasi-1D problem because the complete details for advancing 1D problems was presented in Chapter 2 which involves a simple method of lines (MOL) formulation plus a high order Runge-Kutta integrator. Here, we use an upwind Riemann solver for each interface, which in the case of a scalar equation, is identical to the HLLE approximate Riemman solver.

Algorithm 4.2 Hybrid DG Method for Quasi-1D Equations

0. Initial Projection: Construct all of the $2D$ moments:

$$F^{(\ell),n}_{ij} \longleftarrow f(t=0,x,v)$$

Here, the method is identical to the SLDG method.

1. Convert to 1D Problems: For each row, construct 1D moments:

$$F^{(\ell)}_{1\mathrm{D},ijk}(t^n) \longleftarrow F^{(\ell),n}_{ij}$$

Here, the method is identical to the SLDG method.

2. Evolve 1D Problems: Here, we use sub-cycled RKDG time stepping in place of semi-Lagrangian time stepping.

$$F^{(\ell)}_{1\mathrm{D},ijk}(t^{n+1}) \longleftarrow F^{(\ell)}_{1\mathrm{D},ijk}(t^n)$$

This is the new part. These 1D moments form the coefficients for the initial projection (2.5) on the equation $f_{,t} + v_{jk} f_{,x} = 0$. The spatial discretization happens via the weak form (2.6).

3. Integrate 1D Problems: Integrate the 1D coefficients up to 2D coefficients:

$$F^{(\ell),n+1}_{ij} \longleftarrow F^{(\ell)}_{1\mathrm{D},ijk}(t^{n+1})$$

Here, the method is identical to the SLDG method.

We note that the order in which the splitting takes place is the same as the original Cheng & Knorr method presented in Algorithm 3.1. This may seem counter intuitive given that the expensive part of the problem is now advection over the velocity coordinates, and so it sounds reasonable to swap the roles of the two operators, $\mathcal{A}$ and $\mathcal{B}$ in order to reduce the number of advection equations over the velocity variables. However, stages 1 and 4 only require a time step of length $\Delta t/2$, and so solving two of these problems is essentially identical, in terms of computational cost, to solving a single problem with time step of length Δt. In terms of high-order splitting, more care needs to be considered in terms of efficiency, especially because we have lost part of what semi-Lagrangian schemes gave us: unconditional stability. In the case of Yoshida splitting (c.f. §3.3.3) the time steps required for operators $\mathcal{A}$ and $\mathcal{B}$ are approximately:

$$\begin{aligned} \text{Operator } \mathcal{A} \text{ time steps:} &\quad \approx \quad [0.6756\Delta t, -0.1756\Delta t, -0.1756\Delta t, 0.6756\Delta t]; \\ \text{Operator } \mathcal{B} \text{ time steps:} &\quad \approx \quad [1.3512\Delta t, -1.7024\Delta t, 1.3512\Delta t]. \end{aligned}$$

The sum of each of these time steps is obviously Δt, but more interesting is that from the viewpoint of an explicit RK integrator, one is much more expensive than the other. Given that there are negative time steps involved, the proper way to measure the expense incurred by running an explicit integrator here is to sum the absolute values of every time step for each operator. In terms of Yoshida splitting, this yields:

$$\begin{aligned} \text{Total cost of operator } \mathcal{A} &\quad \approx \quad 1.7024\Delta t. \\ \text{Total cost of operator } \mathcal{B} &\quad \approx \quad 4.4048\Delta t. \end{aligned}$$

Therefore, with Yoshida splitting, we actually save computational cost by running the RKDG method on the operator that incurs *more* stages, which saves us a factor of

roughly 2.5. Ultimately, the total cost of running a high-order operator split method will always be larger than Δt. This is because all high-order operator split methods require the use of negative time steps. One should tailor the choice of which operator to apply to the explicit part based on total cost.

We note that the CFL condition for a monolithic update would normally be given by

$$\Delta t \leq \text{CFL}\frac{\Delta x}{|v_{\max}|}, \tag{4.3}$$

but since each of these problems are completely decoupled, we can utilize sub-cycling in order to reduce the number of expensive RK solves. That is, the CFL restriction for the m^{th} equation, is simply

$$\Delta t \sim \text{CFL}\frac{\Delta x}{|v_m|} \gg \text{CFL}\frac{\Delta x}{|v_{\max}|}, \tag{4.4}$$

which, for the Vlasov-Poisson system, is essentially always much larger than the single row that is responsible for the largest velocity, and hence the smallest time step.

4.2.2 Sub-Cycling for the Hybrid Scheme

Sub-cycling can be accomplished by adaptively choosing a local CFL condition. Let Δt be the macro time step desired for stepping $f_{,t} + \mathbf{v} f_{,\mathbf{x}} = 0$ forward in time. Because there are a total of M equations per row, and a total of m_v rows, there are a total of $M m_v$ equations, all in 1D that need to be evolved. Operator splitting completely decouples each of these N equations. Suppose that the advection speed for the m^{th} equation is given by $|v_m| \ll v_{\max}$. (Here, in order to cut down on the number of indices, we use m to refer to single a row which is indexed by both j and k.) We now choose a micro-step h for this equation. Suppose that the CFL restriction is given by ν, that is, the micro

step must obey $h \leq \nu \frac{\Delta x}{|v_m|}$. One such choice that accomplishes this while at the same time maximizing h is by setting

$$h = \Delta t \Big/ \max\left\{1, \left\lceil \frac{|v_m|\Delta t}{\nu \Delta x} \right\rceil \right\}. \tag{4.5}$$

Here $\lceil \cdot \rceil$ denotes the *ceiling* operation.[1] In the case where $v_m = 0$, or is small, this simply sets $h = \Delta t$, which is the largest possible time step available. For large velocities, this method takes exactly

$$N = \left\lceil \frac{|v_m|\Delta t}{\nu \Delta x} \right\rceil \tag{4.6}$$

steps of length h on the m^{th} equation. To see why this choice of h doesn't violate the local CFL condition, first observe that

$$\max\left\{1, \left\lceil \frac{|v_m|\Delta t}{\nu \Delta x} \right\rceil \right\} \geq \left\lceil \frac{|v_m|\Delta t}{\nu \Delta x} \right\rceil \geq \frac{|v_m|\Delta t}{\nu \Delta x},$$

and hence,

$$h = \Delta t \Big/ \max\left\{1, \left\lceil \frac{|v_m|\Delta t}{\nu \Delta x} \right\rceil \right\} \leq \Delta t \Big/ \frac{|v_m|\Delta t}{\nu \Delta x} = \frac{\nu \Delta x}{|v_m|}.$$

After taking N time steps of length h, we then have a collection of 1D moments at time level $t^{n+1} = t^n + \Delta t$, which can then be integrated up into 2D weights. In practice, we don't choose a different h for each equation. Rather, we choose h based on the largest speed present in the j^{th} row. This makes parallelization somewhat easier, given that after updating all M equations for the j^{th} row, one needs to integrate these equations up to the 2D moments. If we make row assignments to each thread, then a single thread can advance, integrate and therefore update 2D moments, independent of every other thread.

[1] This function takes a real input and rounds up to the largest integer that is greater than or equal to the input.

In some sense, sub-cycling accomplishes exactly what it's meant to do. That is, the macro time-step, Δt, can be arbitrarily large, and hence this update is 'unconditionally stable', just like a semi-Lagrangian update would be. However, this isn't the whole truth, because the sub-problems need to obey a local CFL condition, yet the same idea is still captured in spirit. We note that as an alternative to sub-cycling, one could proceed with an implicit method, which may be the subject of future work. However, it's not clear that an implicit method, which in principle should provide unconditional stability, will actually be a winning venture. The reasons are twofold. First, large CFL numbers produce matrices with increasing condition numbers. Hence, the number of iterations required to take a large time step increases with the CFL number. Second, there do not appear to be many high-order implicit methods that are able to accommodate purely hyperbolic problems.

4.2.3 Summary of the Hybrid (HSLDG) Method

This completes the description of the hybrid semi-Lagrangian method. The details concerning how one proceeds via a RKDG method on 1D problems is described in §2.1, and the details concerning transferring moments between 1D and 2D problems is outlined in §3.3.2. The key difference between the full SLDG method and this hybrid method is simply how the 1D problems are advanced forward in time. There, the only difference is the fact that advancing the unknown variables on physical space is replaced with a sub-cycled RKDG method. One advantage of RKDG methods is that they can operate on unstructured grids, which is described in the following chapter. At present, we now show results of this hybrid method on (1+1)D Vlasov problems. The full method has

been implemented in DogPack [52], and the code has been written with shared memory OpemMP threads for each of the split directions. Specifically, threads are created before each quasi-1D step, and merged afterwards. This amounts to effectively chopping the domain into many rows for advection in the x-direction, and many columns for advection in the v-direction.

