Deterministic Methods for the Boltzmann Equation

by Akil Narayan and Andreas Klöckner <{anaray,kloeckner}@dam.brown.edu>

Table of contents

1 Introduction

The Boltzmann equation is an equation of statistical mechanics that describes the evolution of a rarefied gas.

- In a *fluid* in *continuum mechanics*, all particles in a spatial volume element are approximated as having the same velocity.

- In a *rarefied gas* in *statistical mechanics*, there is enough space that particles in one spatial volume element may have different velocities.

The equation itself is a nonlinear integro-differential equation which decribes the evolution of the density of particles in a monatomic rarefied gas. To be precise, let the density function be $f(x, v, t)$. The quantity $f(x, v, t)\mathrm{d}x\,\mathrm{d}v$ represents the number of particles in the phase-space volume element $\mathrm{d}x\,\mathrm{d}v$ at time t. Both x and v are three-dimensional independent variables, so the density function is a map $f\colon \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}_0^+$. The theory of kinetics and statistical mechanics gives rise to the laws which the density function f must obey in the absence of external forces. From these physical constraints, an evolution law for the density can be derived, which reads

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f = \frac{1}{k_n} Q(f, f), \qquad x, v \in \mathbb{R}^3.$$

In a more general model, we should also allow for the existence of external forces. We introduce the $\mathbb{R}^3$-valued function $L(x, t)$. The function $L\colon \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}$ represents forces which act upon thedensity function that are independent of its own inertia. In many applications where f represents the density of a species of charged particles (e.g. electrons), the forcing function L is the electromagnetic *Lorentz force* induced by the particles themselves. In particular, we may derive charge and current densities from the molecular species density:

$$\left.\begin{aligned} \rho(x, t) &= \int_{\mathbb{R}^3} f(x, v, t)\,\mathrm{d}v \\ J(x, t) &= \int_{\mathbb{R}^3} v\, f(x, v, t)\,\mathrm{d}v \end{aligned}\right\} \tag{1}$$

The charge density $\rho\colon \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}$ and the current density $J\colon \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ create inhomogeneous forcing terms for the governing laws of electromagnetics, *Maxwell's equations:*

$$\begin{array}{ll} \varepsilon \dfrac{\partial E}{\partial t} = \nabla_x \times \dfrac{B}{\mu} - J & \dfrac{\partial B}{\partial t} = -\nabla_x \times E \\[2ex] \nabla_x \cdot B = 0 & \nabla_x \cdot (\varepsilon E) = \rho. \end{array} \tag{2}$$

We have introduced several variables: $E\colon \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ and $B\colon \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ are the electric and magnetic fields, respectively, and are functions of space and time only (i.e. not of velocity v). The parameters μ and ε are the magnetic permeability and electric permittivity, respectively, and represent spatially-varying properties of the ambient material. The electric and magnetic fields give rise to the Lorentz force:

$$L = (E + v \times B). \tag{3}$$

Having defined the Lorentz force, we can introduce this new forcing term to form the full *Boltzmann equation:*

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f + L \cdot \nabla_v f = \frac{1}{k_n} Q(f, f), \qquad x, v \in \mathbb{R}^3, \tag{4}$$

The left hand side of equation (4) is called the *Vlasov Equation*–it is simply advection: the density (i.e. the particles which the density represents) evolves according to its velocity and the Eulerian forces acting upon it. The right hand side operator $Q(f,\, f)$ is a *collision term*, which represents the binary collisions experienced by individual particles. The scalar k_n is a dimensionless quantity called the *Knudsen number* which is defined as follows

$$k_n := \frac{\lambda}{L} := \frac{\text{mean free path}}{\text{representative physical length scale}}.$$

The mean free path is the average distance a particle travels between collisions, given by

$$\lambda = \frac{k_B T}{\sqrt{2}\pi\sigma^2 P},$$

where k_B is Boltzmann's constant [Joule/Kelvin], T is temperature [Kelvin], σ is particle diameter [meters] and P is pressure [Pascal].

In all the following, we shall adopt the notation presented in [15], with small changes. The collision operator $Q(\,\cdot\,,\cdot\,)$ is derived from physical considerations and can be broken up into a gain and a loss term:

$$Q(f, f) = Q^+(f, f) - L[f] f, \tag{5}$$

where the gain and loss terms are defined by

$$Q^+(f, f) = \int_{\mathbb{R}^3} \int_{S^2} B(|v - v_*|, \cos\theta)\, f(v')\, f(v'_*)\, \mathrm{d}\omega\, \mathrm{d}v_*, \tag{6}$$

$$L[f] = \int_{\mathbb{R}^3} \int_{S^2} B(|v - v_*|, \cos\theta)\, f(v_*)\, \mathrm{d}\omega\, \mathrm{d}v_*. \tag{7}$$

Figure 1 illustrates the meaning of these variables. In particular, observe that ω is the (unit-length) particle velocity relative to path of the center of gravity of the system consisting of the two particles.

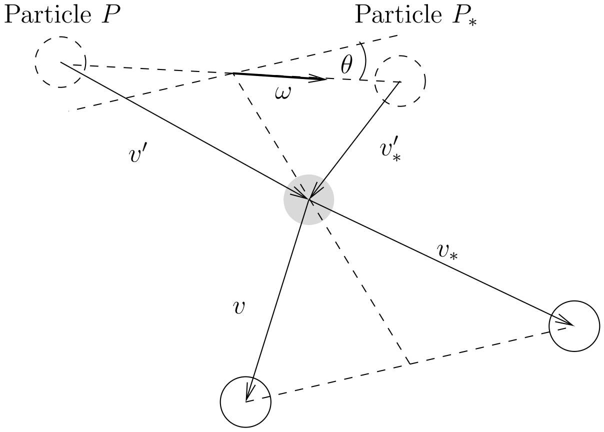

Figure 1. Illustration of the notation in (6) and (7).

The quantities v and v_* represent the velocities of two particles post-collision, and v' and v'_* are the same quantities pre-collision. Given the post-collision velocities and a deflection vector ω, the pre-collision velocities may be calculated by

$$v' = \frac{1}{2}(v + v_* + |v - v_*|\omega), \qquad v'_* = \frac{1}{2}(v + v_* - |v - v_*|\,\omega).$$

The angle θ is the angle between $v - v_*$ and $v' - v'_*$. To gain an intuitive insight, consider the special cases $\theta = 0°$ (head-on collision) and $\theta = 90°$ (one particle in motion, the other at rest). In particular,

$$\cos\theta = \frac{\omega \cdot (v - v_*)}{|v - v_*|}.$$

On the gain side, the term

$$B(|v - v_*|, \cos\theta)\, \underbrace{f(v')\, f(v'_*)}_{\text{availability}} \mathrm{d}\omega\, \mathrm{d}v_*$$

describes, for given post-collision velocities v and v_* and given deflection vector ω, “how many” particles (relatively) take part in this collision and acquire a new velocity v. Of course, the gain depends on “how many” particles of the pre-collision velocities are available, hence we multiply by the pre-collision densities

$$f(v') f(v'_*)$$

highlighted as “availability” above.

Similarly, on the loss side, the term

$$B(|v - v_*|, \cos\theta)\, \underbrace{f(v) f(v_*)}_{\text{availability}} \mathrm{d}\omega\, \mathrm{d}v_*$$

describes, this time for given *pre*-collision velocities v and v_* and given deflection vector ω, how many particles leave the current velocity state. Of couse, we again need to multiply by an availability factor, as highlighted above. Note that each of these considerations needs to be repeated at *each* point x in space.

Several examples (in order of increasing complexity) of kernels B are

- *Maxwellian gas:* $B(|v - v_*|, \cos\theta) = \text{const}$.
- *Hard Sphere gas:* $B(|v - v_*|, \cos\theta) = \text{const} \cdot |v - v_*|$.
- *Variable Hard Sphere (VHS) gas:* $B(|v - v_*|, \cos\theta) = \text{const} \cdot |v - v_*|^\alpha$. (generalizes both cases above)

Of course, one can cook up many more complicated collision kernels.

The Boltzmann equation (4) with nonzero Lorentz force L is sometimes called the Maxwell-Boltzmann system (or the Poisson-Boltzmann system for certain simplifications of L). Solving the Boltzmann equation is tantamount to:

- Solving the 6+1, space+time, nonlinear integrodifferential equation given by equation (4).
- Computing the collision operator (5) by performing the integrals in (6) and (7).
- Computing the charge and current densities given by (1).
- Solving the 3+1, space+time, linear hyperbolic system given by (2) and computing the Lorentz force (3).

In this survey, we shall only discuss methods to deal with the first two items: the Boltzmann equation (4) and the collision term (5). We will assume that the forcing function L can be readily computed by some means.

Even if we consider Lorentz force L as given, one can see that the Boltzmann equation is certainly one of the harder equations in scientific computing. In particular, it presents the following challenges:

- *Dimensionality*: The density function f that we seek to calculate is defined on a six-dimensional space. And as if that was not trouble enough, the collision operator contains another five-dimensional integral–for each point in six-dimensional space. A naive program would have to contain eleven nested loops–one over each dimension. For comparison: Modern supercomputers already choke on three-dimensional problems at moderate resolutions.

- *Timestepping*: It turns out that the collision operator and the advection require fairly different approaches to timestepping. How can the two be merged?

- *Collision Operator*: While we can't dodge the six dimensions of the space that f lives on, we need to be smart about carrying out the collision integrals. How can that be done?

- *Shock waves and Positivity for the Transport*: While the transport part of the equation is not the main consumer of computational time, it is still far from straightforward in that it, among other things, has to maintain positivity and resolve shock waves, while efficiently dealing with the high dimensionality of the data.

In this survey, we shall summarize some of the recent methods that have been used to overcome these difficulties and weigh the advantages and disadvantages of each. We will focus on deterministic methods, noting that these are best applied near an equilibrium state, when additional accuracy is desired. In non-equilibrium situations, or under less stringent accuracy requirements, particle methods may provide for far cheaper computations.

Because of the complexity and challenges which the full Boltzmann equation (4) presents, a very common strategy in the literature is to consider solving the transport and collision contributions separately, and once reliable methods have been developed for each of these problems, they can be combined in some fashion to form a full scheme for the complete Boltzmann equation. We shall refer to this general strategy of treating different parts of a PDE with differents schemes as a 'splitting' strategy. Splitting is the primary method we shall consider in this survey; however, this one class of solvers encompasses many schemes. To make the presentation as lucid as possible, we organize this survey as follows:

We start by including some methods for time-stepping in Section 2; these methods form the foundation of the splitting strategy, which is at the heart of every method presented here. In Section 3, we describe how one may handle the linear advection components of the Boltzmann equation (i.e. the Vlasov equation). In Section 4, we describe some methods to efficiently compute the collision operator (5).

2 Splitting Methods

The full Boltzmann equation (4) is a very challenging equation to solve: the transport and collision contributions must be solved in very different ways (see Sections 3 and 4). Because of this, we shall seek ways to solve the transport contribution, and the collision contribution individually. Once we have developed robust high-order methods to solve such systems, we shall attempt to fuse them together to form a complete scheme for the full Boltzmann equation. The numerical methods which serve as this algorithmic glue of the individual schemes are the time-stepping procedures described in this section.

In Sections 3 and 4 we will describe methods which are used to solve two separate PDE systems. In Section 3 describing the transport equation, we shall be concerned with solving

$$\frac{\partial f}{\partial t} = -v \cdot \nabla_x f - L \cdot \nabla_v f =: \mathcal{A}[f]. \tag{8}$$

Similarly, in Section 4 pertaining to collision we want to solve

$$\frac{\partial f}{\partial t} = \frac{1}{k_n} Q(f, f) =: \mathcal{B}[f], \tag{9}$$

where we recall that $Q(f, f)$ is the collision operator given by (5) and k_n is the Knudsen number, a physical quantity described in the introduction. However, we do not wish to solve (8) and (9) separately. Instead, we wish to solve the full Boltzmann equation (4), which we can rewrite as

$$\frac{\partial f}{\partial t} = \mathcal{A}[f] + \mathcal{B}[f]. \tag{10}$$

Unfortunately there is no simple way to solve (10) by solving (8) and (9) alone: suppose f_1 solves (8) and f_2 solves (9). Then unless $\mathcal{A}[f_2] = -\mathcal{B}[f_1]$, then $f_1 + f_2$ does not solve (10). Thus superposition is not a viable solution method.

Suppose instead that we just derive semi-discrete formulations for equations (8) and (9) separately and then combine the resulting right-hand-side phase-space discretizations. To be precise, consider a transport method, e.g. one of the methods discussed in Section 3 which has degrees of freedom $\{f_i^{\mathcal{A}}\}_{i \in I}$ and the semi-discrete form

$$\frac{\mathrm{d} f_i^{\mathcal{A}}}{\mathrm{d}t} = \mathcal{A}_h[f_i^{\mathcal{A}}] \quad \forall i \in I, \tag{11}$$

for some discretization of the transport operator $\mathcal{A}_h$. Consider also collision method chosen from e.g. Section 4 with degrees of freedom $\{f_i^{\mathcal{B}}\}_{j \in J}$ accompanied by the assoicated semi-discrete form

$$\frac{\mathrm{d} f_j^{\mathcal{B}}}{\mathrm{d}t} = \mathcal{B}_h[f_j^{\mathcal{B}}] \quad \forall j \in J. \tag{12}$$

One might suggest somehow melding these methods into one for the degrees of freedom $\{f_k\}_{k \in K}$ for the full Boltzmann equation (10) which reads

$$\frac{\mathrm{d} f_k}{\mathrm{d}t} = \mathcal{P}_i^k \mathcal{A}_h[f_i^{\mathcal{A}}] + \mathcal{P}_j^k \mathcal{B}_h[f_j^{\mathcal{B}}] \quad \forall k \in K, \tag{13}$$

where $\mathcal{P}_i^k$ is some transformation operator which maps the degrees of freedom $f_i^{\mathcal{A}}$ into the degrees of freedom f_k, and $\mathcal{P}_j^k$ is defined analogously. Indeed this is quite tempting to do until one realizes that there exists a great many problems with this formulation. Firstly, this is an evolution equation for the degrees of freedom f_k, whereas the right hand side is written in terms of $f_i^{\mathcal{A}}$ and $f_j^{\mathcal{B}}$. Thus, we need to somehow transform the degrees of freedom associated with (13) into those associated with (11) and (12). In other words, what are the transformation operators $\mathcal{P}_i^k$, $\mathcal{P}_j^k$ and their inverses $\mathcal{P}_k^i$, $\mathcal{P}_k^j$? If the methods do not share the same type of degrees of freedom (e.g. the transport discretization is discontinuous Galerkin and the collision scheme is a global spectral method), then transferring between the f_k, and the $f_i^{\mathcal{A}}$ and $f_j^{\mathcal{B}}$ may prove complicated and onerous.