4.3 Numerical Examples

4.3.1 Linear advection

Here, we repeat the problem already presented in §3.5.1, but this time with the hybrid method in place. The problem is a linear advection under a divergence-free velocity field:

$$q_{,t} + \mathbf{u} \cdot \nabla q = 0. \tag{4.7}$$

We take the computational domain to be $[0,1] \times [0,1]$ and the velocity field to be solid body rotation around $(0.5, 0.5)$:

$$\mathbf{u} = (u(y), v(x)) = \left(\pi\left(2y-1\right), \pi\left(1-2x\right)\right)^T. \tag{4.8}$$

The initial condition is taken to be a smooth, compactly supported bump that is centered at $(x_0, y_0) = (0.4, 0.5)$:

$$q(0,x,y) = \begin{cases} \cos^6\left(\frac{5\pi}{3}r\right) & \text{if} \quad r \leq 0.3, \\ 0 & \text{otherwise,} \end{cases} \tag{4.9}$$

where

$$r = \sqrt{(x-x_0)^2 + (y-y_0)^2}. \tag{4.10}$$

Mesh	**HSL2 Error**	$\log_2$**(Ratio)**	**HSL4 Error**	$\log_2$**(Ratio)**
10^2	3.380×10^{-1}	–	6.984×10^{-1}	–
20^2	7.635×10^{-2}	2.146	3.440×10^{-2}	4.344
40^2	1.789×10^{-2}	2.093	1.943×10^{-3}	4.146
80^2	4.308×10^{-3}	2.054	1.173×10^{-4}	4.050
160^2	1.075×10^{-3}	2.002	7.181×10^{-6}	4.029
320^2	2.664×10^{-4}	2.013	4.455×10^{-7}	4.011

Table 5: Convergence study for the linear advection equation. Shown are the relative errors computed via (A.14) at time $t = 1$. All calculations were done with 4^{th} order accuracy in space and a CFL number of 5.19, where CFL $:= \Delta t \max\{\max_y |u(y)|/\Delta x, \max_x |v(x)|/\Delta y\}$. Each advection equation for the RK time stepping was run with a local maximum CFL number of 0.10. HSL2 refers to the Strang split hybrid semi-Lagrangian scheme and HSL4 to the fourth-order split hybrid semi-Lagrangian scheme.

This problem, just as the Vlasov-Poisson system, is solved via operator splitting on the two operators:

$$\text{Problem } \mathcal{A}\text{:} \quad q_{,t} + u(y)\, q_{,x} = 0, \tag{4.11}$$

$$\text{Problem } \mathcal{B}\text{:} \quad q_{,t} + v(x)\, q_{,y} = 0. \tag{4.12}$$

We run the initial condition out to time $t = 1$, at which point it should return to its initial state. The errors are computed using the relative L_2 errors defined by equation (A.14) with $M = 5$ and varying $\Delta x = \Delta y$. See A.2 for more details. Convergence studies with Strang and the fourth-order operator splitting results for the proposed hybrid method are shown in Table 5. For time stepping on the 1D RKDG problems, we used classical RK4 with a local CFL number of 0.1.

4.3.2 A forced problem: verifying order of accuracy

The linear advection equation from §4.3.1 indicates we get the correct convergence rates for a problem where the coefficients are non-constant, and it has the added benefit that two operators do not commute, so we are able to observe a splitting error. However, a stronger test is to run the same problem already presented in §4.3.2, that exercises the expansion of the electric field, as well as the full RK time stepping on the sub-problems. For convenience, we repeat the statement of the problem here:

$$f_{,t} + v f_{,x} + E f_{,v} = \psi(t, x, v), \tag{4.13}$$

$$E_{,x} = \int_{-\infty}^{\infty} f(t, x, v)\, dv - \sqrt{\pi}, \tag{4.14}$$

on $(t, x, v) \in [0, \infty) \times [-\pi, \pi] \times (-\infty, \infty)$ with periodic boundary conditions in x. We take the following source term:

$$\begin{aligned}\psi(t, x, v) = \frac{1}{2}\sin(2x - 2\pi t) e^{-\frac{1}{4}(4v-1)^2} \Big\{ &\left(2\sqrt{\pi} + 1\right)\left(4v - 2\sqrt{\pi}\right) \\ &- \sqrt{\pi}\,(4v - 1)\cos(2x - 2\pi t)\Big\}.\end{aligned} \tag{4.15}$$

The exact solution in this case is

$$f(t, x, v) = \left\{2 - \cos\left(2x - 2\pi t\right)\right\} e^{-\frac{1}{4}(4v-1)^2}, \tag{4.16}$$

$$E(t, x) = -\frac{\sqrt{\pi}}{4}\,\sin\left(2x - 2\pi t\right). \tag{4.17}$$

The numerical scheme for this forced problem is the same as the one described in §4.2 with two minor modifications. First, the two operators in the operator split scheme are

$$\text{Problem } \mathcal{A}\text{:} \quad f_{,t} + v\, f_{,x} = \psi(t, x, v), \tag{4.18}$$

$$\text{Problem } \mathcal{B}\text{:} \quad f_{,t} + E(t, x)\, f_{,v} = 0, \tag{4.19}$$

which means that Problem $\mathcal{A}$ is slightly modified from the collisionless Vlasov-Poisson system. In this case, problem $\mathcal{A}$ is solved through a classical RKDG framework. This means, that for each RK stage, one simply projects the source term onto the basis functions. This work can be done in parallel with sub-cycled lines. Problem $\mathcal{B}$ remains identical to what was presented in Chapter 3, where we needed to add in the additional correction terms to the Electric field. This is the case because it's no longer true that time derivatives of the Electric field come simply from moments of f. In this case, one needs to consider moments, as well as derivatives of moments of the source term as well. In Table 6 we present convergence studies for this problem using both Strang splitting as well as Yoshida splitting on the hybrid method. The purpose is to demonstrate what happens when we switch to high-order splitting. In Table 7, we compare two different fourth order split methods. The purpose there is to explore whether or not it's beneficial to add extra stages.

4.3.3 Landau Damping

Here we present simulation results for weak landau damping. C.f. §3.5.4 for details on the problem setup. In Figure 21 we present the linear decay of the L^2 norm of the electric field where our computed decay rate matches the linear decay rate of $\gamma = -0.1533$. In Figure 22 we denote many plots of the distribution function. In these pictures, we subtract out a normalized background Maxwellian, $f_M(v) = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}}$.

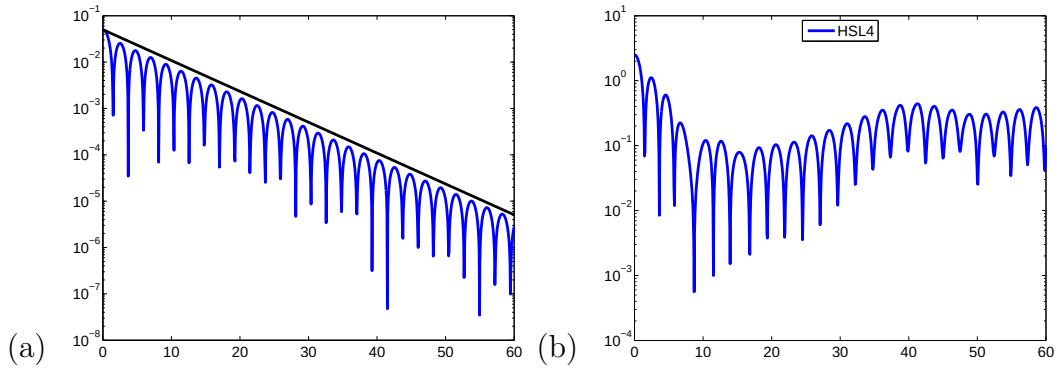


Figure 21: The Landau damping problem. Both panels present semi-log plots of the L_2 norm of the electric field. In panel (a), we present the linear decay for the Weak Landau damping problem ($\alpha = 0.01$). On top of our computed result we also plot the analytical decay rate of $\gamma = -0.1533$. In panel (b), we present the Strong Landau damping problem ($\alpha = 0.5$). Here, we see linear decay of the electric field, followed shortly thereafter by non-linear growth of the electric field due to particle trapping. Both simulations used a constant global CFL number of 2.0, with a local CFL restriction of 0.1. and are 4^{th} order accurate both in space and time. The grid resolution for each run was $(m_x, m_v) = (64, 128)$. We observe that the hybrid semi-Lagrangian method (HSLDG) closely matches the results for the full semi-Lagrangian method (SLDG).

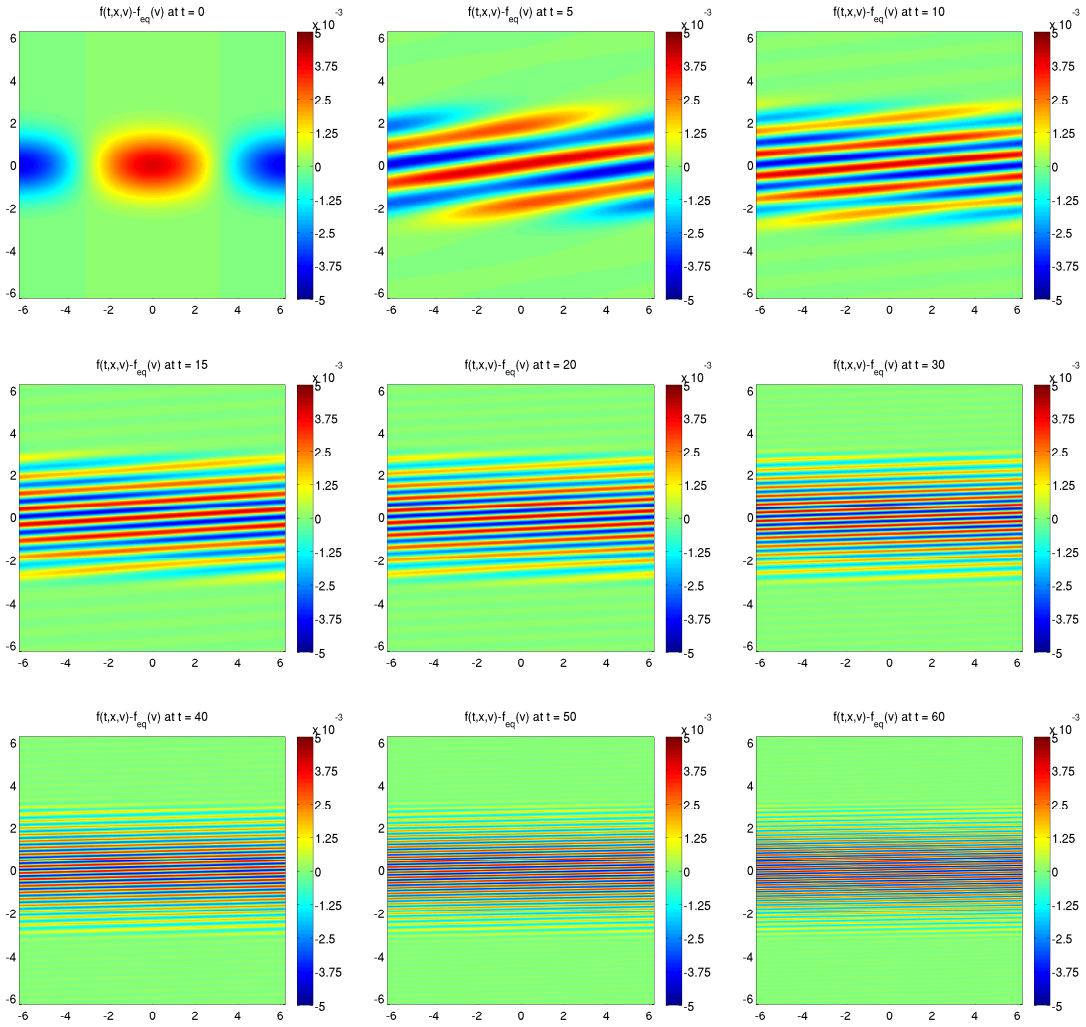


Figure 22: The weak Landau damping problem. Shown in the panels are the distribution function at various points in time, $t \in \{0, 5, 10, 15, 20, 30, 40, 50, 60\}$. This simulation was run with a constant CFL number of 2.0 on a mesh of size $(m_x, m_v) = (129, 129)$ using 4^{th} order accuracy both in space and time. In each panel, we have subtracted out the equilibrium distribution. It is clear from these plots that the high-order discontinuous Galerkin method is able to capture much of the fine-scale structure for the solution.

Mesh	**HSL2 Error**	$\log_2$**(Ratio)**	**HSL4 Error**	$\log_2$**(Ratio)**
10^2	5.193×10^{-1}	–	1.085×10^{0}	–
20^2	1.432×10^{-1}	1.858	2.725×10^{-1}	1.993
40^2	1.640×10^{-2}	3.126	1.652×10^{-2}	4.044
80^2	3.438×10^{-3}	2.255	7.058×10^{-4}	4.548
160^2	8.333×10^{-4}	2.045	3.421×10^{-5}	4.367
320^2	2.068×10^{-4}	2.011	1.953×10^{-6}	4.131
640^2	5.161×10^{-5}	2.003	1.197×10^{-7}	4.028
1280^2	1.290×10^{-5}	2.001	7.470×10^{-9}	4.002

Table 6: Convergence study for the forced Vlasov-Poisson equation. All calculations presented here are 4^{th} order in space and were run with a constant CFL number of 5.0. Shown are the relative errors computed via (A.14) at time $t = 1$. HSL2 refers to the Strang split hybrid scheme and HSL4 to the fourth-order hybrid split semi-Lagrangian scheme. The local time step chosen for each RKDG problem was fixed at a maximum CFL number of 0.1.

4.3.4 Bump-on-Tail

Here, we present a problem that was not presented in Chapter 3, the bump-on-tail problem. Much like the two-stream instability problem, the initial conditions are prescribed by two beams, one traveling to the left, and one traveling to the right. However, the difference here is that one beam has much higher intensity than the other. Specifically, the initial conditions are given by

$$f(t=0,x,v) = \frac{1}{\sqrt{2\pi}} \left(1 + 0.04\cos\left(0.3x\right)\right) \left(0.9e^{-\frac{v^2}{2}} + 0.2e^{-4(v-4.5)^2}\right) \tag{4.20}$$

and solve on the domain $(x,v) \in [-10\pi/3, 10\pi/3] \times [-8,8]$. These are the same parameters used by Banks & Hittinger [3] [38]. Results for time $t = 45$ are presented in

Mesh	**HSL4 Error**	$\log_2$**(Ratio)**	**HRKN4 Error**	$\log_2$**(Ratio)**
10^2	$1.085 \times 10^{+0}$	–	9.309×10^{-1}	–
20^2	2.725×10^{-1}	1.993	2.013×10^{-1}	2.209
40^2	1.652×10^{-2}	4.044	1.207×10^{-2}	4.060
80^2	7.058×10^{-4}	4.548	5.705×10^{-4}	4.403
160^2	3.421×10^{-5}	4.367	3.018×10^{-5}	4.241
320^2	1.953×10^{-6}	4.131	1.793×10^{-6}	4.073
640^2	1.197×10^{-7}	4.028	1.111×10^{-7}	4.013

Table 7: Comparison of two high-order split methods. We run the forced Vlasov-Poisson equation in order to compare errors incurred for Yoshida vs. Blanes and Moan operator splitting. Both methods run the hybrid method, the only difference is which splitting method we run. HSLDG refers to the Yoshida split, hybrid method, and HRKN4 uses Runge-Kutta-Nyström method in place of Yoshida splitting. All calculations presented here are 4^{th} order in space and time, and were run with a constant CFL number of 5.0. Shown are the relative errors computed via (A.14) at time $t = 1$. The local time step chosen for each RKDG problem was fixed at a maximum CFL number of 0.1. We note that the method of Blanes and Moan offers slightly increased accuracy, at the cost of adding extra stages (e.g. 13 stages vs. 7 stages).

Figure 23. We note that in this case, the integration constant J_0 required to attain 4^{th}-order accuracy is non-zero. For this problem only, we use the computed value of $J_0 = 3.181980515339465 \times 10^{-1}$. This term gets added into both $E_{,t}$ and $E_{,t,t,t}$.

4.3.5 Two Stream Instability

Here we present results for the two-stream instability problem. Again, we use the following initial distribution function

$$f(t=0,x,v) = \frac{v^2}{\sqrt{8\pi}}\left(2-\cos\left(\frac{x}{2}\right)\right)e^{-\frac{v^2}{2}}, \tag{4.21}$$

and solve on the domain $(x,v) \in [-2\pi, 2\pi] \times [-2\pi, 2\pi]$. These initial conditions produce two beams, one traveling left and one traveling right. Results for time $t = 45$ are presented in Figure 24 for various mesh sizes. The results for this two stream instability problem are comparable to what was observed in Chapter 3.

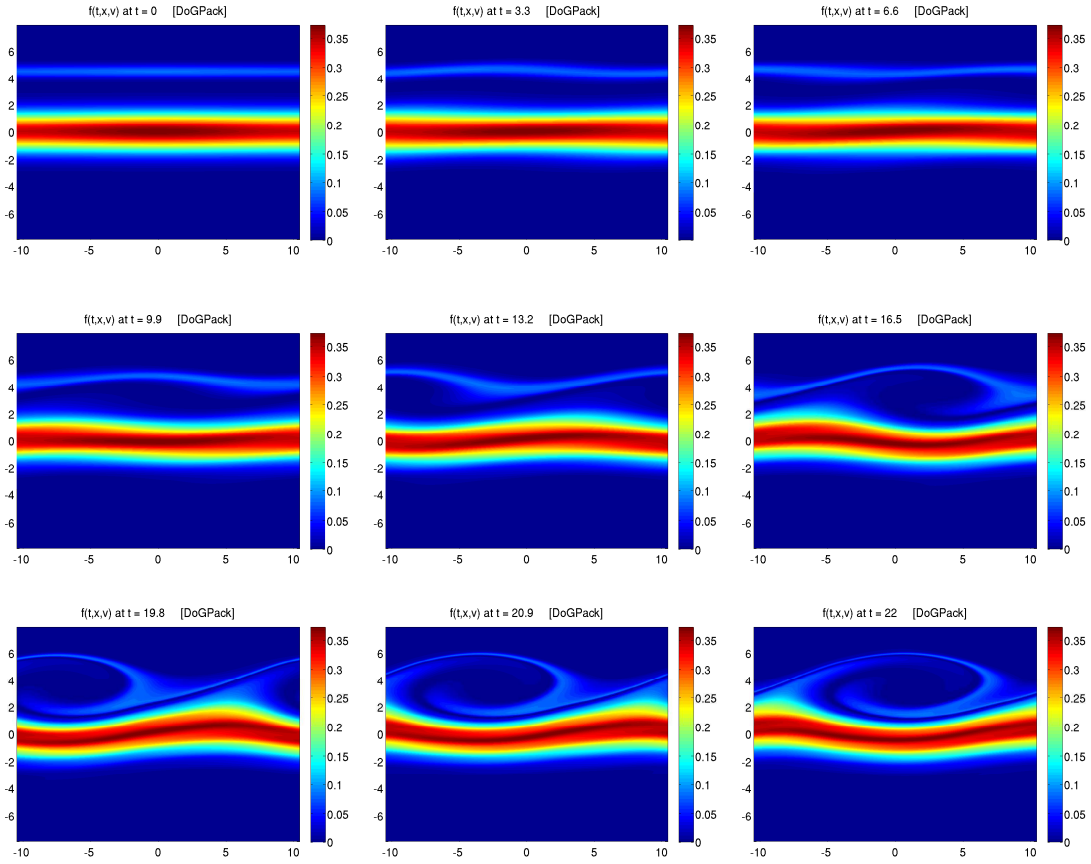


Figure 23: The Bump on Tail problem. All simulations are 4^{th} order in space, and 4^{th} order in time, and use the hybrid algorithm. The mesh size for each picture is $(m_x, m_v) = (129, 129)$. The above sequence of figures were produced by plotting the numerical solution at 5×5 uniform points in each mesh element. The local time step restriction used a CFL number of at most 0.1. The global CFL number was 2.0. We note that for this problem, the average momemtum is non-zero. The numerical value used for these simulation is $J_0 = 3.181980515339465 \times 10^{-1}$. This constant value contributes to both $E_{,t}$ and $E_{,t,t,t}$.