In addition, one quickly realizes that many of the numerical methods presented in section 3 do not have a semi-discrete formulation (11). Indeed, many of the methods presented in that section integrate the degrees of freedom exactly in time: there is no semi-discrete form (11). The temporal approximation comes in solving the characteristic evolution equation (34). Methods like this cannot be molded into forms (11) because the degrees of freedom for (34) are not related to the density f. Thus, the formulation (13) fails for many of the methods described in Section 3.

Another drawback is that even if the transport and the collision degrees of freedom are the same (i.e. $f_i^{\mathcal{A}}$, $f_j^{\mathcal{B}}$, and f_k are all the same degrees of freedom), using one single explicit time-stepping procedure on (13) must adhere to the strictest stability criterion in both $\mathcal{A}_h$ and $\mathcal{B}_h$. To see why this may be a problem, let us generalize problem (13) a little: suppose $\mathcal{A}_h$ is the discretization of simple transport, whereas $\mathcal{B}_h$ is the discretization associated to a fifth derivative operator. Elementary experience in numerical analysis speaks that $\mathcal{B}_h$ will have eigenvalues which grow five times faster algebraically than $\mathcal{A}_h$. This means that guaranteeing the ODE notion of absolute stability for an explicit method requires a moderate timestep restriction on (11), but a fatally debilitating restriction on stepping (12) and (13). Essentially, we are working extremely hard for very little gain on $\mathcal{A}_h$ in equation (13). We might then suggest using an implicit method for (12) to avoid the timestep restriction, while using a high-order explicit method on (11) to maintain good accuracy for the transport term. However, using two separate time-stepping methods is not possible in a straightforward implementation of (13).

This leads us to the topic of this section: time-splitting methods. We suppose that the degrees of freedom $f_i^{\mathcal{A}}$ and $f_j^{\mathcal{B}}$ are the same type of discretization so that we may simply write $f_k^{\mathcal{A}}$ and $f_k^{\mathcal{B}}$ as the degrees of freedom. We also introduce a superscript indexed by n, the time step; i.e. f_k^n is an approximation to $f_k(t_n)$, where the time t is indexed by n. That is, we now suppose that we wish to solve

$$\frac{\mathrm{d}f_k}{\mathrm{d}t} = \mathcal{A}[f_k] + \mathcal{B}[f_k], \tag{14}$$

where we've dropped the subscript h on the operators $\mathcal{A}$ and $\mathcal{B}$ for readability. The augmented variables $f_k^{\mathcal{A}}$ and $f_k^{\mathcal{B}}$ are defined according to the evolution laws

$$\left.\begin{aligned} \frac{\mathrm{d}f_k^{\mathcal{A}}}{\mathrm{d}t} &= \mathcal{A}[f_k^{\mathcal{A}}] \\ \frac{\mathrm{d}f_k^{\mathcal{B}}}{\mathrm{d}t} &= \mathcal{B}[f_k^{\mathcal{B}}] \end{aligned}\right\} \tag{15}$$

Suppose we choose a separate time discretization for both equations (15), and let $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$ be the solution operators for the differential equations in (15). I.e. we can rewrite (15) as

$$\begin{aligned} f_k^{\mathcal{A}}(t^{n+1}) &= S_{\mathcal{A}}(f_k^{\mathcal{A}}(t^n); t_n, t_{n+1}) \\ f_k^{\mathcal{B}}(t^{n+1}) &= S_{\mathcal{B}}(f_k^{\mathcal{B}}(t^n); t_n, t_{n+1}). \end{aligned}$$

If the semi-discrete form is time-autonomous, then we simply take S as an operator with a superscript denoting the time step $t_{n+1} - t_n =: \Delta t$ and write

$$\left.\begin{aligned} f_k^{\mathcal{A}}(t^{n+1}) &= S_{\mathcal{A}}^{\Delta t} f_k^{\mathcal{A}}(t^n) \\ f_k^{\mathcal{B}}(t^{n+1}) &= S_{\mathcal{B}}^{\Delta t} f_k^{\mathcal{B}}(t^n) \end{aligned}\right\} \tag{16}$$

In addition, we introduce stepping operators $S_{\mathcal{A},n}^{\Delta t}$ and $S_{\mathcal{B},n}^{\Delta t}$ as the temporal discretizations associated with (15). Then we can write

$$\left.\begin{aligned} f_k^{\mathcal{A},n+1} &= S_{\mathcal{A},n}^{\Delta t} f_k^{\mathcal{A},n} \\ f_k^{\mathcal{B},n+1} &= S_{\mathcal{B},n}^{\Delta t} f_k^{\mathcal{B},n} \end{aligned}\right\} \tag{17}$$

The stepping operators do not have to be associated to any temporal discretization. For example, $S^{\Delta t}_{\mathcal{A},n}$ may be given by the flux balance method advancement in equation (38). The advantage of this formulation is that we have written the system of PDE's (8) and (9) in both semi-discrete form (16) and fully discrete (17). We shall use both the semi-discrete and the fully discrete formulations to describe our time-stepping procedures.

In the following sections, we shall assume that we are trying to solve the full system (10) so that the superscript $\mathcal{A}$ or $\mathcal{B}$ on the unknowns f_k becomes unnecessary as we are no longer solving two different equations (8) and (9). Thus, we shall write the semidiscrete degrees of freedom as simply $f_k(t)$ and the fully discrete ones as f_k^n.

2.1 Operator Splitting

Our first solution to the splitting problem is the operator splitting assertion. Assume we have a linear ODE

$$\frac{\mathrm{d}y}{\mathrm{d}t} = (A + B)y,$$

where A and B are some operators. The solution to this system can be written as

$$y = e^{(A+B)t}.$$

We wish to deal with A and B separately, so we may attempt to form a scheme which tries to approximate

$$e^{(A+B)t} \approx P(e^{At}, e^{Bt})$$

for some function P. That is, instead of computing $e^{(A+B)t}$, we instead compute e^{At} and e^{Bt} and combine them in some clever fashion so that we still obtain temporal accuracy and convergence. We now survey some of the most popular methods of this type that are used.

2.1.1 Simple First-Order

The simplest thing one could think of to employ both of the stepping operators $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$ is to use them in a sequential manner. I.e.

$$f_k(t^{n+1}) = S^{\Delta t}_{\mathcal{B}} S^{\Delta t}_{\mathcal{A}} f_k(t^n) \tag{18}$$

This amounts to solving the following ODE systems in a sequential manner at each time level:

$$\left.\begin{aligned} \frac{\mathrm{d}\tilde{f}_k(t)}{\mathrm{d}t} &= \mathcal{A}[\tilde{f}_k(t)] \\ \tilde{f}_k(t^n) &= f_k^n \end{aligned}\right\} \tag{19}$$

$$\left.\begin{aligned} \frac{\mathrm{d}f_k(t)}{\mathrm{d}t} &= \mathcal{B}[f_k(t)] \\ f_k(t^n) &= \tilde{f}_k(t^{n+1}) \end{aligned}\right\} \tag{20}$$

We can then set calculate $f_k(t^{n+1})$ as given by the solution to system (20). The advantage here is clear: we can independently apply methods from sections 3 and 4 to solve (19) and (20), respectively. Thus, all the analysis and validation from those sections apply here. The downside is that one can show that this simple method is only first-order accurate in time (see e.g. [38]). This is clearly undesireable: we have $O(N^6)$ unknowns for the full six-dimensional phase space. Thus, getting only a linearly decreasing error in time versus degrees of freedom is quite suboptimal and grossly computationally inefficient.

2.1.2 Second-Order Strang Splitting

A very popular alternative to the suboptimal first-order splitting technique presented in the previous section is a second-order in time splitting strategy first suggested by Strang in [48]. The main idea is to symmetrically straddle the operators on the temporal intervals. I.e.

$$f_k(t^{n+1}) = S_{\mathcal{A}}^{\Delta t/2} S_{\mathcal{B}}^{\Delta t} S_{\mathcal{A}}^{\Delta t/2} f_k(t^n) \tag{21}$$

Strang shows that this method is second-order accurate. Thus, as long as the fully discrete operators $S_{\mathcal{A},h}$ and $S_{\mathcal{B},h}$ are second-order accurate then the entire scheme will be second-order accurate in time. The Strang splitting technique is one of the most popular algorithms for solving this type of problem due to its simplicity and its relative accuracy.

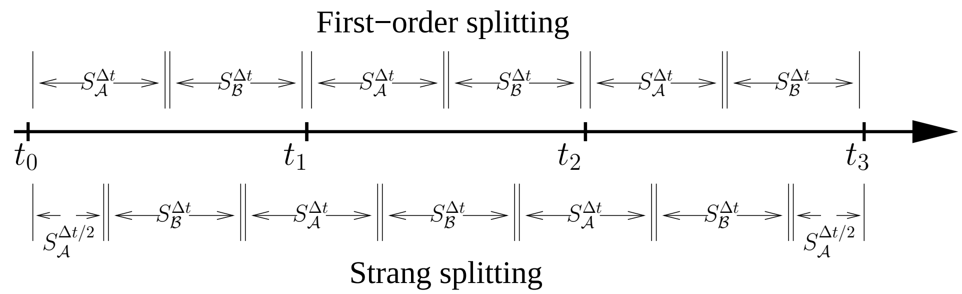
First−order splitting
$S_{\mathcal{A}}^{\Delta t}$
$S_{\mathcal{B}}^{\Delta t}$
$S_{\mathcal{A}}^{\Delta t}$
$S_{\mathcal{B}}^{\Delta t}$
$S_{\mathcal{A}}^{\Delta t}$
$S_{\mathcal{B}}^{\Delta t}$
t_0
t_1
t_2
t_3
$S_{\mathcal{A}}^{\Delta t/2}$
$S_{\mathcal{B}}^{\Delta t}$
$S_{\mathcal{A}}^{\Delta t}$
$S_{\mathcal{B}}^{\Delta t}$
$S_{\mathcal{A}}^{\Delta t}$
$S_{\mathcal{B}}^{\Delta t}$
$S_{\mathcal{A}}^{\Delta t/2}$
Strang splitting

Figure 2. An illustration of the first-order splitting method and the Strang splittng method.

2.1.3 More complex strategies

The Strang and first-order splitting methods are the most commonly employed methods. There are other methods available, but it turns out that these methods are more complicated and they do not provide any extraordinary accuracy. For example, there is another second-order splitting method in constrast to Strang's method due to Ohwada [38]. For this method, we explicity define the fully-discrete operator

$$S_{\mathcal{B},h}^{\Delta t} f_k^n = f_k^n + \Delta t\, \mathcal{B}[f_k^n],$$

which is basically a forward-Euler method. This allows us to define the scheme as

$$\begin{aligned} f_k^{n+1} &= S_{\mathcal{A}}^{\Delta t} f_k^n + \frac{1}{2}\left(S_{\mathcal{B},h}^{\Delta t} S_{\mathcal{B},h}^{\Delta t} S_{\mathcal{A}}^{\Delta t} f_k^n - S_{\mathcal{B},h}^{\Delta t} S_{\mathcal{A}}^{\Delta t} f_k^n\right) + \frac{1}{2}\left(S_{\mathcal{B},h}^{\Delta t} S_{\mathcal{A}}^{\Delta t} f_k^n - S_{\mathcal{A}}^{\Delta t} f_k^n\right) \\ &= \frac{1}{2}\left(\mathcal{I} + S_{\mathcal{B},h}^{\Delta t} S_{\mathcal{B},h}^{\Delta t}\right) S_{\mathcal{A}}^{\Delta t} f_k^n. \end{aligned} \tag{22}$$

Ohwada shows that this scheme is second-order accurate, and that it preserves mass, momentum, and energy in the context of the Boltzmann equation. Note that this fully-discrete scheme uses the exact solution operator $S_{\mathcal{A}}^{\Delta t}$ and not some temporal discretization of it. However, given that $\mathcal{A}$ is simply a convection, this can be solved exactly in principle. The Ohwada splitting scheme (22) has roughly the same computational complexity as the Strang splitting, with the advantage of conservation of relavant quantities for the Boltzmann equation.

There are also other types of higher-order splitting methods in e.g. [13]. For example, we can can consider two different operator splitting schemes of the form

$$f_k(t^{n+1}) = \frac{1}{3}\Big(4\, S_{\mathcal{A}}^{\Delta t/4} S_{\mathcal{B}}^{\Delta t/2} S_{\mathcal{A}}^{\Delta t/2} S_{\mathcal{B}}^{\Delta t/2} S_{\mathcal{A}}^{\Delta t/4} - S_{\mathcal{A}}^{\Delta t/2} S_{\mathcal{B}}^{\Delta t} S_{\mathcal{A}}^{\Delta t/2}\Big) f_k(t^n) \tag{23}$$

$$f_k(t^{n+1}) = \frac{1}{6}\Big(4\, S_{\mathcal{A}}^{\Delta t/2} S_{\mathcal{B}}^{\Delta t} S_{\mathcal{A}}^{\Delta t/2} + 4 S_{\mathcal{B}}^{\Delta t/2} S_{\mathcal{A}}^{\Delta t} S_{\mathcal{B}}^{\Delta t/2} - 2\, S_{\mathcal{A}}^{\Delta t} S_{\mathcal{B}}^{\Delta t} - S_{\mathcal{B}}^{\Delta t} S_{\mathcal{A}}^{\Delta t}\Big) f_k(t^n) \tag{24}$$

Dia [13] shows that these splitting strategies are fourth-, and third-order accurate in time, respectively. However, they suffer from some stability issues and thus are not used very extensively. In addition, evaluating $S_{\mathcal{B}}^{\Delta t}$ multiple times can be a computational burden.

2.2 Implicit-Explicit Methods

To avoid using the operator splitting strategy, some have devised implicit-explict Runge-Kutta (IMEX-RK) schemes which appear to boast the advantages of the splitting strategy without the unfortunate restriction to second-order. These methods were first introduced by Ascher et al. in [2]. The basic idea of these methods is as follows: recall that we have an ODE system

$$\frac{\mathrm{d}f_k}{\mathrm{d}t} = \mathcal{A}_h[f_k] + \mathcal{B}_h[f_k]. \tag{25}$$

The operator-splitting approach sought to find solution operators for each right-hand side contribution individually and then to apply them in a clever way to attain consistency and accuracy. The IMEX-RK approach is exactly the same, except that we shall introduce a very general mathematical framework by which we can explicitly derive schemes. In order to conform to the notation in the literature, we rewrite (25) in the form

$$y' = f(y) + g(y), \tag{26}$$

where $f(y)$ is some non-stiff, possibly highly nonlinear, term and $g(y)$ is a stiff, but 'close' to linear, term. The stiffness of $g(y)$ can be attributed to implicit dependence on some relaxation parameter ε; we shall not discuss the effect of the degree of stiffness on the numerical solution of (26), but in general, we should replace $g(y)$ with $\frac{1}{\varepsilon}g(y)$, and consider stability and accuracy as functions of ε, which controls the stiffness of the system. However, we shall make only brief mentions of the relaxation parameter ε in the future, and so omit its explicit role in equation (26). For the Boltzmann equation, the relaxation parameter ε is the Knudsen number, previously labelled k_n.