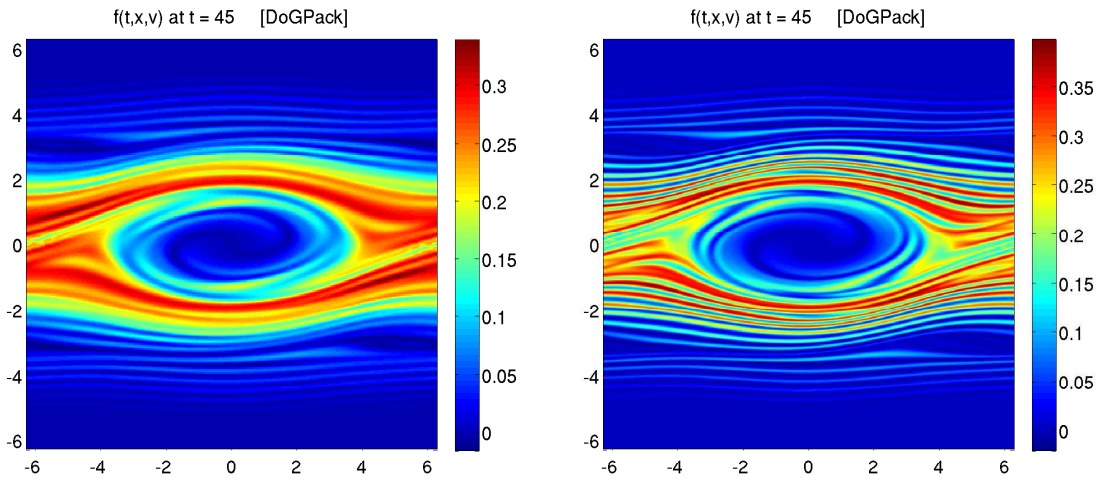


Figure 24: The two-stream instability problem. Grid refinement for the two stream instability problem. The panel on the left was run with a grid resolution $(m_x, m_v) = (65, 65)$, and the panel on the right was run with a grid resolution of $(m_x, m_v) = (129, 129)$. Both simulations were run to a final time of $t = 45$. Both simulations used the 4^{th} order Yoshida split method, and both used a 4^{th} order spatial discretization. Each picture was produced by plotting exactly 5^2 uniformly chosen points per grid cell. Both photos used a global CFL number of 2.00 for the simulations, and a local RK CFL number of 0.1 for the hybrid scheme.

Chapter 5

Towards a Hybrid Semi-Lagrangian Method for 2+2 Vlasov-Poisson

Efficient, high-order, positivity-preserving, stable, semi-Lagrangian DG methods are incredibly difficult to construct on unstructured grids, and at this point, we are unaware of any methods that retain all of these characteristics. Given that there are many problems with interesting solutions which require working on unstructured physical grids, we would like to create an efficient and accurate method which accommodates all of the above. Much of the groundwork has been presented in Chapter 4, and the focus of this chapter is to describe how those methods can be extended to higher dimensions. Here, we describe a full implementation of a hybrid, split, semi-Lagrangian DG (HSLDG) method based on a discontinuous Galerkin representation of the solution where one of the primary goals of this method is to be able to run a high order, continuum Vlasov solver on on an unstructured grid. Given the high dimensionality of the problem, a full implementation is too computationally expensive to run on a single computer, and will be the subject of future work. Here, we demonstrate that all of the necessary lower dimensional building blocks for a full implementation are in place.

A list of the required routines to be written include the following:

1. A 2D solver for $f_{,t} + (v^1 f)_{,x^1} + (v^2 f)_{,x^2} = 0$ that operates on an unstructured

grid. Here, we need only assume that the velocity field, $\mathbf{v} = (v^1, v^2)$ is constant. Because there is no simple way to construct a stable semi-Lagrangian solver on an unstructured grid, we choose to operate this problem using a classical Runge-Kutta (RKDG) method of lines formulation. We can dramatically improve the overall efficiency with the use of sub-cycling. In §5.2.1 we present an example of this method that uses periodic boundary conditions on a square, and we present a more complicated version of this in §5.2.2 on a circle. These two examples serve to illustrate (a) the existence of a mesh generator, (b) order of accuracy as well as (c) the existence of a working RKDG method on an unstructured grid.

2. A 2D Poisson solver that operates on an unstructured grid. This Poisson solver can be LU-factored once at the start of the run, and will be the subject of future work.

3. A 2D solver for $f_{,t} + (E^1(t, \mathbf{x})f)_{,v^1} + (E^2(t, \mathbf{x})f)_{,v^2} = 0$ that operates on a structured grid. Here, we choose to extend the semi-Lagrangian methods presented in the previous chapters. The purpose is to run a high-order method that operates with a single step, with the added benefits of remaining unconditionally stable. Here, we use the notation, $\mathbf{E} = (E^1, E^2)$ for the electric field.

5.1 Basic Scheme

The scheme for a 4D problem proceeds very close to the scheme developed for a quasi-1D problem as presented in Section 4.2.1. In place of "Convert to 1D Problems" and "Evolve 1D Problems", we now "Convert to 2D Problems" and "Evolve 2D Problems."

Additionally, in place of "Integrate 1D Problems", we now "Integrate 2D Problems" up to 4D coefficients. In this chapter, we focus on the "Convert to 2D Problems" step, as well as the "Evolve 2D Problems" step.

We begin with describing a method for advancing the following quasi-2D advection equation:

$$q_{,t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} q = 0; \tag{5.1}$$

where we are working on a structured grid cross an unstructured grid.

First, we present some notation that will be used for remainder of this section. With a slight abuse of notation, denote each triangular element in an unstructured grid in physical space $\mathbf{x}$ by $\mathcal{T}_i$, and denote each Cartesian element in velocity space $\mathbf{v}$ by $\mathcal{T}_{\mathbf{j}}$, with index $\mathbf{j} = (j^1, j^2)$. Here, the index (j^1, j^2) refers to the 2D grid element centered at $a_{v^1} + (j^1 - 1/2)\Delta v^1$ and $a_{v^2} + (j^2 - 1/2)\Delta v^2$ where a_{v^1} and a_{v^2} refer to the left endpoint of the Cartesian velocity mesh. Of course, indexing into a structured grid is much simpler than indexing into a unstructured grid, so it is possible for us to write this formula down. The mesh for the entire computational domain will then be denoted by

$$\mathcal{T}_{i\mathbf{j}} := \mathcal{T}_i \otimes \mathcal{T}_{\mathbf{j}} \subset \mathbb{R}^4 \tag{5.2}$$

We will denote the list of M_T quadrature points required for integration on a single 2D triangle by $\vec{\xi}_m$ with their associated quadrature weights, ω_m^T. We will denote the list of M^2 quadrature points and weights for each canonical velocity element by $\vec{\eta}_m$ and ω_m. Note that for each Cartesian slab, the list of points and weights are given simply by a tensor product of 1D weights and points:

$$\eta := \xi_{1\mathrm{D}} \otimes \xi_{1\mathrm{D}}; \quad \omega := \omega_{1\mathrm{D}} \otimes \omega_{1\mathrm{D}}, \tag{5.3}$$

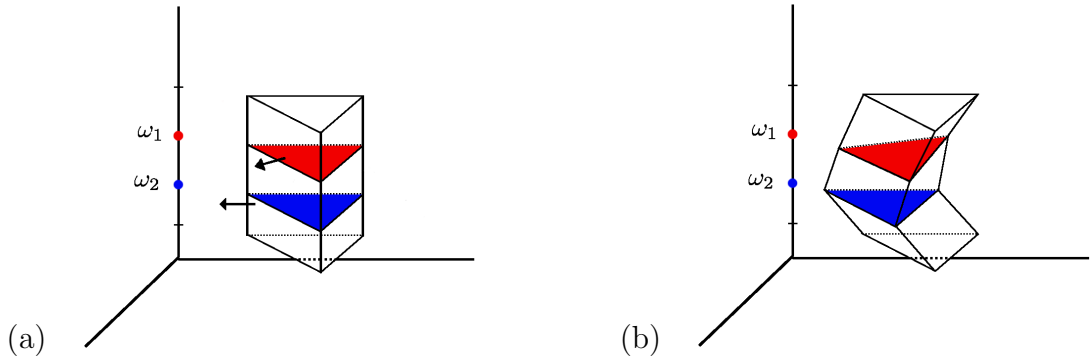


Figure 25: Diagram indicating evolution of 2D, unstructured equations. In this figure, we present two 'slices' of the distribution function. This function is projected onto finitely many slabs, and here we illustrate two such slabs. Each of these slabs receives a constant velocity, $\mathbf{v}_m$, as cartooned in panel (a). In panel (b), we present what the solution would look like at time t^{n+1} before projecting back onto the mesh elements, $\mathcal{T}_{i\mathbf{j}}$. The vertical axis represents a single 2D Cartesian velocity slab, $\mathcal{T}_{\mathbf{j}}$, and each triangle in this picture comes from a single quadrature point chosen from the velocity slab.

and therefore there are exactly M^2 of these points. We now direct the reader to the complete algorithm described in Algorithm 5.1.

Algorithm 5.1 describes evolution for (5.1), but it is easily modified to account for advection along characteristics defined by the electric field, $\mathbf{E}$:

$$q_{,t} + \mathbf{E}(t, \mathbf{x}) \cdot \nabla_{\mathbf{v}} q = 0. \tag{5.7}$$

Step 0 is identical, and in Step 1, the projection onto 2D coefficients simply replaces quadrature points for the velocity slab with quadrature points for the triangle. This produces a list of equations for each triangle. Step 2 is evolution via a SLDG method: for each equation m, this is an advection equation whose speeds depend only on time, which can be evolved using methods from §3.3.

Algorithm 5.1 Hybrid SLDG Method for Quasi-2D Equations

0. Initial Projection: Start with the Galerkin representation of the solution. When restricted to a single cell $\mathcal{T}_{i\mathbf{j}}$, this is simply

$$f^h(t,\mathbf{x},\mathbf{v})|_{\mathcal{T}_{i\mathbf{j}}} = \sum_{k=1}^{M_4} F_{i\mathbf{j}}^{(k)} \varphi^{(k)}(\vec{\xi},\vec{\eta}),$$

where the coefficients $F_{i\mathbf{j}}^{(k)}$ are a list of all M_4 required 4D DG moments.

1. Convert to 2D Problems: For each index $\mathbf{j}$, consider M^2 planes provided by the M^2 quadrature points $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{M^2}\}$. For each of these planes, project 4D Legendre moments onto 2D moments via

$$F_{2\mathrm{D},i}^{(k)}(t^n, m) := \frac{1}{|\mathcal{T}_{i,\cdot}|} \int \phi_{2D}^{(k)}(\vec{\xi}(\mathbf{x}))\, f^h(t^n,\mathbf{x},\mathbf{v}_m)\, d\mathbf{x}. \tag{5.4}$$

Each 2D integral can be evaluated exactly using enough quadrature points because $f^h(\cdot,\mathbf{v}_m)$ is simply a product of 2D polynomials.

2. Evolve 2D Problems: For each plane (henceforth referred to as equation), $m = 1, 2, \ldots, M^2$, take a step of length Δt on $f_{,t} + \mathbf{v}_m f_{,\mathbf{x}} = 0$, using the 2D method and produce $F_{2\mathrm{D},i}^{(k)}(t^{n+1}, m)$. In an RKDG setting, this will be accomplished through parallelized RK integration.

3. Integrate 2D Problems: Integrate the 2D coefficients up to 4D coefficients. This is accomplished through a tensor product of 2D quadrature points.

$$\begin{aligned} F_{i\mathbf{j}}^{(\ell)}(t^{n+1}) &= \frac{1}{|\mathcal{T}_{i\mathbf{j}}|} \iint_{\mathcal{T}_{i\mathbf{j}}} \varphi^{(\ell)}(\vec{\xi}(\mathbf{x}),\vec{\eta}(\mathbf{v})) f^h(t^{n+1},\mathbf{x},\mathbf{v})\, d\mathbf{x}\, d\mathbf{v} &(5.5)\\ &= \sum_{k,m_1,m_2} \frac{\omega_{m_1}\omega_{m_2}}{s} F_{2\mathrm{D},i}^{(k)}(t^{n+1}, m_2) \varphi^{(\ell)}(\vec{\xi}_{m_1}, \vec{\eta}_{m_2}) \phi_{2\mathrm{D}}^{(k)}(\vec{\xi}_{m_1}). &(5.6) \end{aligned}$$

Here, s is the size of the canonical element, and the range of ω_1 and ω_2 are a fixed number depending only on the number of quadrature points required for each triangle and 2D Cartesian cell.

Initial Grid Spacing	**Error**	$\log_2$**(Ratio)**
$h = 0.2$	1.48765×10^{-1}	–
$h = 0.1$	9.05163×10^{-3}	4.04
$h = 0.05$	3.26820×10^{-4}	4.79
$h = 0.025$	2.07851×10^{-5}	3.97
$h = 0.0125$	1.41659×10^{-6}	3.86

Table 8: Convergence study for the constant coefficient linear advection equation. All simulations were run with a constant CFL number of 0.1, and are 4^{th} order accurate both in space and time.

5.2 Numerical Examples

5.2.1 Periodic Advection

In this problem, we consider the domain $[0, 1] \times [0, 1]$ with an unstructured grid. Ghost cells are padded with periodic boundary conditions, and we take equation (5.1) with constant velocity field prescribed by $\mathbf{v} = (1, 1)$. We use this problem to demonstrate high-order convergence of the unstructured DG solver. In Table 8 we present a convergence study.

5.2.2 Solid Body Rotation

In this section we repeat the problem presented in §3.5.1, but this time with using an unstructured grid. That is, we consider a linear advection under a divergence-free velocity field:

$$q_{,t} + \mathbf{u} \cdot \nabla q = 0. \tag{5.8}$$

Because this solver uses unstructured grids, we can now take the computational domain to be the circle centered at $(0.5, 0.5)$ with radius 0.5, in place of the square, $[0,1] \times [0,1]$. The velocity field is chosen to produce a solid body rotation around its center:

$$\mathbf{u} = (u(y), v(x)) = (\pi (2y - 1), \pi (1 - 2x))^T . \tag{5.9}$$

The initial condition is taken to be a smooth, compactly supported bump that is centered at $(x_0, y_0) = (0.4, 0.5)$:

$$q(0, x, y) = \begin{cases} \cos^6\left(\frac{5\pi}{3} r\right) & \text{if} \quad r \leq 0.3, \\ 0 & \text{otherwise,} \end{cases} \tag{5.10}$$

where

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2}. \tag{5.11}$$

This problem, is solved via a RKDG approach. We run the initial condition out to time $t = 1$, at which point it should return to its initial state.

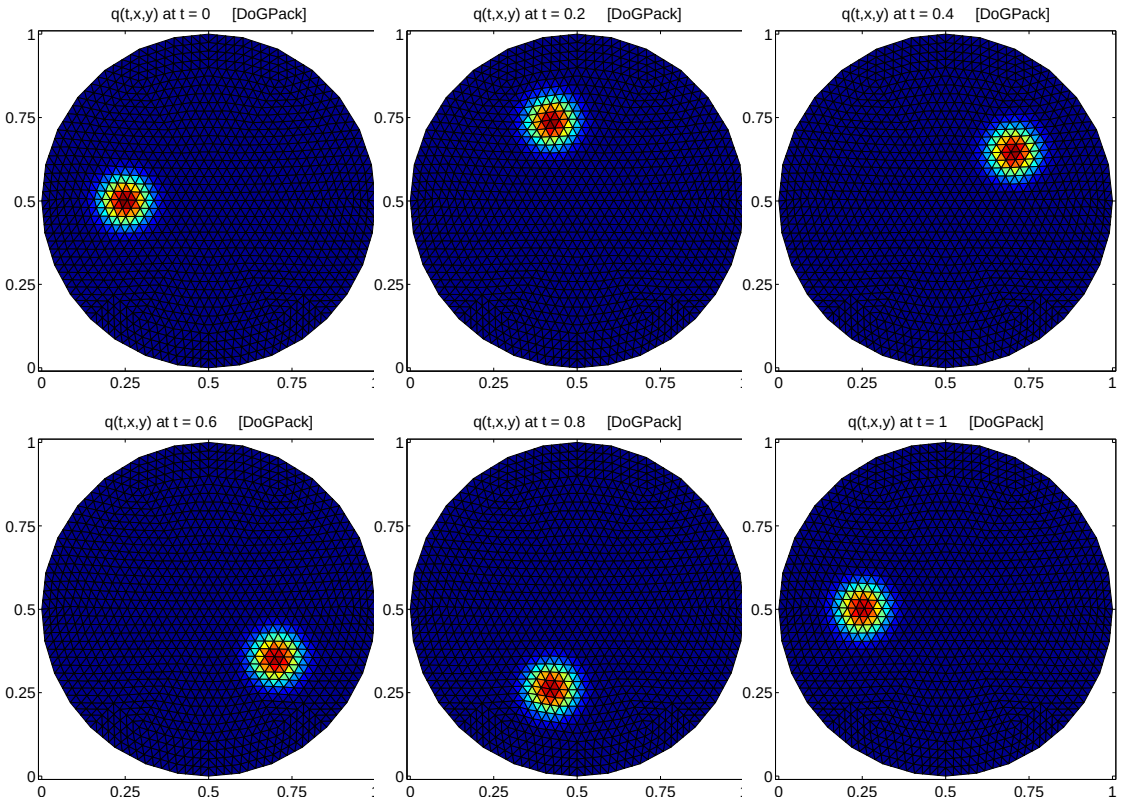


Figure 26: Solid body rotation problem. The initial conditions are a smooth compactly supported bump, and the problem is integrated until time $t = 1$, at which point it should return to the initial conditions. Each element in this simulation uses a 4^{th} order accurate representation of the solution, and a 4^{th} order time integrator was used.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this work, we have presented novel semi-Lagrangian methods based on a discontinuous Galerkin representation of the solution. We have presented two methods on the (1+1)D Vlasov-Poisson system, and we have produced results indicating that the correct ingredients are in place for solving a (2+2)D problem.