For example, equation (26) may arise from some convection-diffusion PDE, with $f(y)$ representing the semi-discrete form of the convection, and $g(y)$ representing the semi-discrete form of the diffusion. In this way, diffusive second-derivative terms are more stiff than convective terms. In the context of the Boltzmann equation, $f(y)$ represents the convective Vlasov-equation terms, and $g(y)$ represents the possibly stiff collision operator. Note that we have, without loss, suppressed any non-autonomous dependence in (26): a non-autonomous system can always be written as an autonomous one, but in any case we shall develop the IMEX-RK methods with non-autonomous generality. In the following, we write $f = f(t, y)$ and $g = g(t, y)$.

An IMEX-RK scheme is defined by the number of *stages* s; by four $s \times 1$ vectors b, $\tilde{b}$, c, and $\tilde{c}$; and by two $s \times s$ matrices a and $\tilde{a}$. The method takes the unknowns y^n at time level t^n to time level $t^{n+1} = t^n +$

Δt via the scheme

$$y^{n+1} = y^n + \Delta t \sum_{j=1}^{s} \tilde{b}_j\, f(t^n + \tilde{c}_j\, \Delta t, Y_j) + \Delta t \sum_{j=1}^{s} b_j\, g(t^n + c_j\, \Delta t, Y_j), \tag{27}$$

where the intermediate stage variables $\{Y_i\}_{i=1}^{s}$ are given by

$$Y_i = y^n + \Delta t \sum_{j=1}^{s} \tilde{a}_{ij} f(t^n + \tilde{c}_j\, \Delta t, Y_j) + \Delta t \sum_{j=1}^{s} a_{ij}\, g(t^n + c_j \Delta t, Y_j). \tag{28}$$

Equations (27) and (28) form the IMEX-RK scheme. We must take some time to discuss a few observations: All we have done is written down a generalization of the usual RK scheme. Indeed, if $b = \tilde{b}$, $c = \tilde{c}$, and $a = \tilde{a}$, then the IMEX-RK scheme degenerates into a familiar s-stage RK scheme. The weights a, b, and c, and their tilde'd brothers are usually arranged in a *Butcher tableau* [9] as depicted in Figure 3.

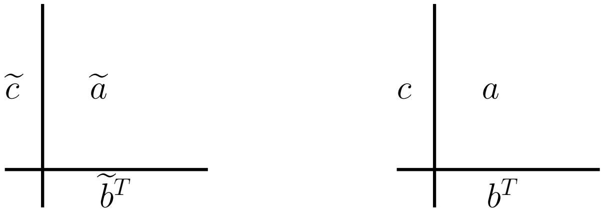

Figure 3. Arrangement of Butcher tableaux.

Recall that our motivation for this particular type of splitting is that the term f is usually a relatively-straightforward part to integrate with an explicit method. Thus, we shall define the tableau for f (the tilde'd variables) to be an explicit RK method. Recall that in the arrangement of the tableau, the nonzero entries of the matrix $\tilde{a}$ determine the implicit or explicit nature of a method:

- An *explicit* method has zero entries on and above the main diagonal: $\tilde{a}_{ij} = 0$ for $j \geq i$.
- A *diagonally implicit* method has zeros above the main diagonal: $\tilde{a}_{ij} = 0$ for $j > i$.
- A method not satisfying any of the above two restrictions is called fully *implicit*.

We shall restrict our attention to solving f via an explicit method. Thus, we hereafter assume that $\tilde{a}_{ij} = 0$ for $j \geq i$. In addition, although we can in principle allow g to be solved with any fully implicit method, practical considerations may become problematic if g is not strictly linear. Thus, we also assume that g is solved with a diagonally implicit Runge-Kutta (DIRK) method, which satisfies $a_{ij} = 0$ for $j > i$.

The purpose of this survey is not to describe the theory of IMEX-RK methods. All that we shall say is that in order to determine the order of accuracy for traditional RK schemes, the essential method is Taylor expansion: the exact solution is substituted into the numerical scheme, is Taylor expanded, and the highest-magnitude Taylor expansion term which is not eliminated by the scheme dictates the order of convergence in Δt. However, these IMEX-RK methods are not traditional RK methods; in fact, one can view these split methods as special cases of *partitioned* RK methods [28]. Thus, we can apply the order condition derivation presented by Hairer in [25] to determine the order of convergence of these IMEX-RK methods. In fact, Pareschi and Russo derive some of these conditions in [40], which we present below to illustrate the characteristics of the method.

2.2.1 Order of Convergence

Before providing some examples of these methods, we shall show and interpret some of the order conditions for IMEX-RK schemes that have been derived in [40]. We suppose that we are looking for a scheme of order p convergence in Δt. This requirement leads to a set of conditions on the values of the elements in the Butcher tableau. We present conditions for the first three orders below:

- $p=1$. The elements in the Butcher tableau in figure 3 must satisfy

$$\sum_{i=1}^{s} \tilde{b}_i = 1 \qquad \sum_{i=1}^{s} b_i = 1$$

- $p=2$. The elements in the Butcher tableau in figure 3 must satisfy

$$\sum_{i=1}^{s} \tilde{b}_i\, \tilde{c}_i = \frac{1}{2} \qquad \sum_{i=1}^{s} b_i\, c_i = \frac{1}{2}$$

$$\sum_{i=1}^{s} \tilde{b}_i\, c_i = \frac{1}{2} \qquad \sum_{i=1}^{s} b_i\, \tilde{c}_i = \frac{1}{2}$$

- $p=3$. The elements in the Butcher tableau in figure 3 must satisfy

$$\sum_{i,j=1}^{s} \tilde{b}_i\, \tilde{a}_{ij}\, \tilde{c}_j = \frac{1}{6} \qquad \sum_{i,j=1}^{s} b_i\, a_{ij}\, c_j = \frac{1}{6} \qquad \sum_{i=1}^{s} \tilde{b}_i\, \tilde{c}_i\, \tilde{c}_i = \frac{1}{3} \qquad \sum_{i=1}^{s} b_i\, c_i\, c_i = \frac{1}{3}$$

$$\sum_{i,j=1}^{s} \tilde{b}_i\, a_{ij}\, c_j = \frac{1}{6} \qquad \sum_{i,j=1}^{s} b_i\, \tilde{a}_{ij}\, c_j = \frac{1}{6} \qquad \sum_{i,j=1}^{s} b_i\, a_{ij}\, \tilde{c}_j = \frac{1}{6} \qquad \sum_{i,j=1}^{s} \tilde{b}_i\, \tilde{a}_{ij}\, c_j = \frac{1}{6}$$

$$\sum_{i,j=1}^{s} b_i\, \tilde{a}_{ij}\, \tilde{c}_j = \frac{1}{6} \qquad \sum_{i,j=1}^{s} \tilde{b}_i\, a_{ij}\, \tilde{c}_j = \frac{1}{6}$$

$$\sum_{i=1}^{s} \tilde{b}_i\, c_i\, c_i = \frac{1}{3} \qquad \sum_{i=1}^{s} b_i\, \tilde{c}_i\, c_i = \frac{1}{3} \qquad \sum_{i=1}^{s} \tilde{b}_i\, \tilde{c}_i\, c_i = \frac{1}{3} \qquad \sum_{i=1}^{s} b_i\, \tilde{c}_i\, \tilde{c}_i = \frac{1}{3}$$

We can observe some very interesting properties from the order conditions presented above: for first-order approximations consistency of the entire method is guaranteed by consistency of the individual methods. I.e., the requirement for first-order consistency places no restriction on the RK interaction between the tilde'd and non-tilde'd schemes. For second order, the first line of equations is simply the usual second-order consistency requirements for RK schemes; the second line of equations are the consistency requirements because of the coupling of the schemes. The same can be seen with third-order $p=3$: the first line of equations are the natural third-order RK consistency requirements, whereas all the subsequent requirements arise due to the implicit-explicit scheme coupling.

Of course, the error term in deriving these schemes depends heavily on the stiffness in the term g (i.e. depends on the relaxation parameter ε). In order to control stiffness of the system, Pareschi and Russo [40] show that L-stability of the implicit RK scheme guarantees the advertised order of convergence even for very stiff terms.

2.2.2 Examples

In this section we shall present some examples of IMEX-RK methods. We introduce some notation: we let the triplet (s, σ, p) represent the number of implicit stages, number of explicit stages, and the convergence order, respectively. (Note that in our presentation, the number of implicit and explicit stages are the same, but one may introduce intermediate implicit-only stages and then pad the explicit tableau with rows of zeros.) Figure 4 shows two relatively simple IMEX-RK methods: the first is a standard splitting method in which forward-Euler is performed on the explicit part f and backward-Euler is performed on the implicit part g. Application of formulas (28) and (27) show that the standard splitting (1,1,1) scheme is equivalent to first-order operator splitting given in (18) with $S_{\mathcal{A},h}$ being forward-Euler and $S_{\mathcal{B},h}$ being backward-Euler.

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array} \qquad \begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}$$

Standard splitting (1,1,1)

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad \begin{array}{c|cc} -1 & -1 & 0 \\ 2 & 1 & 1 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Jin splitting (2,2,2)

Figure 4. Two examples of splitting methods. Explicit tableaus (left) and implicit tableaus (right).

In figure 4 we also show a second order scheme due to Jin [31]. For many higher-order schemes and further analysis, we refer the reader to [40] and [7].

It turns out that many general single-step splitting schemes can be formulated as IMEX-RK schemes. Simple first-order splitting (18) and Strang splitting (21) can be re-cast as IMEX-RK methods. In addition, Jin's (2,2,2) scheme was actually first proposed as an operator-splitting type method.

2.3 Other Methods

There have been a great number of other methods which have been proposed; some for solving ODE's of the mixed type (26), some specifically designed for the Boltzmann equation.

Gerisch and Weiner [21] derive some explicit RK methods which they can show preserves positivity of the solution. Of course, this can be of monumental importance in solving the Boltzmann equation which has a non-negative density function as the unknown. The downside is that these methods do not offer much in the way of dealing with stiff terms.

If we consider equation (26) as a function of the relaxation parameter ε, then we can consider what should happen in the limit as $\varepsilon \to \infty$, i.e. as the ODE system becomes a semi-discrete form of a hyperbolic conservation law. It is reasonable that we ask that in this limit, we obtain some temporal discretization which preserves some properties which we desire when simulating a hyperbolic conservation law, e.g. the strong stability-preserving (SSP) property. (Formerly called the total variation diminishing property, [23].) It turns out that such schemes, which are IMEX-RK schemes (and hence retain their robustness in the stiff regime under the assumption of L-stability), yet preserve the SSP property in the limit of the hyperbolic conservation law, do exist. They are described in a later paper by Pareschi and Russo, [42].

We have only described *single-stage* and *multi-stage* methods in this survey, but there are also *multi-step* methods, which require accessing the value of the unknown at previous time-steps. These methods have IMEX variations [3] and SSP-IMEX variations [22].

As a final remark, we have only considered splitting-type algorithms for solving ODE's. The motivation for this stems from the complex nature of the Boltzmann equation. Of course one may use any of the plethora of ODE schemes that exist in order to time-step the Boltzmann equation. We have presented operator splitting methods and IMEX-RK methods, and their variants. Having now described how we connect split transport and collision steps in a time-discretization, we now turn to the problem of actually solving the PDEs associated with each of those operators individually.

3 Transport

In this section, we shall assume that we wish to solve a transport equation of the form

$$\frac{\partial f}{\partial t} + v \cdot \nabla_x f + L \cdot \nabla_v f = 0. \tag{29}$$

where f, the density function, depends on seven independent variables (including time), and L is the spatially-dependent forcing function (e.g. the electromagnetic Lorentz force). This equation models the natural inertia of the system with external forces, and is called the collisionless Boltzmann equation, or more frequently, the *Vlasov equation*. It is an ordinary advection equation, except that we desire to solve this in the full 6-dimensional (x, v) phase space. The equation is implicity nonlinear since the forcing function L does depend on the density f, see Section 1. This equation arises naturally in many problems of plasma physics.

In Section 2 we described many methods of time-stepping once a semi-discrete or fully discrete form of (29) has been found. This survey cannot possibly do even a fraction of the deserved justice to the voluminous literature on the subject of finding a semi-discrete form for a nonlinear hyperbolic conservation law. (v is x-divergence free, and L is v-divergence free, so (29) is easily recast into a conservation law form with six 'spatial' dimensions, x-v phase space.) We can only mention a brief selection of these methods and point to a few references that serve as a very small pebble on the tip of a very large iceberg:

Conservation laws may be converted into semi-discrete form with finite-difference [24], finite-volume [34], finite-element [8], or spectral [26] degrees of freedom. Discontinuous Galerkin methods [12, 27] and nonlinear reconstruction techniques like ENO/WENO in [46] provide robust, high-order solvers for nonlinear problems.

Once a semi-discrete method has been found, one can use this in tandem with a solver of choice for the collision term (see Section 4) and apply e.g. an IMEX-RK method, as described in Section 2. Of course, one may also choose to adopt an operator-splitting technique, but this requires some choice of time discretization. For methods which have an appropriate semi-discrete form, the transport step has a fairly well-defined shape.

Thus, instead of focusing on well-explored ways to turn (29) into a semi-discrete scheme non-specific to the Boltzmann equation, we shall concentrate on:

- Fully-discrete schemes.
- Semi-discrete schemes specifically tailored for Vlasov-type equations.

Among the fully-discrete methods we shall consider are the semi-Lagrangian method (discussed in [4] and [17]) and the flux balance method (introduced in [14]). Some semi-discrete type schemes tailored for the Vlasov equation include Gamba and Proft's DG scheme [20] and Klimas's filtered spectral method [32].

3.1 Desired Scheme Properties

We recall that we are solving equation (29). The density function is a physical quantity, and we therefore ask our numerical approximation of the density to satisfy certain properties:

- Positivity: $f(x, v, t) \geq 0 \ \forall \ x, v, t$
- Conservation
  - Mass: $\int f \, \mathrm{d}v \, \mathrm{d}x = C$ for some constant scalar C independent of time

- ○ Momentum: $\int v\, f\, \mathrm{d}v\, \mathrm{d}x = C$ for some constant vector C independent of time
- ○ Energy: $\int \ |v|^2\, f\, \mathrm{d}v\, \mathrm{d}x = C$ for some constant scalar C independent of time

Unfortunately, these restrictions (especially positivity) prevent us from blindly applying the usual well-developed numerical schemes for hyperbolic conservation laws. In addition, conservation of momentum and energy is an extremely difficult task and few numerical methods we present here actually address these goals. The following sections describe some of the high-order accurate numerical methods that have been presented to deal with these difficulties.