6.1.1 Semi-Lagrangian Discontinuous Galerkin

Our novel semi-Lagrangian discontinuous Galerkin (SLDG) numerical scheme for solving the Vlasov-Poisson system is based on solving a series of quasi-1D advection equations. We achieve unconditional stability by appealing to semi-Lagrangian time stepping, and high-order time accuracy is accomplished through high-order operator splitting methods. High-order spatial accuracy is attained through the use of a discontinuous Galerkin representation in space. The Poisson equation is solved to high-order via a modified local DG method, where the boundary conditions are set so that the discrete Laplacian matrix is by construction LU factored, which makes the Poisson solver extremely fast. In Chapter 3, we demonstrated the accuracy and robustness of the proposed SLDG method on several classical test problems from the literature, where we have managed

to simultaneously accomplish all of the following:

1. Unconditional stability;
2. Mass conservation;
3. Positivity-preserving;
4. 4^{th} order accuracy in time;
5. 5^{th} order accuracy in space.

In its current state, the method is limited to operating on Cartesian grids.

6.1.2 Hybrid Semi-Lagrangian Discontinuous Galerkin

In chapter 4, we proposed a hybrid, semi-Lagrangian discontinuous Galerkin (HSLDG) scheme for the Vlasov system. This scheme is based on classical RKDG methods as well as our novel SLDG methods. This scheme utilizes semi-Lagrangian methods for time stepping the electric field, and RK time stepping for solving problems in physical space. In order to alleviate strict CFL conditions, the RK problems utilized sub-cycled time stepping based on a local CFL condition, which allowed us to take larger time steps for smaller velocities. We ran this scheme on several (1+1)D test problems from the literature, and we argue that the results are promising. Many methods, such as the Poisson solver, the high order expansion of the electric field and time stepping on the electric field that were developed for the full SLDG method were utilized inside the hybrid scheme.

We admit that the hybrid method is not as efficient as the full semi-Lagrangian scheme for structured grids, however the real purpose of presenting the method there

was to demonstrate its efficacy on a reduced model (i.e. lower dimensions). Given its success there, we argue that it will make for a useful scheme in higher dimensions.

6.1.3 Hybrid SLDG Methods for (2+2)D Unstructured Grids

The complete details for extensions of the HSLDG method to higher dimensions, and in particular to unstructured grids was demonstrated in Chapter 5. There, we demonstrated that the bulk of routines are in place for operating a (2+2)D Vlasov-Poisson solver on unstructured grids. These ingredients include a description of the method, and methods for evolving 2D problems. For our operator split approach, there are two types of grids that need to be evolved: structured 2D grids for velocity space, and unstructured 2D grids for physical space. We choose a structured velocity space grid in order to incorporate semi-Lagrangian time stepping. We choose an unstructured grid for physical space in order to accommodate irregular domains. We have demonstrated that a 2D grid generator is in place, and we have run our high order RKDG method on sample unstructured grid problems. The evolution step for the velocity coordinates is identical to what has already been presented for the full SLDG method.

6.2 Future Work

Direct extensions of the full semi-Lagrangian (SLDG) method to higher dimensions can certainly be accomplished. We could construct a method that permits unconditional stability through semi-Lagrangian time stepping, but as of now, the method would be limited to working on structured grids. We therefore would like to turn towards a method

that will accommodate unstructured grids, and therefore, the proposed hybrid semi-Lagrangian discontinuous Galerkin (HSLDG) method presented in Chapter 5 proves to be a promising avenue of research. Through the use of semi-Lagrangian time-stepping and unstructured grids, we hope that ultimately the methods presented there will be viewed as a bridge between particle and pure Eulerian methods, in the sense that we retain the ability to take large time-steps (semi-Lagrangian) and can handle complex geometries (unstructured meshes).

The complete method has been described in Chapter 5, but has not been completely implemented. Given the constraints of working on a single computer with such large data sets, full implementation will require the use of parallelization with MPI (Message Passing Interface) communications and distributed computing. In short, when the the complete scheme has been implemented, the full HSLDG method for the (2+2)D problem will simultaneously accomplish all of the following:

1. Unstructured grids in physical space;
2. Mass conservative;
3. Positivity-preserving;
4. 4^{th} order accurate in time;
5. 5^{th} order accurate in space.

It will still retain its unconditional stability for advection in the velocity coordinates, but we will use sub-cycling (as has already been presented in Chapter 4) to alleviate the strict CFL condition present for advection over the physical coordinates.

Future work will focus not only on extending the results described in this paper to higher-dimensional Vlasov-Poisson equations, but also modifications of the current approach to both the non-relativistic and the relativistic Vlasov-Maxwell equations will be considered.

Appendix A

Numerical evaluation of conserved quantities

The conserved quantities defined in (1.49) (L_1-norm), (1.50) (L_2-norm), (1.51) (total energy), and (1.52) (entropy) are used as diagnostics of the numerical methods proposed in this work.

In order to evaluate all of these conserved quantities in the numerical evolution, we define the following functional:

$$I^h\big(g(f^h)\big) := \frac{\Delta x\,\Delta v}{4}\sum_{i=1}^{m_x}\sum_{j=1}^{m_v}\sum_{k=1}^{M^2}\omega_k\, g\left(f^h\left(x_i+\frac{\xi_k\Delta x}{2}, v_j+\frac{\eta_k\Delta v}{2}\right)\right), \tag{A.1}$$

where m_x is the number of elements in the x-direction, m_v is the number of elements in the v-direction, and ω_k and (ξ_k, η_k) are the M^2 Gauss-Legendre quadrature weights and points on $[-1,1]\times[-1,1]$, respectively. Expression (A.1) gives a numerical approximation to integrals of the form:

$$I\left(g(f)\right) := \int_{-L}^{L}\int_{-\infty}^{\infty} g\left(f(x,v)\right)\, dv\, dx. \tag{A.2}$$

Using (A.1) we define the following numerical approximations to the norms defined by

(1.49)–(1.52):

$$\|f^h\|_{L_1} := I^h\left(\left|f^h\right|\right), \tag{A.3}$$

$$\|f^h\|_{L_2} := \left\{\frac{\Delta x\,\Delta v}{4}\sum_{i=1}^{m_x}\sum_{j=1}^{m_v}\sum_{\ell=1}^{M(M+1)/2}\left[F_{ij}^{(\ell)}\right]^2\right\}^{\frac{1}{2}}, \tag{A.4}$$

$$\text{Total energy} := \frac{1}{2}I^h\left(v^2\,f^h\right) + \frac{\Delta x}{4}\sum_{i=1}^{m_x}\sum_{\ell=1}^{M}\left[E_i^{(\ell)}\right]^2, \tag{A.5}$$

$$\text{Entropy} := -I^h\left(f^h\,\log(f^h)\right). \tag{A.6}$$

A.1 Relative L_2-norm error in 1D

Let $f(x)$ be the exact solution of some problem of interest. Let $f^h(x)$ denote an approximation to $f(x)$ using a discontinuous Galerkin method. On each element $f^h(x)$ and $f(x)$ can be written as

$$f^h(x)\Big|_{\mathcal{T}_i} = \sum_{k=1}^{M} F_i^{(k)}\,\varphi^{(k)}(\xi), \tag{A.7}$$

$$f(x)\Big|_{\mathcal{T}_i} = \sum_{k=1}^{\infty} \mathcal{F}_i^{(k)}\,\varphi^{(k)}(\xi), \tag{A.8}$$

respectively. The relative L_2-norm of the difference on the domain $x \in [a,b]$ between the approximation, $f^h(x)$, and the exact solution, $f(x)$, is given by

$$\begin{aligned}\frac{\|f(x)-f^h(x)\|_{L_2}}{\|f(x)\|_{L_2}} &= \left\{\frac{\int_a^b\left[f(x)-f^h(x)\right]^2\,dx}{\int_a^b f(x)^2\,dx}\right\}^{\frac{1}{2}} \\ &= \left\{\frac{\sum_{i=1}^{N}\sum_{k=1}^{M}\left[F_i^{(k)}-\mathcal{F}_i^{(k)}\right]^2}{\sum_{i=1}^{N}\sum_{k=1}^{M}\left[\mathcal{F}_i^{(k)}\right]^2}\right\}^{\frac{1}{2}} + \mathcal{O}\left(\Delta x^M\right),\end{aligned} \tag{A.9}$$

where N is the total number of grid elements and $\Delta x = (b-a)/N$. Therefore, we take as our relative L_2-norm indicator the following easily computable quantity:

$$E_2(\Delta x, M) := \left\{ \frac{\sum_{i=1}^{N} \sum_{k=1}^{M} \left[F_i^{(k)} - \mathcal{F}_i^{(k)} \right]^2}{\sum_{i=1}^{N} \sum_{k=1}^{M} \left[\mathcal{F}_i^{(k)} \right]^2} \right\}^{\frac{1}{2}} . \tag{A.10}$$

A.2 Relative L_2-norm error in 2D

Let $f(x,y)$ be the exact solution of some problem of interest. Let $f^h(x,y)$ denote an approximation to $f(x,y)$ using a discontinuous Galerkin method. On each element $f^h(x,y)$ and $f(x,y)$ can be written as

$$f^h(x,y)\Big|_{\mathcal{T}_{ij}} = \sum_{k=1}^{M(M+1)/2} F_{ij}^{(k)} \, \varphi^{(k)}(\xi,\eta), \tag{A.11}$$

$$f(x,y)\Big|_{\mathcal{T}_{ij}} = \sum_{k=1}^{\infty} \mathcal{F}_{ij}^{(k)} \, \varphi^{(k)}(\xi,\eta), \tag{A.12}$$

respectively. The relative L_2-norm of the difference on the domain $(x,y) \in [a_x, b_x] \times [a_y, b_y]$ between the approximation, $f^h(x,y)$, and the exact solution, $f(x,y)$, is given by

$$\begin{aligned} \frac{\|f(x,y) - f^h(x,y)\|_{L_2}}{\|f(x,y)\|_{L_2}} &= \left\{ \frac{\int_{a_x}^{b_x} \int_{a_y}^{b_y} \left[f(x,y) - f^h(x,y) \right]^2 dy\, dx}{\int_{a_x}^{b_x} \int_{a_y}^{b_y} f(x,y)^2 \, dy\, dx} \right\}^{\frac{1}{2}} \\ &= \left\{ \frac{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{M(M+1)/2} \left[F_{ij}^{(k)} - \mathcal{F}_{ij}^{(k)} \right]^2}{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{M(M+1)/2} \left[\mathcal{F}_{ij}^{(k)} \right]^2} \right\}^{\frac{1}{2}} + \mathcal{O}\left(\Delta x^M, \Delta y^M \right), \end{aligned} \tag{A.13}$$

where N_x and N_y are the the number of grid elements in each coordinate direction, $\Delta x = (b_x - a_x)/N_x$, and $\Delta y = (b_y - a_y)/N_y$. Therefore, we take as our relative L_2-norm indicator the following easily computable quantity:

$$E_2(\Delta x, \Delta y, M) := \left\{ \frac{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{M(M+1)/2} \left[F_{ij}^{(k)} - \mathcal{F}_{ij}^{(k)} \right]^2}{\sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{M(M+1)/2} \left[\mathcal{F}_{ij}^{(k)} \right]^2} \right\}^{\frac{1}{2}} . \tag{A.14}$$

Appendix B

Numerical Integration

Quadrature Rules

Quadrature rules are used to numerically compute integrals of the form:

$$\int_{-1}^{1} q(\xi)\, d\xi. \tag{B.1}$$

Specifically, they are used as a method of approximating the above integral using a finite set of points, and summing function values at these together with appropriate weights. If we define a set of quadrature points, $\{\xi_1, \xi_2, \ldots, \xi_n\}$ together with their associated quadrature weigths, $\{\omega_1, \omega_2, \cdots \omega_n\}$, we can approximate a finite integral with

$$\sum_{m=1}^{n} \omega_m q(\xi_n) \approx \int_{-1}^{1} q(\xi)\, d\xi.$$

In Table 9 we present a complete list of Gauss-Legendre quadrature points, together with their associated weights. Gauss-Legendre integration is built to optimize integration of polynomials. This means that when n points are used, the approximate integral integrates any polynomial of degree at most $2n-1$ exactly.

Numerical integration for 2D problems can be performed by taking a tensor product of all the 1D numerical weights. If $B = [-1,1] \times [-1,1]$ is the unit box, then

$$\iint_B q(\xi,\eta)\; d\xi d\eta \approx \sum_{m_1,m_2=1}^{M} \omega_{m_1}\omega_{m_2} q(\xi_{m_1}, \eta_{m_2}). \tag{B.2}$$

Quadrature Points:	ξ_1	ξ_2	ξ_3	ξ_4	ξ_5
1	0	–	–	–	–
2	$-\frac{1}{\sqrt{3}}$	$\frac{1}{\sqrt{3}}$	–	–	–
3	$-\sqrt{\frac{3}{5}}$	0	$\sqrt{\frac{3}{5}}$	–	–
4	$-\frac{\sqrt{3+\sqrt{4.8}}}{\sqrt{7}}$	$-\frac{\sqrt{3-\sqrt{4.8}}}{\sqrt{7}}$	$\frac{\sqrt{3-\sqrt{4.8}}}{\sqrt{7}}$	$\frac{\sqrt{3+\sqrt{4.8}}}{\sqrt{7}}$	–
5	$-\frac{1}{3}\sqrt{5+\gamma}$	$-\frac{1}{3}\sqrt{5-\gamma}$	0	$\frac{1}{3}\sqrt{5-\gamma}$	$\frac{1}{3}\sqrt{5+\gamma}$
Quadrature Weights:	ω_1	ω_2	ω_3	ω_4	ω_5
1	2.0	–	–	–	–
2	1.0	1.0	–	–	–
3	$\frac{18-\sqrt{30}}{36}$	$\frac{18+\sqrt{30}}{36}$	$\frac{18+\sqrt{30}}{36}$	$\frac{18-\sqrt{30}}{36}$	–
4	$\frac{18-\sqrt{30}}{36}$	$\frac{18+\sqrt{30}}{36}$	$\frac{18+\sqrt{30}}{36}$	$\frac{18-\sqrt{30}}{36}$	–
5	$\frac{322-13\sqrt{70}}{900}$	$\frac{322+13\sqrt{70}}{900}$	$\frac{128}{225}$	$\frac{322+13\sqrt{70}}{900}$	$\frac{322-13\sqrt{70}}{900}$

Table 9: List of $1D$ quadrature weights and points, listed in increasing order. Each row containing M points will integrate polynomials of degree $2M-1$ exactly. The rules for using five points use the constant $\gamma = 2\sqrt{10/7}$ in order to make the entire list fit on the table.

Each ξ_i, η_j and ω_m come from the list of 1D quadrature points and weights listed in Table 9. This integral formula can be constructed from the 1D formula by integrating each coordinate in turn.

Bibliography

[1] D.N. Arnold, F. Brezzi, B. Cockburn, and L.D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39:1749–1779, 2002.

[2] J. W. Banks, R. L. Berger, S. Brunner, B. I. Cohen, and J. A. F. Hittinger. Two-dimensional Vlasov simulation of electron plasma wave trapping, wavefront bowing, self-focusing, and sideloss. *Physics of Plasmas*, 18(5):052102, 2011.

[3] J.W. Banks and J.A.F. Hittinger. A new class of nonlinear finite-volume methods for Vlasov simulation. *IEEE Transactions on Plasma Science*, 38:2198–2207, 2010.

[4] J. Barnes and P. Hut. A hierarchical O($N \log N$) force-calculation algorithm. *Nature*, 324:446 – 449, 1986.

[5] R. Belaouar, N. Crouseilles, P. Degond, and E. Sonnendrücker. An asymptotically stable semi-Lagrangian scheme in the quasi-neutral limit. *J. Sci. Comput.*, 41:341–365, 2009.

[6] N. Besse, J. Segre, and E. Sonnendrücker. Semi-Lagrangian schemes for the two-dimensional Vlasov-Poisson system on unstructured meshes. *Transp. Theory and Stat. Phys.*, 34:311–332, 2005.

[7] N. Besse, J. Segré, and E. Sonnendrücker. Semi-Lagrangian schemes for the two-dimensional Vlasov-Poisson system on unstructured meshes. *Transport Theory and Statistical Physics*, 34:311–332, Aug 2005.

[8] N. Bessho and A. Bhattacharjee. Fast collisionless reconnection in electron-positron plasmas. *Physics of Plasmas*, 14:056503, 2007.

[9] C.K. Birdsall and A.B. Langdon. *Plasma physics via computer simulation*. Taylor & Francis Group, 1985.

[10] J. Birn, J.F. Drake, M.A. Shay, B.N. Rogers, R.E. Denton, M. Hesse, M. Kuznetsova, Z.W. Ma, A. Bhattacharjee, A. Otto, and P.L. Pritchett. Geospace environmental modeling (GEM) magnetic reconnection challenge. *Journal of Geophysical Research - Space Physics*, 106(A3):3715–3719, 2001.

[11] S. Blanes and P.C. Moan. Practical symplectic partitioned Runge-Kutta and Runge-Kutta-Nyström methods. *Journal of Computational and Applied Mathematics*, 142(2):313 – 330, 2002.

[12] C. Cheng and G. Knorr. The integration of the Vlasov equation in configuration space. *J. Comp. Phys.*, 22:330–351, 1976.

[13] A.J. Christlieb, W.N.G. Hitchon, and E.R. Keiter. A computational investigation of the effects of varying discharge geometry for an inductively coupled plasma. *IEEE Transactions on Plasma Science*, 28:2214–2231, 2000.

[14] A.J. Christlieb, R. Krasny, and J.P. Verboncoeur. Efficient particle simulation of a virtual cathode using a grid-free treecode Poisson solver. *IEEE Transactions on Plasma Science*, 32:384–389, 2004.