3.2 More Splitting

Solving equation (29) requires a six-dimensional solver which can be quite complex to formulate and code. Because of this many have observed that, as we did in Section 2, we can take advantage of operator splitting to simplify our approach. To be precise, we can split the physical space x-advection and the velocity space v-advection. If we apply this splitting, then we obtain the collection of equations

$$\left.\begin{aligned} \frac{\partial f}{\partial t} + v \cdot \nabla_x f = 0 \\ \\ \frac{\partial f}{\partial t} + L \cdot \nabla_v f = 0 \end{aligned}\right\} \tag{30}$$

However, since v is independent of x and L is independent of v, then each of these equations is a scalar multidimensional hyperbolic law of the form

$$\frac{\partial u}{\partial t} + c \cdot \nabla u = 0. \tag{31}$$

That is, we are dealing with constant-coefficient advection. (Note that L is not really independent of f, so it's not strictly linear; however, the dependence on f is very complicated, and so we shall consider L as a given function.) We can then solve each equation in (30) separately and then use an operator splitting approach to advance the density in time.

Note that even devising methods for solving (31) is not trivial since we ask for our positivity and conservation goals and u is a function of three spatial variables. Many of the methods we shall present below are not written as generalized for three dimensions. However, one may consider implementing the below schemes on tensor-product meshes, which is a straightforward formulation for some; one may also consider yet another splitting strategy in which separate schemes deal with only one-dimensional advection.

3.3 Flux Balance Methods

When we consider discretizing the full (6+1)-dimensional Vlasov equation (29), we first start solving one-dimensional problems of the form

$$\frac{\partial f}{\partial t} + \frac{\partial (v\, f)}{\partial x} = 0. \tag{32}$$

for some velocity $v(x, t)$, and $f(x, t)$, $x \in \mathbb{R}$. (Although it is clearly the case from (29) that v is independent of x and t, we shall assume some generality here.) We shall solve this system exactly using the method of characteristics. If we are given the solution at some time t^n, then we can easily write down the solution at time step $n+1$ as

$$f(x, t^{n+1}) = f(X(t^n; t^{n+1}, x), t^n). \tag{33}$$

where we define $X(\,\cdot\,; t^{n+1}, x)$ as the solution to the ODE

$$\begin{aligned} \frac{\mathrm{d}X(s)}{\mathrm{d}x} &= v(X(s), s) \\ X(t^{n+1}) &= x. \end{aligned} \tag{34}$$

Intuitively, $X(t^n; t^{n+1}, x)$ means 'the starting location at time t^n of a particle that ends up at x at time t^{n+1} under the influence of the velocity field v'. Mathematically, it means 'the solution $X(t^n)$ to the evolution law (34) with initial condition $X(t^{n+1}) = x$'. Refer to Figure 5 for a pictorial explanation.

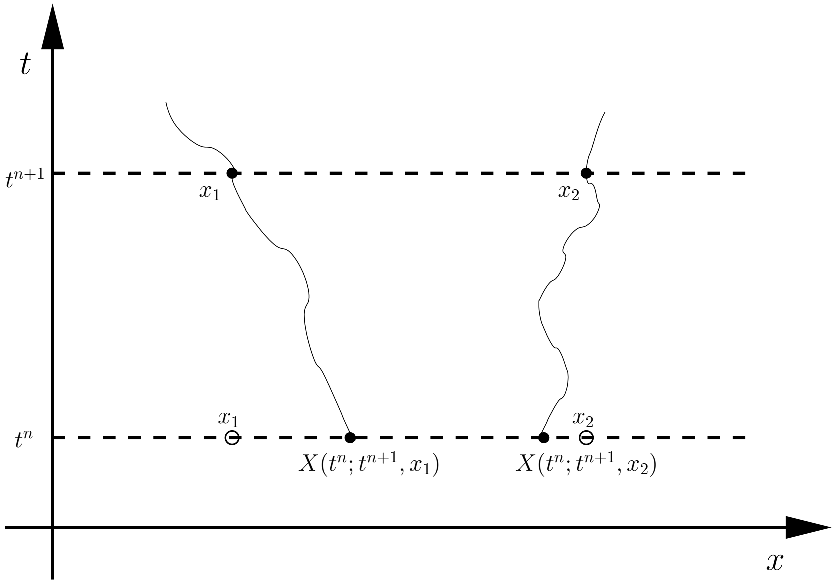

Figure 5. Pictorial representation of the characteristic flow governed by the evolution law (34).

We suppose that the coordinate direction x is divided into a finite number of non-overlapping cells indexed by i whose interfaces are given by the ordered nodal points $x_{i\pm1/2}$, and whose centers are given by $x_i = \frac{1}{2}\big(x_{i-1/2} + x_{i+1/2}\big)$. For the flux balance method (FBM), we shall consider the cell averages as degrees of freedom:

$$f_i^n = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} f(t^n, x)\,\mathrm{d}x$$

and introducing the flux terms

$$\Phi_{i+1/2}(t^n) = \int_{x_{i+1/2} - v\Delta t}^{x_{i+1/2}} f(t^n, x)\,\mathrm{d}x, \tag{35}$$

We note here that elementary calculus tells us that the flux Φ can be computed using the primitive of f:

$$\Phi_{i+1/2}(t^n) = F(t^n, x_{i+1/2}) - F(t^n, x_{i+1/2 - v\Delta t}) \tag{36}$$

The primitive function F satisfies the properties $F(t^n, x_{i+1/2}) - F(t^n, x_{i-1/2}) = \Delta x\, f_i^n$ and $F(t^n, x_{i+1/2}) = \Delta x \sum_{k=0}^{i} f_k^n =: w_i^n$. Assuming v is divergence-free (which it is in the Vlasov equation), it follows from the conservation law formulation (32) that f is conserved along characteristic boundaries. That is,

$$\int_{x_{i-1/2}}^{x_{i+1/2}} f(t^{n+1}, x)\,\mathrm{d}x = \int_{X(t^n;t^{n+1},x_{i-1/2})}^{X(t^n,t^{n+1},x_{i+1/2})} f(t^n, x)\,\mathrm{d}x, \tag{37}$$

which we may write in a finite volume framework as

$$f_i^{n+1} = f_i^n + \frac{\Phi_{i-1/2}(t^n) - \Phi_{i+1/2}(t^n)}{\Delta x}. \tag{38}$$

Equation (38) forms the basis for the solvers we shall discuss in this section. The flux terms Φ are computed using either (35) or (36), and the *modus operandi* for computing these fluxes is what differentiates each of the methods below.

3.3.1 The linear FBM

The FBM scheme (38) was introduced by Fijalkow in [14], where a linear interpolation procedure was used to determine a numerical reconstruction of the density f_h:

$$f_h(x) = f_i + (x - x_i)\,\frac{f_{i+1} - f_{i-1}}{2\,\Delta x}. \tag{39}$$

Using this reconstruction $f_h(x)$ at time level t^n, one can compute the fluxes explicitly using (35). The great advantage of this method is the ease of implementation. However, the main drawback is that positivity of the density is nowhere preserved, and spurious oscillations may develop.

3.3.2 The Positive and Flux Conservative Method

One well-known method to control positivity of the solution was introduced in [18]. The authors mention that using the property of the primitive function

$$F(t^n, x_{i+1/2}) = \Delta x \sum_{k=0}^{i} f_k^n \tag{40}$$

one may use as large a centered finite volume stencil as desired to reconstruct an n^{th} order primitive polynomial, which by differentiation yields an n^{th} order accurate reconstruction of f, which may then be used to explicitly evaluate the integral in (35). However, there is again no way of preserving positivity with this method, and it will not even control spurious oscillations if the reconstructing stencil is not chosen in an adaptive way as in e.g. ENO [46].

In order to control the positivity, we adopt the e.g. third-order fixed-stencil reconstruction previously described, but nonlinear slope limiters $\varepsilon_i^{\pm}$ are introduced (see [17], [18] for details). The result is that the density reconstruction $f_h(x)$ has the form

$$\begin{aligned} f_h(x) \;\; = \;\; & f_i + \frac{\varepsilon_i^+}{6\Delta x^2}\big[2(x - x_i)(x - x_{i-3/2}) + (x - x_{i-1/2})(x - x_{i+1/2})\big](f_{i+1} - f_i) \\ & - \frac{\varepsilon_i^-}{6\Delta x^2}\big[2(x - x_i)(x - x_{i-3/2}) + (x - x_{i-1/2})(x - x_{i+1/2})\big](f_i - f_{i-1}). \end{aligned} \tag{41}$$

And the slope limiters are defined by

$$\varepsilon_i^{\pm} = \begin{cases} \min\left\{1, \dfrac{2 f_i}{f_{i\pm1} - f_i}\right\} & \text{if } \; f_{i\pm1} - f_i > 0, \\ \min\left\{1, -2\,\dfrac{f_\infty - f_i}{f_{i\pm1} - f_i}\right\} & \text{if } \; f_{i\pm1} - f_i > 0, \end{cases}$$

where $f_\infty = \max_i f_i$ is the maximum cell average value.

Using this reconstruction for the density allows for integration to find the value of the primitive. The authors of [18] show that using this as the reconstruction of the density preserves both positivity and conservation of the average. That is, $f_h(x)$ satisfies

- *Conservation*: for every i, $\int_{x_{i-1/2}}^{x_{i+1/2}} f_h(t^n, x)\,\mathrm{d}x = \Delta x\, f_i^n$

- *Positivity:* for all x, $0 \leq f_h(t^n, x)$

With this well-behaved reconstruction for f_h given by (41), one may then proceed to explicitly evaluate the integral (35) to advance the scheme (38); this method is called the positive and flux conservative (PFC) method. One can prove first-order convergence for this method. One-dimensional tests shown in [18] indicate that this scheme seems to control oscillations as well as ENO, and 2D tests confirm the validity of the procedure in higher dimensions.

A possible generalization of this method is to unify both the well-accepted oscillation-controlling properties of the ENO scheme with the provable positivity and conservation of the PFC method: the above method fixes a central stencil for the third-order reconstruction. One may consider instead using an adaptively-chosen stencil prescribed by the ENO procedure, and then use a reconstruction similar to (41) with the appropriately modified slope correctors. Although the PFC scheme has already heuristically shown its ability to control oscillations, this adaptation may prove a profitable union of the positivity, conservation, and oscillation-free properties.

3.3.3 PWENO Reconstruction

Very recently, there have been some researchers who have applied the 'pointwise' WENO (PWENO) reconstruction procedure outlined in [44] to reconstruct the value of the primitive function to evaluate (36). They present their formulation and results in [10]. We shall briefly describe the PWENO procedure here.

We recall that in our finite volume framework, the cell boundaries are denoted by $x_{i-1/2}$ and our unknowns are the cell averages f_i^n. Equation (36) requires the evaluate of $F(t^n, x)$ for some value of x. Note that we can explicity calculate $F(t^n, x_{i-1/2})$ for all i using (40). Thus, our PWENO problem is: calculate $F(x)$ given $F(x_{i-1/2})$ for all i. Given a point $x \in (x_{i-1/2},\ x_{i+1/2})$ we assume that we fix some 'ambient' stencil S: the collection of degrees of freedom which we will use to create a reconstruction. In this example, we shall use an ambient 5-point stencil. We associate with this ambient stencil three smaller 3-point stencils S_r which are contained inside the ambient stencil S. See figure 6.

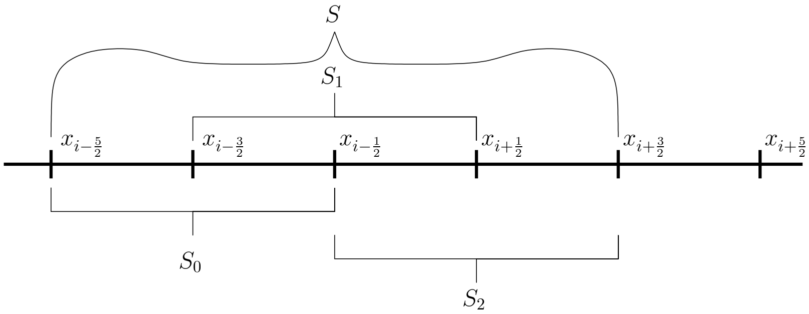

Figure 6. Depiction of a stencil setup for an ambient 5-point stencil comprised of three 3-point stencils.

The ENO/WENO schemes first introduced were concerned with reconstructing the function f at the point $x = x_{i\pm1/2}$. The algorithm requires the construction of polynomials approximating the primitive F on each sub-stencil. Notationally, we shall denote $P_r(x)$ as the polynomial reconstruction from the degrees of freedom $F(x_{i-1/2})$ on stencil S_r. Then, for example,

$$P_0(x) = \sum_{k=0}^{2} F[x_{i-5/2}, ..., x_{i+l-5/2}] \prod_{l=0}^{k-1} (x - x_{i-5/2+l}), \qquad (42)$$

is the reconstruction on stencil S_0 where the Newton divided differences are calculated via the definition

$$F[x_i, ..., x_{i+p}] = \frac{F[x_{i+1}, ..., x_{i+p}] - F[x_i, ...x_{i+p-1}]}{x_{i+p} - x_i},$$

and the index i may be shifted to e.g. $i - 5/2$ for the calculation of $P_0(x)$. The divided difference $F[x_i] := F(x_i)$. To then obtain the reconstruction for our desired function of interest, f, we can differentiate (42) to obtain

$$p_0(x) = \sum_{k=1}^{2} F[x_{i-5/2}, ..., x_{i+l-5/2}] \sum_{l=0}^{k-1} \prod_{\substack{m=0 \\ m \neq l}}^{k-1} (x - x_{i-5/2+m})$$

We now have a representation for our function of interest which depends only the divided differences of F of the first order or higher. We can exploit this by noticing that the first-order divided difference F is

$$F[x_{i-1/2}, x_{i+1/2}] = \frac{F(x_{i+1/2}) - F(x_{i-1/2})}{x_{i+1/2} - x_{i-1/2}} = f_i$$

In other words, we can find reconstructions for f, e.g. p_0, without considering the primitive function at all.

The ENO procedure reads: determine which stencil S_r admits the least-oscillatory reconstruction $p_i(x)$, and use that stencil to evaluate the reconstructed value. This is done by starting with a stencil size of 1, and increasing the stencil size by including the adjacent point which has the smallest divided difference (a mesaure of the oscillatory behavior of the reconstructed polynomial). More points are added until the intended stencil size is reached (for this example, 3 points). Once the stencil is computed, determine the reconstructing polynomial For example, suppose we wish to determine $f(x_{i-1/2})$. The ENO reconstruction reads:

1. Start with $\tilde{S} = \{x_{i-3/2}, \ x_{i-1/2}\}$, and specify a maximum stencil size of 3 points. Compute $F[x_{i-5/2}, x_{i-3/2}, x_{i-1/2}]$ and $F[x_{i-3/2}, x_{i-1/2}, x_{i+1/2}]$.

2. Suppose $F[x_{i-5/2}, \ x_{i-3/2}, \ x_{i-1/2}]$ is smaller in magnitude. Then add $x_{i-5/2}$ to the stencil so that $\tilde{S} = \{x_{i-5/2}, x_{i-3/2}, x_{i-1/2}\}$.

3. The maximum stencil size is reached and $\tilde{S} = S_0$. Reconstruct the polynomial $p_0(x)$ and evaluate it at $x = x_{i-1/2}$. Then $p(x_{i-1/2})$ is the desired value.

A generalization of the ENO procedure is the WENO procedure, which uses information from the entire ambient stencil S, rather than just one sub-stencil S_r. The ENO procedure has a sharp criterion for using a particular reconstruction: either one stencil is used entirely, or it is not used at all. The disadvantage of this is that, in the example in figure 6, we use information from 5 points to reconstruct something with 3 points of accuracy. In an effort to increase accuracy for a given ambient stencil, WENO introduces the convex weights d_r which are chosen so that

$$f_h(x_{i-1/2}) = \sum_{r=0}^{2} d_r p_r(x_{i-1/2}) + O(\Delta x^5),$$

and

$$\sum_{r=0}^{2} d_r = 1.$$

One can compute these weights readily enough based on Taylor expansions. However, we have simply computed a high-order reconstruction. This is what we would like the convex combination to behave like in smooth regions. To include some of the oscillation-reducing qualities of the ENO procedure in the vicinity of shocks, we introduce the smoothness indicators β_r associated with each reconstruction $p_r(x)$, which we define as

$$\beta_r = \sum_{l=1}^{k-1} \int_{x_{i-1/2}}^{x_{i+1/2}} \Delta x^{2l-1} \left(\frac{\partial^l p_r(x)}{\partial x^l}\right)^2 \mathrm{d}x$$

where k is the number of cells used in the reconstruction process. This indicator is a Sobolev-type seminorm based on the heuristic observation that highly oscillatory functions harbor high-magnitude derivatives. Thus, a higher value of β_r indicates that a smaller contribution from $p_r(x)$ should be considered. To be precise, let $\varepsilon > 0$ be some small (e.g. 10^{-6}) parameter which prevents machine overflow, and define

$$\tilde{\omega_r} = \frac{d_r}{(\varepsilon + \beta_r)^2}, \tag{43}$$

and

$$\omega_r = \frac{\tilde{\omega_r}}{\sum_{l=0}^{k-1} \tilde{\omega_l}}. \tag{44}$$

We weighted the constants d_r by the smoothness indicators: in smooth regions, the β_r are all nearly equal so that the stencils are all weighed approximately equally. In oscillatory regions, oscillatory reconstructions have very small weights. The ω_r are the convexly normalized to preserve consistency. The WENO procedure then uses

$$p(x_{i-1/2}) = \sum_{r=0}^{k-1} \omega_r p_r(x_{i-1/2}) \tag{45}$$

as the reconstructed value.

The WENO algorithm can be summarized as follows: we wish to determine $f(x_{i-1/2})$ using the point values of $F(x_{i-1/2})$:

1. Compute β_r for each reconstruction p_r.

2. Calculate the ω_r for each reconstruction using (43) and (44).

3. Use equation (45) to evaluate $f(x_{i-1/2}) = p(x_{i-1/2})$.

The WENO procedure reads: calculate three reconstructed values, one for each stencil S_0, S_1, and S_2; and combine them in a convex manner such that the more oscillatory reconstructions have a smaller contribution. This procedure has the weights tailored to admit high accuracy when the reconstruction is used at cell interfaces.

Suppose that we wish to reconstruct function values which are not on a cell interface. This is where the 'pointwise' WENO (PWENO) procedure makes its debut. The PWENO procedure is the same as the WENO procedure in spirit. However, some care must be taken since the desired reconstruction location x is not necessarily located at the interface of any element. (I.e. the solution to (34) is not necessarily any grid point.) To mold the WENO calculation into the PWENO formulation, all that must be done is to augment the weights $d_r \rightarrow d_r(x)$ and the smoothness indicators $\beta_r \rightarrow \beta_r(x)$ so that they are functions of the location of reconstruction. Of course, the weights d_r were derived from a high-order Taylor expansion, so solving for the d_r is in principle possible, it is not trivial computationally. The qualitative explanation of this is relatively straightforward, but the quantitative description and details of this method are quite complex, and we refer the reader to [44] and [10] for details.

The procedure for solving (32) is then clear: first we solve (34) for the value $X(t^n;\, t^{n+1},\, x_{i-1/2})$. Then we determine an ambient stencil around $X(t^n;\, t^{n+1},\, x_{i-1/2})$ and perform the PWENO process to reconstruct the value $F(t^n, X(t^n; t^{n+1}, x_{i-1/2}))$ given the pointwise primitive values defined by (40). The difference in this application is that we are interested in reconstructing the primitive and the original function f. However, the method for choosing the stencil remains the same.

Unfortunately, due to the relatively complicated nature of the PWENO reconstruction procedure, one cannot prove any of the qualities such as positivity or conservation for this type of flux-balance method. However, the oscillation-reducing properties of the WENO method are quite desirable for the Vlasov equation.

3.4 The Semi-Lagrangian Method

The semi-Lagrangian method intends to use the same motivation as is shown in figure 5: namely that the solution to the hyperbolic problem (32) can be determined by tracing the characteristics back in time using (34). The procedure is as follows: given $\{f(t^n, x_i)\}_{i\in I}$

- *Follow the characteristics* - Compute $\left\{X(t^n; t^{n+1}, x_i)\right\}_{i\in I}$ using equation (34)
- *Interpolate* - Use the known values $\{f(t^n,\, x_i)\}_{i\in I}$ to interpolate the values $\left\{f(t^n, X(t^n; t^{n+1}, x_i))\right\}_{i\in I}$, set $f(t^{n+1}, x_i) = f(t^n, X(t^n; t^{n+1}, x_i))$. See Figure 5.

The first step is relatively easy to accomplish without much of a problem, especially since the velocity field in equation (34) is constant for the x-advection. However, the second step of interpolation is quite a challenge. Indeed, this step is what differentiates each semi-Lagrangian method from the next. We present a few algorithms below and refer the reader to [47] and [17] for a more in-depth survey of these methods.

3.4.1 Lagrange Interpolations

The first semi-Lagrangian method was introduced by Cheng and Knorr in [11]. Their attempts include linear, quadratic, and cubic interpolation. For a general $(2m+1)$th-order polynomial approximation of $f(t^n, \cdot\,)$ at x, we first determine i such that $x \in [x_i,\, x_{i+1}]$. Then we fit a Lagrangian polynomial of order $2m+1$ with the nodal abscissae at $\{x_{i-m}, \ldots, x_{i-1}, x_i, \ldots, x_{i+m-1}\}$. This is given by the formula

$$f_h(x) = f(t^n, x_{i-m}) + \sum_{k=1}^{2m+1} f[x_{i-m}, \ldots, x_{i-m+k}] \prod_{l=0}^{k} (x - x_{i-m+1}),$$

where

$$f[x_i, \ldots, x_{i+p}] = \frac{1}{p!} \frac{f[x_{i+1}, \ldots, x_{i+p}] - f[x_i, \ldots x_{i+p-1}]}{x_{i+p} - x_i}$$

are the recursively-defined Newton divided differences, with $f[x_i] = f(t^n, x_i)$. This method relies on a centered stencil and is thus limited to odd-degree interpolation, but an even-degree interpolant may easily be implemented as well.

The authors in [11] observe that linear interpolation is too dissipative and maintain that cubic interpolation is satisfactory. Numerical simulations in [17] indicate that increasing interpolation orders (test were performed up to a 9th degree interpolation) decreases the amount of dissipation and also decreases phase error. However, increasing the degree of the interpolant requires more computational effort, and widens the reconstructing stencil. It should be noted that this is a high-order reconstruction method, but it does not inherently preserve positivity or mass.

3.4.2 Hermite Interpolations

The previous section introduced an interpolant which was only a continuous approximation to the density f. The Hermite interpolation method introduced in [17] proposes to use a C^1-accurate reconstruction. Consider for simplicity of exposition a cubic reconstruction. The reconstruction $f_h(x)$ for $x \in [x_i, x_{i+1}]$ is sought so that

$$\begin{aligned} f_h(x_i) &= f(t^n, x_i) & f_h'(x_i) &= \frac{\partial f}{\partial x}(t^n, x_i) \\ f_h(x_{i+1}) &= f(t^n, x_{i+1}) & f_h'(x_{i+1}) &= \frac{\partial f}{\partial x}(t^n, x_{i+1}) \end{aligned}$$

That is, we ask that the reconstruction match the values and the derivatives at the points x_i and x_{i+1}. Of course, the values $f(t^n, x_i)$ and $f(t^n, x_{i+1})$ are known, so these two constraints pose little problem. However, it is not straightforward at all how to calculate the derivative $\frac{\partial f}{\partial x}$ at the grid points. A method is introduced in [36] by which a second transport equation is introduced for which the unknowns are the derivatives $\frac{\partial f}{\partial x}$ evaluated at the grid points. This new transport equation reads

$$\frac{\partial(\partial_x f)}{\partial t} + \frac{\partial[v(\partial_x f)]}{\partial x} = 0$$

where we have assumed that the velocity field v is divergence-free. The disadvantage of this method is that one must solve another transport equation in tandem with the original, which increases computational cost and memory requirements, especially if we must do this in (6+1)-dimensional space.

An alternative is proposed by Filbet et al. in [17], where the authors approximate the derivative as

$$\frac{\partial f}{\partial x}(x_i) = \frac{1}{12\,\Delta x}\,[8(f(x_{i+1}) - f(x_{i-1})) - (f(x_{i+2}) - f(x_{i-2}))], \tag{46}$$

which is a well-known fourth-order accurate finite-difference formula. Use of this approximation renders the algorithm relatively easy to implement and makes it cost-effective. Numerical results show that this method performs on par with a 3rd or 5th order Lagrange interpolant.

Of course one may imagine a generalization of this Hermite interpolation method. Indeed, consider some integer $k \geq 0$ and some stencil size $s \geq 1$. Then given that $x \in [x_i, x_{i+s}]$, we could attempt to find a reconstruction $f_h(x)$ which satisfies

$$f_h^{(q)}(x_r) = f^{(q)}(x_r) \qquad \forall\, q = 0, 1, \ldots k \text{ and } r = 0, 1, \ldots s. \tag{47}$$

On the other hand, formulating this scheme is not too difficult to do, but implementing it is another thing. How do we compute $f^{(q)}(x_r)$? Finite-difference formulas of the form (46) are not readily available for any order derivative with any stencil. Additionally, these difference formulas may produce spurious oscillations when used. Indeed, implementing schemes of the form (47) seem at this time academic in interest only.

3.4.3 PWENO Reconstruction

We ask the reader to recall the PWENO reconstruction treated in section 3.3.3. This reconstruction forms the basis for this method. Given $x = X(t^n; t^{n+1}, x_i)$, suppose $x \in [x_j, x_{j+1}]$ for some j. We form the reconstruction $f_h(x) \approx f(t^n, X(t^n; t^{n+1}, x_i))$ from the point values $\{f(t^n, x_i\}_{i \in I}$ with PWENO. This idea can be attributed to Carrillo and Vecil in [10]. Their numerical simulations show that using PWENO as a reconstruction method instead of the usual Lagrange interpolation for a one-dimensional advection problem produces much better results in the discrete L^1 norm, and also preserves the discrete total variation when the Lagrange interpolation method does not. More complicated two-dimensional phase space tests also indicate that the method performs very well when compared to other semi-Lagrangian or flux balance methods.

3.5 Other Methods

In the previous sections we have summarized some of the methods which have been developed to solve the Vlasov equation (29). So far these methods have been lumped into the general category of FBM solvers and semi-Lagrangian solvers. However, there are other methods that have been presented in the literature to solve this equation while maintaining positivity and conservation. Gamba and Proft [20] have applied a discontinous Galerkin (DG) method to solve the problem, Arakawa [1] introduced a finite-difference method in 1966 for incompressible fluid flow which was later adapted by Filbet and Sonnendrücker in [17] for the Boltzmann equation, and Klimas and Farrell have explored spectral method representations in [33] and [32]. In this section, we shall briefly describe these three methods and their properties.