[15] A.J. Christlieb, R. Krasny, J.P. Verboncoeur, J.W. Emhoff, and I.D. Boyd. Grid-free

plasma simulation techniques. *IEEE Transactions on Plasma Science*, 34:149–165, 2006.

[16] A.J. Christlieb, R. Krasny, J.P. Verboncoeur, J.W. Emhoff, and I.D. Boyd. Grid-free plasma simulation techniques. *Plasma Science, IEEE Transactions on*, 34(2):149 – 165, April 2006.

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

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

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

[20] B. Cockburn and C.-W. Shu. The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems. *J. Comp. Phys.*, 141:199–224, 1998.

[21] Bernardo Cockburn and Chi-Wang Shu. The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws. *RAIRO Modél. Math. Anal. Numér.*, 25(3):337–361, 1991.

[22] O. Coulaud, E. Sonnendrücker, E. Dillon, P. Bertrand, and A. Ghizzo. Parallelization of semi-Lagrangian Vlasov codes. *J. Plasma Phys.*, 61:435–448, 1999.

[23] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzengleichungen der mathematischen physik. *Mathematische Annalen*, 100:32–74, 1928. 10.1007/BF01448839.

[24] N. Crouseilles, G. Latu, and E. Sonnendrücker. Hermite spline interpolation on patches for parallelly solving the Vlasov-Poisson equation. *Int. J. Appl. Math. and Comp. Sci.*, 17:335–349, 2007.

[25] N. Crouseilles, M. Mehrenberger, and E. Sonnendrücker. Conservative semi-Lagrangian schemes for Vlasov equations. *J. Comp. Phys.*, 229:1927–1953, 2010.

[26] N. Crouseilles, T. Respaud, and E. Sonnendrücker. A forward semi-Lagrangian method for the numerical solution of the Vlasov equation. *Comp. Phys. Comm.*, 180:1730–1745, 2009.

[27] Nicolas Crouseilles, Erwan Faou, and Michel Mehrenberger. High order Runge-Kutta-Nyström splitting methods for the Vlasov-Poisson equation. `http://www.irisa.fr/ipso/perso/faou/publis/cfm2.pdf`, 2011.

[28] Nicolas Crouseilles, Michel Mehrenberger, and Eric Sonnendrücker. Conservative semi-Lagrangian schemes for Vlasov equations. *J. Comput. Phys.*, 229(6):1927–1953, 2010.

[29] Bernd Einfeldt. On Godunov-type methods for gas dynamics. *SIAM Journal on Numerical Analysis*, 25(2):294–318, 1988.

[30] F. Filbet and E. Sonnendrücker. Comparison of Eulerian Vlasov solvers. *Comp. Phys. Comm.*, 150:247–266, 2003.

[31] E. Forest and R.D. Ruth. Fourth-order symplectic integration. *Physica D: Nonlinear Phenomena*, 43:105–117, 1990.

[32] R. Glassey. *The Cauchy Problem in Kinetic Theory*. Miscellaneous Titles Series. Society for Industrial and Applied Mathematics, 1996.

[33] Sigal Gottlieb, David I. Ketcheson, and Chi-Wang Shu. High order strong stability preserving time discretizations. *J. Sci. Comput.*, 38(3):251–289, 2009.

[34] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Phys.*, 73:325–348, 1987.

[35] Amiram Harten, Peter D. Lax, and Bram Van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):pp. 35–61, 1983.

[36] R.E. Heath, I.M. Gamba, P.J. Morrison, and C. Michler. A discontinuous Galerkin method for the Vlasov—Poisson system. *Journal of Computational Physics*, 231(4):1140 – 1174, 2012.

[37] J.S. Hesthaven and T. Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer, 2007.

[38] J. A. F. Hittinger and J. W. Banks. Block-structured adaptive mesh refinement algorithms for vlasov simulation. *CoRR*, abs/1204.3853, 2012.

[39] R.W. Hockney and J.W. Eastwood. *Computer simulation using particles*. Institute of Physics Publishing, 1988.

[40] Y. Idomura, M. Ida, and S. Tokuda. Conservative gyrokinetic Vlasov simulation. *Communications in Nonlinear Science and Numerical Simulation*, 13:227–233, 2008.

[41] G. B. Jacobs and J. S. Hesthaven. Implicit-explicit time integration of a high-order particle-in-cell method with hyperbolic divergence cleaning. *Comput. Phys. Comm.*, 180(10):1760–1767, 2009.

[42] G.B. Jacobs and J.S. Hesthaven. High-order nodal discontinuous Galerkin particle-in-cell method on unstructured grids. *J. Comp. Phys.*, 96–121(214), 2006.

[43] C.-W. Shu J.M. Qiu. Positivity preserving semi-Lagrangian discontinuous Galerkin formulation: theoretical analysis and application to the Vlasov-Poisson system. *J. Comp. Phys.*, 230 (23), 2011.

[44] David I. Ketcheson. Runge-Kutta methods with minimum storage implementations. *J. Comput. Phys.*, 229(5):1763–1773, 2010.

[45] K. Lindsay and R. Krasny. A particle method and adaptive treecode for vortex sheet motion in three-dimensional flow. *J. Comp. Phys.*, 172:879–907, 2001.

[46] C. Mouhot and C. Villani. Landau damping. *J. Math. Phys.*, 51(015204), 2010.

[47] Clément Mouhot and Cédric Villani. On Landau damping. *Acta Math.*, 207(1):29–201, 2011.

[48] G.J. Parker and W.N.G. Hitchon. Convected scheme simulations of the electron

distribution function in a positive column plasma. *Jpn. J. Appl. Phys.*, 36:4799–4807, 1997.

[49] Jing-Mei Qiu and Andrew Christlieb. A conservative high order semi-Lagrangian WENO method for the Vlasov equation. *J. Comput. Phys.*, 229(4):1130–1149, 2010.

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

[51] M. Restelli, L. Bonaventura, and R. Sacco. A semi-Lagrangian discontinuous Galerkin method for scalar advection by incompressible flows. *J. Comput. Phys.*, 216(1):195–215, Jul 2006.

[52] J.A. Rossmanith. DoGPack software. Available from `http://www.dogpack-code.org`.

[53] James A. Rossmanith and David C. Seal. A positivity-preserving high-order semi-Lagrangian discontinuous Galerkin scheme for the Vlasov-Poisson equations. *J. Comput. Phys.*, 230(16):6203–6232, 2011.

[54] Jack Schaeffer. Global existence of smooth solutions to the Vlasov Poisson system in three dimensions. *Communications in Partial Differential Equations*, 16(8-9):1313–1335, 1991.

[55] H. Schmitz and R. Grauer. Darwin–Vlasov simulations of magnetised plasmas. *J. Comp. Phys.*, 214:738—756, 2006.

[56] Q. Sheng. Solving linear partial differential equations by exponential splitting. *IMA Journal of Numerical Analysis*, 9(2):199–212, 1989.

[57] Chi-Wang Shu. Total-variation-diminishing time discretizations. *SIAM Journal on Scientific and Statistical Computing*, 9(6):1073–1084, 1988.

[58] N.J. Sircombe and T.D. Arber. Valis: A split-conservative scheme for the relativistic 2D Vlasov-Maxwell system. *Journal of Computational Physics*, 228(13):4773 – 4788, 2009.

[59] E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo. The semi-Lagrangian method for the numerical resolution of the Vlasov equation. *J. Comp. Phys.*, 149:201–220, 1999.

[60] G. Strang. On the construction and comparison of difference schemes. *SIAM J. Num. Anal.*, pages 506–517, 1968.

[61] H. F. Trotter. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society*, 10(4):pp. 545–551, 1959.

[62] J.-L. Vay, P. Colella, J.W. Kwan, P. McCorquodale, D.B. Serafini, A. Friedman, D.P. Grote, G. Westenskow, J.-C. Adam, A. Héron, and I. Haber. Application of adaptive mesh refinement to particle-in-cell simulations of plasmas and beams. *Phys. Plasmas*, 11(2928), 2004.

[63] H. Yoshida. Construction of higher order symplectic integrators. *Phys. Lett. A*, 150:262–268, 1990.

[64] Haruo Yoshida. Recent progress in the theory and application of symplectic integrators. *Celestial Mechanics and Dynamical Astronomy*, 56:27–43, 1993.

[65] S.I Zaki, L.R.T Gardner, and T.J.M Boyd. A finite element code for the simulation of one-dimensional Vlasov plasmas. I: Theory. *Journal of Computational Physics*, 79(1):184 – 199, 1988.

[66] X. Zhang and C.-W. Shu. On maximum-principle-satisfying high order schemes for scalar conservation laws. *J. Comp. Phys.*, 229:3091–3120, 2010.

[67] Xiangxiong Zhang and Chi-Wang Shu. Maximum-principle-satisfying and positivity-preserving high-order schemes for conservation laws: Survey and new developments. *Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.*, 467(2134):2752–2776, 2011.

[68] Tie Zhou, Yan Guo, and Chi-Wang Shu. Numerical study on Landau damping. *Physica D: Nonlinear Phenomena*, 157(4):322 – 333, 2001.

[69] Tie Zhou, Yinfan Li, and Chi-Wang Shu. Numerical comparison of WENO finite volume and Runge-Kutta discontinuous Galerkin methods. *J. Sci. Comput.*, 16(2):145–171, 2001.