3.5.1 An Adapted Finite-Difference Scheme

In 1966, Arakawa [1] introduced a finite-difference method to solve the incompressible Euler equation of fluid flow in two dimensions. He showed that his scheme conserved the 'mean Jacobian', squared vorticity, kinetic energy. Later in 2003, Filbet [17] adapted Arakawa's scheme for Vlasov-Poisson models and noted that the conserved quantities for the Euler equations which Arakawa derived corresponded to the quantities mass, energy, and mean-square of f, respectively. We follow Filbet's adaptation here: consider the $1+1$ phase-space Vlasov-Poisson model:

$$\begin{aligned} &\frac{\partial f}{\partial t} + v\,\frac{\partial f}{\partial x} + \frac{\partial \phi}{\partial x}\,\frac{\partial f}{\partial v} = 0 \\ &\nabla_x^2 \phi = \int f \,\mathrm{d}x\,\mathrm{d}v - 1 \end{aligned} \tag{48}$$

We define the auxilliary variables

$$\psi = \phi - \frac{v^2}{2} \qquad\qquad J(\psi, f) = \frac{\partial \psi}{\partial x}\,\frac{\partial f}{\partial v} - \frac{\partial \psi}{\partial v}\,\frac{\partial f}{\partial x}, \tag{49}$$

which we can use to rewrite (48) as

$$\frac{\partial f}{\partial t} + J(\psi, f) = 0.$$

Having made this transformation, Filbet notices that this is exactly the equation which Arakawa solved for fluid problems. One can show that Arakawa's conserved quantities can be translated into Boltzmann-type conserved quantities as:

$$\begin{aligned} &\int J_h(\psi, f)\,\mathrm{d}x\,\mathrm{d}v = 0 \longrightarrow \int f_h(t)\,\mathrm{d}x\,\mathrm{d}v = \int f_h(0)\,\mathrm{d}x\,\mathrm{d}v \\ &\int \psi_h\, J_h(\psi, f)\ \mathrm{d}x\,\mathrm{d}v = 0 \longrightarrow \int \psi_h(t)\ f_h(t)\,\mathrm{d}x\,\mathrm{d}v = \int \psi_h\,(0) f_h(0)\,\mathrm{d}x\,\mathrm{d}v \\ &\int f_h\, J_h(\psi, f)\ \mathrm{d}x\,\mathrm{d}v = 0 \longrightarrow \int f_h^2(t)\,\mathrm{d}x\,\mathrm{d}v = \int f_h^2(0)\,\mathrm{d}x\,\mathrm{d}v, \end{aligned} \tag{50}$$

where the integrals over the discrete representations refer to sums of the nodal values. These three conserved quantities correspond to mass, energy, and the L^2 norm of f. We recall some standard notation for spatial differences finite-difference methods:

$$D_i^0(f_h)_{i,j} = (f_h)_{i+1,j} - (f_h)_{i-1,j} \qquad D_i^+(f_h)_{i,j} = (f_h)_{i+1,j} - (f_h)_{i,j} \qquad D_i^-(f_h)_{i,j} = (f_h)_{1,j} - (f_h)_{i-1,j}$$

$$D_{i,j}^{+,-}(f_h)_{i,j} = (f_h)_{i+1,j} - (f_h)_{i,j-1} \quad D_{i,j}^{-,+}(f_h)_{i,j} = (f_h)_{i,j+1} - (f_h)_{i-1,j}$$

In order to define the spatial differences which we shall use to approximate J_h, Arakawa first defines some intermediate Jacobians:

$$\begin{aligned}
\mathbb{J}_{i,j}^{++}(f_h,\psi_h) &= \frac{1}{4\,h^2}\big[D_i^0(f_h)_{i,j}\,D_j^0(\psi_h)_{i,j} - D_j^0(f_h)_{i,j}\,D_i^0(\psi_h)_{i,j}\big]\\
\mathbb{J}_{i,j}^{+\times}(f_h,\psi_h) &= \frac{1}{4\,h^2}\big[D_i^0((f_h)_{i,j}\,D_j^0(\psi_h)_{i,j}) - D_j^0((f_h)_{i,j}\,D_i^0(\psi_h)_{i,j})\big]\\
\mathbb{J}_{i,j}^{\times+}(f_h,\psi_h) &= \frac{1}{4\,h^2}\Big[(f_h)_{i+1,j+1}(D_j^+ - D_i^+)(\psi_h)_{i,j} - (f_h)_{i-1,j-1}(D_j^+ - D_i^+)(\psi_h)_{i-1,j-1} -\\
&\qquad (f_h)_{i-1,j+1}D_{i,j}^{-;-}(\psi_h)_{i,j+1} + (f_h)_{i+1,j-1}D_{i,j}^{-;-}(\psi_h)_{i+1,j}\Big]\\
\mathbb{J}_{i,j}^{\times+}(f_h,\psi_h) &= \frac{1}{8\,h^2}\Big[D_{i,j}^{0,0}(f_h)_{i,j}\;(D_j^0(\psi_h)_{i-1,j} - D_i^0(\psi_h)_{i,j-1}) -\\
&\qquad (D_j^0(f_h)_{i-1,j} - D_i^0(f_h)_{i,j-1})\,D_{i,j}^{0,0}(\psi_h)_{i,j}\;\Big]
\end{aligned}$$

Using these finite-difference approximations, he forms the following approximations of the term $J(f,\psi)$:

$$\mathbb{J}_{i,j} = \alpha\,\mathbb{J}_{i,j}^{++} + \beta\,\mathbb{J}_{i,j}^{+\times} + \gamma\,\mathbb{J}_{i,j}^{\times+} + \delta\,\mathbb{J}_{i,j}^{\times\times}. \tag{51}$$

Finally, he proves the following properties:

Proposition 1. *(Finite difference accuarcy, conservation [1])*

Let (51) be the finite difference approximation to $J(f,\psi)$ given in (49). The following hold:

1. *$\alpha = \beta = \gamma = \frac{1}{3}$ and $\delta = 0$ yields a second-order accurate scheme which conserves the properties in (50).*
2. *$\beta = \gamma = \delta = \frac{1}{3}$ and $\alpha = 0$ yields a second-order accurate scheme which conserves the properties in (50).*
3. *$\alpha = \frac{2}{3}$, $\beta = \gamma = \frac{1}{3}$, and $\delta = -\frac{1}{3}$ yields a fourth-order accurate scheme which conserves the properties in (50).*

Although it is clear that this method has desirable properties for the Vlasov equation, it is unclear how to generalize this method to higher-dimensional phase space. Indeed, Arakawa's original method was developed explicitly for the two-dimensional system, and so generalizations to six dimensions is not *a priori* straightforward.

3.5.2 A Spectral Method

A number of studies have been published regarding spectral methods discretizations for the Vlasov equation. We shall use some of the work by Klimas [33] and [32] to give a brief exposition of the method. We start with the $1+1$ phase-space Vlasov-Poisson model

$$\begin{aligned}
&\frac{\partial f}{\partial t} + v\,\frac{\partial f}{\partial x} + \frac{\partial \phi}{\partial x}\,\frac{\partial f}{\partial v} = 0\\
&\nabla_x^2\phi = \int f\,\mathrm{d}x\,\mathrm{d}v - 1
\end{aligned} \tag{52}$$

Klimas uses a Strang-splitting method to advect the Vlasov equation: we assume that there is periodicity in the x-direction and that f has zero support outside $|v| \leq V_{\max}$ for some $V_{\max} \geq 0$ (see Section 4). The x-advection satisfies

$$\frac{\partial f}{\partial t} + v\,\frac{\partial f}{\partial x} = 0, \tag{53}$$

and assuming a Fourier Transform in x

$$f(x, v, t) = \sum_{n=-N}^{N} \hat{f}_n(v, t)\, e^{-inx},$$

and similarly in v:

$$f(x, v, t) = \sum_{n=-N}^{N} \tilde{f}_n(x, t)\, e^{-inv},$$

then we may write the exact advection step for a time-step Δt as

$$f(x, v, t + \Delta t) = \sum_{n=-N}^{N} \hat{f}\,(v, t)\, e^{iv\Delta t}\, e^{-inx}. \tag{54}$$

Thus, we can evolve the split stages of (52) by simply phase-shifting the modes $\hat{f}\,(v,\ t)$. Similarly, we assume that we have computed the appropriate $E = \frac{\partial \phi}{\partial x}$ satisfying (52), so that a v-advection step takes the form

$$f(x, v, t + \Delta t) \approx \sum_{n=-N}^{N} \tilde{f}\,(x, t)\, e^{iE\Delta t}\, e^{-inv}. \tag{55}$$

Thus, up to the approximation in (55) and the the Strang splitting error, we can solve the system (52) using successive steps of (54) and (55) and the FFT. This is an extremely attractive concept: a spectral treatment of the unknown and accurate temporal advancement. However, there are many problems with this method: firstly, we have assumed that f is periodic in x and has compact support in v. For many problems (e.g. an equilibrium Maxwellian density) this is not satisfied. Additionally, the support of f in v increases as time is advanced. Thus, eventually we shall develop a numerical solution f which has very oscillatory behavior; i.e. the high modes are flooded. This is very easily shown: equation (53) has the solution $f(x, v, t) = f_0(x - v\,t, v)$ where $f_0(x, v)$ is the initial condition. Then we can calculate

$$\frac{\partial f}{\partial v} = \frac{\partial f_0}{\partial v} - t\,\frac{\partial f_0}{\partial x},$$

and the second term is clearly unbounded in time, i.e. the velocity dependence in f has steep gradients, which translates into energy in the high modes of the v-space Fourier Transform. This is the so-called *filamentation* problem. Filamentation causes corruption of the numerical solution due to the fundamental nature of the spectral method. A simple solution would be to simply filter the resulting solution in velocity space. The problem with filtering is that it produces unpredictable damage to the accuracy of the solution.

Klimas [33] showed that a particular type of Gaussian filtering in v-space produces a solution which, instead of solving the system (52), solves a new system. Let us introduce the filter that we shall use over v-frequency space, and we denote v-frequency as ω_v:

$$G(\omega) = v_0\,\sqrt{\frac{2}{\pi}}\, e^{-2(\omega v_0)^2}$$

where $v_0 > 0$ is a parameter. Note that the limit of $v_0 \rightarrow 0$, then there is no filtering. Suppose that we consider the exact solution f to (52) and we apply $G(v)$ to its v-Fourier Transform, and then transform back to v-space to obtain a solution $\bar{f}$. I.e, $\bar{f}$ is defined by

$$\bar{f}\,(x,v,t) = \frac{1}{2\pi\,(2N+1)} \sum_{n=-N}^{N} \left(\sum_{n=-N}^{N} G\!\left(\frac{2\,\pi n}{V_{\max}}\right) \tilde{f}_n(x,t)\, e^{-inv} \right) e^{inv}.$$

Klimas then shows that this new function (or, rather, its continuous Fourier Transform counterpart) satisfies the evolution system

$$\begin{aligned} \frac{\partial \bar{f}}{\partial t} + v\,\frac{\partial \bar{f}}{\partial x} + \frac{\partial \phi}{\partial x}\,\frac{\partial \bar{f}}{\partial v} &= -\,v_0^2 \frac{\partial^2 \bar{f}}{\partial x\,\partial v} \\ \nabla_x^2 \phi &= \int \bar{f}\ \mathrm{d}x\,\mathrm{d}v - 1 \qquad (56) \\ \frac{\partial}{\partial t}\!\left(\frac{\partial \phi}{\partial x}\right) &= -\int v\,\bar{f}\ \mathrm{d}x\,\mathrm{d}v. \end{aligned}$$

This by itself is not too useful, but the real advantage comes in with a filamentation analysis: Klimas also showed that the parameter v_0 can be chosen dependent on the initial condition so that filamentation does not develop, i.e. so that one can obtain a provable bound on the velocity derivative of the function $\bar{f}$. In addition, $\bar{f}$ conserves the same mass and momentum as f:

$$\int f\,\mathrm{d}x\,\mathrm{d}v = \int \bar{f}\ \mathrm{d}x\,\mathrm{d}v$$

$$\int f\,v\,\mathrm{d}x\,\mathrm{d}v = \int \bar{f}\,v\,\mathrm{d}x\,\mathrm{d}v$$

The advantages of using the solution to the new system (56), or rather, of doing a specific kind of filtering to the original system (52), are provable prevention of velocity-space filamentation while preserving certain moments of the distribution function.

3.5.3 A Discontinuous Galerkin Method

Discontinuous Galerkin (DG) methods have recently been used in a variety of applications in numerical analysis [27]. DG methods have the advantage of combining the spectral-type convergence of modal representations with the finite-volume robustness for handling shock waves in hyperbolic systems. Very recently, Gamba and Proft have proposed a DG scheme specifically designed for linear Vlasov-Boltzmann models [20]. Their scheme is as follows:

Assume we have a space $\Omega \subset \mathbb{R}^3 \times \mathbb{R}^3$ representing a subspace of (x,v) phase space with finite volume. In particular, we assume Ω has rectangular boundaries such that $x_i \in [0, L_i]$ and $v_i \in [-V_i, V_i]$ for $i = 1, 2, 3$. We assume a regular finite element triangulation $\mathcal{T}_h = \{\omega\}$ of the space Ω. $\mathcal{T}_h$ has boundary faces $\Gamma \circeq \cup_{\omega \in \mathcal{T}_h} \partial\omega$, and we split Γ up into seven regions defined by

Γ_i The set of interior faces: $\Gamma \backslash \partial\Omega$
Γ_{0^-} The set of faces $\gamma \in \Gamma$ representing $x_i = 0, v_i < 0$
Γ_{0^+} The set of faces $\gamma \in \Gamma$ representing $x_i = 0, v_i > 0$
Γ_{L^-} The set of faces $\gamma \in \Gamma$ representing $x_i = L_i, v < 0$
Γ_{L^+} The set of faces $\gamma \in \Gamma$ representing $x_i = L_i, v > 0$
Γ_{V^-} The set of faces $\gamma \in \Gamma$ representing $v_i = -V_i$
Γ_{V^+} The set of faces $\gamma \in \Gamma$ representing $v_i = V_i$

We define the space of discontinuous functions for some $k \geq 0$:

$$\Phi_h := \left\{ \phi \in L^2(\Omega) \colon \phi|_{(x,v)\in\omega} \in P^k(\omega) \, \forall \, \omega \in \mathcal{T}_h \right\},$$

where $P^k(\omega)$ represents the space of k'th order polynomials on the element ω. We associate with each face of each element an outward-pointing normal vector $\hat{n}_\omega$. At the interface of an element ω, the functions $\phi \in \Phi_h$ are discontinuous, so we specify the local value of the function as ϕ^- and the external value of the function as ϕ^+. Similarly, $\hat{n}_\omega^-$ is the outward-pointing normal vector and $\hat{n}_\omega^+$ is the inward-pointing normal vector. Then for any element ω at any interface $\gamma \in \Gamma$ in the triangulation, and for any $\phi \in \Phi_h$, we define the average and jump operators

$$\{\phi\} = \frac{1}{2}(\phi^- + \phi^+) \qquad [\![\phi]\!] = \hat{n}_\omega^- \cdot \phi^- + \hat{n}_\omega^+ \cdot \phi^+$$

Finally, we define a numerical flux $\hat{\phi}$ upwind as

$$\hat{\phi} = \{\phi\} + \frac{1}{2}[\![\phi]\!] \tag{57}$$

We seek a solution $f_h \in \Phi_h$ and rewrite the Vlasov equation (29) as

$$\frac{\partial f}{\partial t} + \nabla \cdot (A\, f) = 0, \tag{58}$$

where

$$A = \begin{pmatrix} v \\ L \end{pmatrix},$$

and the ∇ operator acts on the full (x, v) phase space. Then we can form the usual DG scheme by multiplying (58) by a test function $\phi \in \Phi_h$ and enforcing L^2 orthogonality of the residual with an integration by parts:

$$\sum_{\omega\in\Omega} \int_\omega \frac{\partial f_h}{\partial t} \phi \, \mathrm{d}x \, \mathrm{d}v - \sum_{\omega\in\Omega} \int_\omega A\, f_h \cdot \nabla\phi \, \mathrm{d}x \, \mathrm{d}v + \sum_{\omega\in\Omega} \int_{\gamma=\partial\omega} \phi \, (A\, f_h)^* \cdot \hat{n}_\omega^- \, \mathrm{d}\gamma = 0, \tag{59}$$

where we have yet to define the numerical flux $(A\, f_h)^*$. Indeed, the choice of numerical flux is at the heart of each DG method. Recalling the upwind flux definition in (57), Gamba uses the following form for the fluxes:

$$(A\, f_h)^* = \begin{cases} \widehat{A\, f_h} & \text{on } \Gamma_i \\ -A\, f_h & \text{on } \Gamma_{V^\pm} \\ A\, f_h & \text{on } \Gamma_{0^-} \text{ and } \Gamma_{L^+} \\ A\, f_0 & \text{on } \Gamma_{0^+} \\ A\, f_L & \text{on } \Gamma_{L^-} \end{cases} \tag{60}$$

where we have introduced f_0 and f_L as the boundary conditions on appropriate x-inflow boundaries. These choices of fluxes corresponds to upwinding in the interior of the domain, x-inflow and x-outflow at the appropriate x-boundaries, and homogeneous Dirichlet conditions at v-boundaries.

Gamba and Proft also prove the following properties of the scheme (59)-(60):

Proposition 2. *(DG Stability, Convergence, Conservation, Positivity [20])*

The scheme defined by (59)-(60) for the system (58) has the following properties:

1. L^1 *stability:* $\|f_h(t=T)\|_{L^1(\Omega)} \leq \|f_h(t=0)\|_{L^1(\Omega)}$

2. *Convergence:* $\|f - f_h\|_{L^1(\Omega)} \leq t\ h^k(C_0 + C_1)$*, where* C_0 *depends on the mesh and higher-order Sobolev norms of* f*, and* C_1 *depends on the mesh and higher-order Sobolev norms of* $f(t=0)$*.*

3. *Conservation:*

$$\frac{\partial}{\partial t}\int_{\Omega} f_h \,\mathrm{d}x\,\mathrm{d}v = 0$$

4. *Positivity: if k=0 (piecewise-constant approximation), then if* $f(t=0)\geq 0$*, then the numerical solution to the semi-discrete form of (59) satisfies* $f_h \geq 0$ *for* $t \geq 0$*.*

Gamba and Proft have implemented the scheme on 1 + 1-dimensional phase space with encouraging results.

4 Collision Operator

We ask the reader to recall the forms of the collision operator in equations (5), (6), and (7). Since we will be timestepping the collision operator and the advection term separately, we reduce (4) to time evolution using *only* the collision term, ignoring transport:

$$\frac{\partial f}{\partial t} = \frac{1}{k_n} Q(f, f). \tag{61}$$

Our first observation here is that the collision operator is local in x. Without the transport term, $f(x)$ and $f(x^*)$ for $x \neq x^*$ evolve completely independently of each other. Therefore, we may limit our considerations in this section to one particular point in space. Further, this allows easy (spatial-subdivision-based) parallelization of algorithms for treating the Boltzmann equation.

We will begin this section by reviewing a little bit of the theory surrounding the Boltzmann collision operator, and we will encounter several (often drastic) changes that were made to the rather onerous collision integral with the goal of simplifying it. Next, we will discuss the discretization of velocity space and, armed with that knowledge, we will review a few representative methods in more detail. We will focus on deterministic methods, noting that stochastic methods for the collision operator [37, 30] are a fast, but noisy alternative for problems where high accuracy is not of the utmost importance. Finally, we will discuss a time stepping method that is particularly well-suited to the requirements of the Boltzmann collision operator.

4.1 Theory and Possible Simplificiations

Let d be the dimension of velocity space. The Boltzmann collision operator conserves a few of its *moments*, namely mass, momentum and energy:

$$\int_{\mathbb{R}^d} Q(f, f)\begin{cases} 1 \\ v \\ |v|^2 \end{cases} \mathrm{d}v = 0.$$

The solution f of (61) evolves towards a steady state called a *Maxwellian*:

$$f_\infty(v) := \frac{\rho}{(2\pi T)^{d/2}} \exp\left(-\frac{|V - v|^2}{2T}\right).$$

The Maxwellian depends on a few moments of the initial distribution:

Density ρ.

$$\rho := \int_{\mathbb{R}^d} f(v)\mathrm{d}v$$

Mean/Bulk Velocity.

$$V := \frac{1}{\rho}\int_{\mathbb{R}^d} v\, f(v)\mathrm{d}v$$

Temperature.

$$T := \frac{1}{3\rho}\int_{\mathbb{R}^d} |V-v|^2 f(v)\mathrm{d}v$$

This evolution towards the Maxwellian is true even for the whole Boltzmann equation, with the same Maxwellian f_∞, but with slightly modified moments. This Maxwellian is then (sensibly) uniform in x.

Furthermore, the following technical theorem tells us about how (61) evolves toward equilibrium and is sometimes useful as a diagnostic for our numerical methods:

Theorem 3. (Boltzmann's H-Theorem) *The collision operator Q satisfies the inequality*

$$\int_{\mathbb{R}^d} Q(f,f)\log(f)\mathrm{d}v \leqslant 0.$$

The term

$$H(f) := \int_{\mathbb{R}^d} Q(f,f)\log(f)\mathrm{d}v$$

can be seen as negative entropy, and Theorem 3 implies that entropy can only increase.

4.1.1 Linearizing the Collision Operator

Recall that our collision operator from above is written as

$$Q(f,f) = \int_{\mathbb{R}^3}\int_{S^2} B(|v-v_*|,\cos\theta)\big[f(v')\,f(v'_*) - f(v)f(v_*)\big]\mathrm{d}\omega\,\mathrm{d}v_*. \tag{62}$$

We know that the density function f will eventually evolve toward a Maxwellian equilibrium f_∞. The first idea we might have about simplifying (62) is to linearize it about this equilibrium, taking

$$f := f_\infty + g,$$

and computing

$$\begin{aligned}
Q(f,f) &= Q(f_\infty+g, f_\infty+g)\\
&= \underbrace{Q(f_\infty,f_\infty)}_{0} + Q(g,f_\infty) + Q(g,f_\infty) + \underbrace{Q(g,g)}_{\approx 0}\\
&= Q(g,f_\infty) + Q(g,f_\infty)\\
&= \int_{\mathbb{R}^3}\int_{S^2} B(|v-v_*|,\cos\theta)\big[g(v')\,f_\infty(v'_*) - g(v)f_\infty(v_*)\big]\mathrm{d}\omega\,\mathrm{d}v_*\\
&\quad + \int_{\mathbb{R}^3}\int_{S^2} B(|v-v_*|,\cos\theta)\big[f_\infty(v')\,g(v'_*) - f_\infty(v)g(v_*)\big]\mathrm{d}\omega\,\mathrm{d}v_*\\
&= \int_{\mathbb{R}^3}\int_{S^2} B(|v-v_*|,\cos\theta)\big[g(v')\,f_\infty(v'_*) + f_\infty(v')\,g(v'_*) - g(v)f_\infty(v_*) - f_\infty(v)g(v_*)\big]\mathrm{d}\omega\,\mathrm{d}v_*.
\end{aligned}$$

While the above is what many articles refer to as the *linearized Boltzmann collision operator*, some authors also work with the expression

$$Q^{\text{lin}}(f,f)=\int_{\mathbb{R}^3} k(v,v')f_\infty(v)f(v')-k(v',v)f_\infty(v')f(v)\mathrm{d}v'.$$

Equation (4) with this collision operator is termed the *linear Vlasov-Boltzmann equation.*

4.1.2 The Bhatnagar-Gross-Krook Approximation

Even simpler than the above linear representation, the *Bhatnagar-Gross-Krook ("BGK")Approximation* [49] rests on the fact that the effect of the collision term in (61) is mainly to push the distribution function towards an equilibrium state. Therefore, the idea here is to simply replace $Q^{\text{lin}}(f)$ by the difference between f and its equilibrium value, divided by a relaxation time controlling the speed of the evolution:

$$Q^{\text{BGK}}(f)=\frac{1}{\tau}(f_\infty-f).$$

It is surprising that even the rather crude BGK approximation of the full Boltzmann equation is still powerful enough to derive the full set of "classical" hydrodynamic equations from it, including the continuity equation, and the Navier-Stokes equation [49].

This implies that computations of the BGK-Boltzmann equation capture *at least* as much detail as classical CFD calculations. A class of CFD methods called *Lattice Boltzmann Methods* makes use of exactly this fact.

4.2 Discretizing Velocity Space

The first step in any numerical calculation is the choice of an appropriate discretization. While the consequences of choosing a spatial discretization are most directly felt when treating the transport term, the choice of discretization of velocity space has its major impact on the computation of the collision term. While many methods make do with only one velocity space representation, some actually use two different representations and constantly transform back and forth between them.

The largest problem in dealing with velocity space is clearly the fact that it, in contrast to our computational resources, is infinite, and there is no 'preferred' or 'obvious' method of squeezing it into a finite representation. Some attempts [45] have been made to expand f into functions of unbounded support (often Hermite polynomials), however none of these seem to have managed to move past the BGK-approximated collision term.

We will thus first try to find some justification for only allowing f to have finite support.

Proposition 4. *(Compact support of the density function, Lemma 1 in [16], proved as Proposition 2.1 in [39])*

Suppose $\operatorname{supp}_v(f)\subset\mathcal{B}(0,R)$*. Then*

1. $\operatorname{supp}_v(Q(f,f)(v))\subset\mathcal{B}(0,R\sqrt{2})$.

2. *We define the change of variables* $g:=v-v_*$*. If we suppose* $v\in\mathcal{B}(0,R)$*, then we do not change the value of* Q *if we restrict* $g\in\mathcal{B}(0,2R)$*, i.e.*

$$Q(f,f)(v)=\int_{\mathcal{B}(0,2R)}\int_{S^2} B(|g|,\cos\theta)\left[f(v')\,f(v'_*)-f(v)\,f(v-g)\right]\mathrm{d}\omega\,\mathrm{d}g.$$

3. *With this limitation, we automatically also have* $v_*=v-g, v', v'_*\in\mathcal{B}(0,(2+\sqrt{2})R)$*.*

$\mathcal{B}(C, R)$ is the ball of radius R centered at C. $\operatorname{supp}_v(f)$ denotes the set $\{v: f(v) \neq 0\}$.

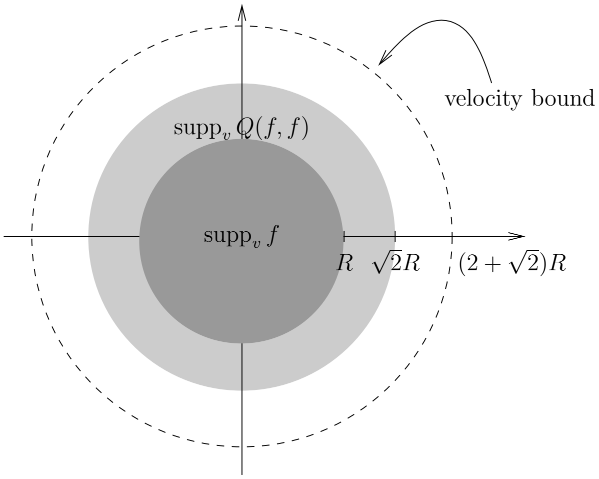

Figure 7. The Situation of Proposition 4.

Observe that this result does not encourage us to truncate the support of f: Even if at timestep n, the support of f falls within a ball of radius R, the support might have grown by a factor of $\sqrt{2}$ in the very next timestep, and grown larger in the step after that, etc.

4.2.1 Fourier Transforms in Velocity Space

Quite a few of the following methods use Cartesian Fourier transforms of the density function f in velocity space. Since this is such a common theme, we discuss the implications of doing this beforehand. The aim in all cases is to use the Fast Fourier Transform (FFT) to speed up the computation of the expensive collision integral similarly to how convolution integrals are reduced from quadratic complexity to $O(N \log N)$ by the FFT.

We begin by discretizing velocity space $\mathbb{R}^3$ into the lattice $h\mathbb{Z}^3$:

$$v_k := h_v k, \quad h_v > 0, \quad k \in \mathbb{Z}^3. \tag{63}$$

Then, we approximate the integral

$$\varphi(\xi) = \int_{\mathbb{R}^3} f(v) e^{i(v \cdot \xi)} \mathrm{d}v$$

with the midpoint-rule like sum

$$\tilde{\varphi}(\xi) = h_v^3 \sum_{k \in \mathbb{Z}^3} f(v_k) e^{i(v \cdot \xi)}. \tag{64}$$

This immediately leads to the question at which points ξ we would need to evaluate $\tilde{\varphi}(\xi)$. We resolve this by also picking a lattice in Fourier-velocity space:

$$\xi_j := h_\xi j, \quad h_\xi > 0, \quad j \in \mathbb{Z}^3. \tag{65}$$

In order to be able to carry out the transform by the FFT, we require

$$h_v h_\xi = \frac{2\pi}{n}, \quad n \in \mathbb{N}.$$

We see

$$\sum_{k \in \mathbb{Z}^3} f(v_k) e^{i\frac{2\pi}{n}((k+e_i n)\cdot j)} = \sum_{k \in \mathbb{Z}^3} f(v_k) e^{i\frac{2\pi}{n}(k\cdot j)} = \sum_{k \in \mathbb{Z}^3} f(v_k) e^{i\frac{2\pi}{n}(k\cdot(j+e_i n))},$$

where e_i is the ith unit vector. We can therefore limit our grids (63) and (65) to n^3 elements each, and thereby automatically make our velocity space $2L$-periodic, with

$$L := \frac{n\,h_v}{2}.$$

The goal must be to pick L as small as possible to improve resolution within $[-L, L]^3$, while picking it large enough to avoid spurious influence from periodic copies of the main mode of f somewhere else in velocity space, an effect commonly known as *aliasing*.

Suppose $\operatorname{supp}_v(f) \subset \mathcal{B}(0, R_0)$. Taking into account Proposition 4, we know $\operatorname{supp}_v(Q(f, f)) \subset B(0, \sqrt{2}R_0)$, and that the relative velocity (g above) is bounded in $B(0, 2R_0)$. Since we would really like our solutions within $\operatorname{supp}_v Q(f, f)$ to be correct, we need to make sure that these velocities keep a distance of $2R_0$ from the closest spurious non-zero location. Therefore, we obtain

$$L \geqslant \frac{1}{2}(\sqrt{2}R_0 + 2R_0 + R_0) = \frac{3 + \sqrt{2}}{2} R_0$$

as a guideline for the minimum period. Figure 8 aims to clarify this formula.

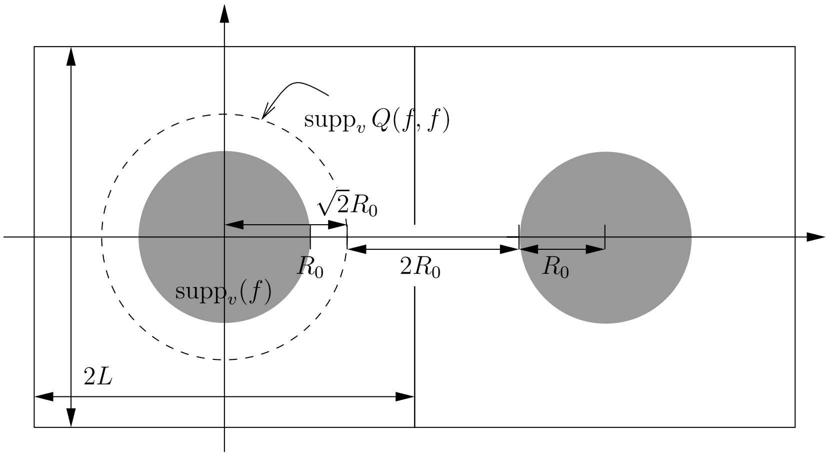

Figure 8. Considerations for mitigating the artificial L-periodicity in velocity space.

As a final note, some methods below choose to change the definition (63) to originate at the (conserved) bulk velocity. This makes particular sense for situations where one is truly only interested in a solution of a space-homogeneous Boltzmann equation. (I.e. the velocity situation is assumed to be the same over all real space.) In this case, (63) is changed to

$$v_k := V + h_v k, \quad h_v > 0, \quad k \in \mathbb{Z}^3,$$

and (64) becomes

$$\tilde{\varphi}(\xi) = h_v^3 e^{i(V \cdot \xi)} \sum_{k \in \mathbb{Z}^3} f(v_k) e^{i(v \cdot \xi)},$$

with the bulk velocity V defined as follows:

$$\begin{array}{rcl} V & := & \dfrac{1}{\rho} \displaystyle\int_{\mathbb{R}_v^3} v\, f(v, t) \mathrm{d}v = \dfrac{1}{\rho} \displaystyle\int_{\mathbb{R}_v^3} v\, f(v, 0) \mathrm{d}v, \\ \rho & := & \displaystyle\int_{\mathbb{R}_v^3} f(v, t) = \int_{\mathbb{R}_v^3} f(v, 0). \end{array}$$

4.3 A Fourier Method by Ibragimov and Rjasanow

The article [29] by I. Ibragimov and S. Rjasanow introduces a way of slicing and dicing the collision gain term (6) that is firmly rooted in the Fourier transform. To match the article and avoid confusion, we introduce the authors' notation for the Fourier transform:

$$\begin{aligned}\mathcal{F}_{v\to\xi}[g(v)](\xi) &:= \int_{\mathbb{R}^3} g(v)e^{i(v\cdot\xi)}\mathrm{d}v,\\ \mathcal{F}^{-1}_{\xi\to v}[\hat{g}(\xi)](v) &:= \frac{1}{(2\pi)^3}\int_{\mathbb{R}^3}\hat{g}(\xi)e^{-i(v\cdot\xi)}\mathrm{d}\xi.\end{aligned}$$

Since the gain term is the most expensive part of approximating the collision operator (62), this method focuses on its treatment, beginning with the standard (substituted) form:

$$\begin{aligned}Q_+(f,f)(v) &= \int_{\mathbb{R}^3}\int_{S^2} B(|g|,\cos\theta)\, f(v')\, f(v'_*)\,\mathrm{d}\omega\,\mathrm{d}g\\ &= \int_{\mathbb{R}^3}\int_{S^2} B(|g|,\cos\theta)\, f\Big(\frac{1}{2}(v+v_*+|g|\omega)\Big)\, f\Big(\frac{1}{2}(v+v_*-|g|\,\omega)\Big)\mathrm{d}\omega\,\mathrm{d}g\\ &= \int_{\mathbb{R}^3}\int_{S^2} B(|g|,\cos\theta)\, f\Big(\frac{1}{2}(v+v-g+|g|\omega)\Big)\, f\Big(\frac{1}{2}(v+v-g-|g|\,\omega)\Big)\mathrm{d}\omega\,\mathrm{d}g\\ &= \int_{\mathbb{R}^3}\int_{S^2} B(|g|,\cos\theta)\, f\Big(v+\frac{g}{2}+\frac{|g|\omega}{2}\Big)\, f\Big(v+\frac{g}{2}-\frac{|g|\omega}{2}\Big)\mathrm{d}\omega\,\mathrm{d}g\\ &= \int_{\mathbb{R}^3}\int_{S^2} B(|g|,\cos\theta)\, f\Big(v-\frac{g}{2}+\frac{|g|\omega}{2}\Big)\, f\Big(v-\frac{g}{2}-\frac{|g|\omega}{2}\Big)\mathrm{d}\omega\,\mathrm{d}g\end{aligned}$$

We switch the integration of g over $\mathbb{R}^3$ to spherical coordinates

$$Q_+(f,f)(v) = \int_0^\infty r^2\int_{S^2}\int_{S^2} B(r,\omega\cdot\tilde{\omega})\, f\Big(v-\frac{r\tilde{\omega}}{2}+\frac{r\omega}{2}\Big) f\Big(v-\frac{r\tilde{\omega}}{2}-\frac{r\omega}{2}\Big)\mathrm{d}\omega\,\mathrm{d}\tilde{\omega}\,\mathrm{d}r,$$

revealing a very pretty symmetric form.

We substitute again, using $2u=r\omega$ (implying $r^2\mathrm{d}r\, d\omega=8\mathrm{d}u$), giving

$$Q_+(f,f)(v) = 8\int_{\mathbb{R}^3}\int_{S^2} B(2|u|,\cos\theta)\, f(v-|u|\tilde{\omega}+u)f(v-|u|\tilde{\omega}-u)\,\mathrm{d}\tilde{\omega}\mathrm{d}u.$$

sWhile very similar to the expression we started with, we have unified the dependency of the f-terms on $\tilde{\omega}$. We omit the tilde over $\tilde{\omega}$:

$$Q_+(f,f)(v) = 8\int_{\mathbb{R}^3}\int_{S^2} B(2|u|,\cos\theta)\, f(v-|u|\omega+u)f(v-|u|\omega-u)\,\mathrm{d}\omega\mathrm{d}u.$$

The goal is now to extract this common dependency using the Fourier transform:

$$f(v-|u|\omega+u)f(v-|u|\omega-u) = \int_{\mathbb{R}^3} e^{iv\cdot(v-|u|\omega)}\mathcal{F}^{-1}_{z\to y}[f(z+u)f(z-u)]\mathrm{d}y.$$

Altogether, we obtain

$$Q_+(f,f)(v) = 8\int_{\mathbb{R}^3}\int_{S^2} B(2|u|,\cos\theta)\int_{\mathbb{R}^3} e^{iv\cdot(v-|u|\omega)}\mathcal{F}^{-1}_{z\to y}[f(z+u)f(z-u)]\mathrm{d}y\,\mathrm{d}\omega\mathrm{d}u.$$

Plucking this formula apart yields the representation

$$Q_+(f,f)(v) = \mathcal{F}_{y\to v}\Big[\int_{\mathbb{R}^3} T(u,y)\mathcal{F}_{z\to y}[f(z-u)f(z+u)](u,y)\mathrm{d}u\Big](v), \tag{66}$$

where the transformed kernel T encapsulates the integration over the unit sphere, making it available for precomputation outside the timestepping loop:

$$T(u,y)=8\int_{S^2}B(2|u|,\cos\theta)e^{-i|u|y\cdot\omega}\mathrm{d}\omega.$$

A few remarks are in order:

- It seems we have made a horrible mess of the operator. Where we had integrals over five dimensions before, we now have *nine*. However, six of those are Fourier transforms that only contribute $O(N^3\log N)$ complexity.

  Altogether, the final complexity of this method will turn out to be “only” $O(N^6\log N)$.

- It is an important benefit that we are able to move the unwieldy evaluation of the unit sphere integral into a preprocessing step, where we can invest as much or as little time as we please.

- Some more analysis reveals that the kernel T depends only on $|u|$, $|y|$ and $u\cdot y$. In the VHS case, it even only depends on $|u|$ and $|y|$.

- Carrying out the computations for (66) is reasonably straightforward, since we are only dealing with Cartesian integrals that need to be truncated appropriately. Large savings can be realized by taking advantage of the limited dependencies of T, if used properly. The reader is encouraged to look up the (somewhat messy) details in [29].

- This analysis only treats the (more expensive) gain term. Therefore, the loss term has to be computed separately. We refer to the source article [29] for some ideas.

4.4 The Integral Transform Method by Bobylev-Rjasanow

This method, introduced by Bobylev and Rjasanow in [6], is based on a clever reformulation of the collision operator. To start out, we once again use the substituted operator

$$\begin{aligned} Q(f,f) &= \int_{\mathbb{R}^3}\int_{S^2}B(|v-v_*|,\cos\theta)\left[f(v')\,f(v'_*)-f(v)f(v_*)\right]\mathrm{d}\omega\,\mathrm{d}v_*\\ (g:=v-v_*) \quad &= \int_{\mathbb{R}^3}\int_{S^2}B(|g|,\cos\theta)\left[f(v')\,f(v'_*)-f(v)f(v-g)\right]\mathrm{d}\omega\,\mathrm{d}g. \end{aligned}$$

It turns out that this particular method works only for the hard spheres model ($B(|g|,\theta)=|g|$), but we will delay specializing for a bit, because our initial results will come in handy in Section 4.6.

4.4.1 Some Calculus Magic

Lemma 5. *(Lemma 1 in [6])*

For any “nice enough” test function $\varphi\colon\mathbb{R}^3\to\mathbb{R}$, we have

$$\int_{S^2}\varphi(|g|\omega-g)\mathrm{d}\omega=\frac{1}{|g|}\int_{\mathbb{R}^3}\delta\left(z\cdot g+\frac{1}{2}|z|^2\right)\varphi(z)\mathrm{d}z.$$

What’s happening here is that we’re replacing a two-dimensional (sphere) integral with a three-dimensional (volume) integral that includes a δ-function. Not very special, really–just mechanics, taking into account the special shape of the integrands.

Lemma 6. *(Part 1 of Lemma 2 in [6])*

The collision operator can be represented as

$$Q(f,f)=4\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\tilde{B}(y,z)\delta(z\cdot y)[f(v+z)(f(v+y)-f(v)f(v+y+z)]\mathrm{d}y\,\mathrm{d}z, \tag{67}$$

with

$$\tilde{B}(y,z):=\frac{1}{|y+z|}B\bigg(|y+z|,\frac{y\cdot(y+z)}{|y||y+z|}\bigg).$$

Proof. We define

$$\begin{aligned}\varphi(|g|\omega-g) &:= f(v')f(v_*')-f(v)f(v_*)\\ &= f\bigg(\frac{1}{2}(v+v_*+|g|\omega)\bigg)f\bigg(\frac{1}{2}(v+v_*-|g|\,\omega)\bigg)-f(v)f(v-g)\\ &= f\bigg(\frac{1}{2}(v+v-g+|g|\omega)\bigg)f\bigg(\frac{1}{2}(g+v_*+v_*-|g|\,\omega)\bigg)-f(v)f(v-g)\\ &= f\bigg(v+\frac{|g|\omega-g}{2}\bigg)f\bigg(v_*-\frac{|g|\omega-g}{2}\bigg)-f(v)f(v_*),\end{aligned}$$

where v and v_* are parameters. Applying Lemma 5, we find

$$\begin{aligned}\int_{S^2}\varphi(|g|\omega-g)\mathrm{d}\omega &= \frac{1}{|g|}\int_{\mathbb{R}^3}\delta\bigg(z\cdot g+\frac{1}{2}|z|^2\bigg)\varphi(z)\mathrm{d}z\\ \int_{S^2}\big[f(v')f(v_*')-f(v)f(v_*)\big]\mathrm{d}\omega &= \frac{1}{|g|}\int_{\mathbb{R}^3}\delta\bigg(z\cdot g+\frac{1}{2}|z|^2\bigg)\Big[f\Big(v+\frac{z}{2}\Big)f\Big(v_*-\frac{z}{2}\Big)-f(v)f(v_*)\Big]\mathrm{d}z.\end{aligned}$$

Recognizing the left-hand side as part of the collision integral, we multiply the collision kernel B and integrate to find

$$Q(f,f)=\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\delta\bigg(z\cdot g+\frac{1}{2}|z|^2\bigg)\Big[f\Big(v+\frac{z}{2}\Big)f\Big(v_*-\frac{z}{2}\Big)-f(v)f(v_*)\Big]\mathrm{d}z\mathrm{d}g.$$

Substituting $\tilde{z}:=z/2$ (or $z=2\tilde{z}$) and killing the tilde sign right away gives us

$$\begin{aligned}Q(f,f) &= 2^3\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\delta\bigg(2z\cdot g+\frac{1}{2}|2z|^2\bigg)[f(v+z)f(v_*-z)-f(v)f(v_*)]\mathrm{d}z\mathrm{d}g\\ &= 8\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\frac{\delta\big(z\cdot g+|z|^2\big)}{2}[f(v+z)f(v_*-z)-f(v)f(v_*)]\mathrm{d}z\mathrm{d}g\\ &= 4\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\delta\big(z\cdot g+|z|^2\big)[f(v+z)f(v_*-z)-f(v)f(v_*)]\mathrm{d}z\mathrm{d}g,\end{aligned}$$

taking into account $\delta(\alpha x)=\delta(x)/|\alpha|$.

Next, we substitute $y:=v_*-z-v=-z-g$ (or $v_*=y+z+v$, $g=-y-z$) and obtain

$$\begin{aligned}Q(f,f) &= 4\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\delta(z\cdot(z-y-z))[f(v+z)f(y+z+v-z)-f(v)f(y+z+v)]\mathrm{d}z\mathrm{d}g\\ &= 4\int_{\mathbb{R}^3}\int_{\mathbb{R}^3}\frac{1}{|g|}B(|g|,\cos\theta)\delta(z\cdot y)[f(v+z)f(v+y)-f(v)f(v+y+z)]\mathrm{d}z\mathrm{d}g,\end{aligned}$$

proving the claim. □

Theorem 7. *(Part 2 of Lemma 2 in [6])*

The collision operator for hard spheres *can also be represented as*

$$Q(f,f)=\int_{S^2}\int_{S^2}\delta(\omega_1\cdot\omega_2)[\Phi(v,\omega_1)\Phi(v,\omega_2)-f(v)\Psi(v,\omega_1,\omega_2)]\mathrm{d}\omega_1\,\mathrm{d}\omega_2, \tag{68}$$

where

$$\Phi(v,\omega) := \int_{-\infty}^{\infty} |\rho| f(v+\rho\omega) \mathrm{d}\rho \tag{69}$$

and

$$\Psi(v,\omega_1,\omega_2) := \int_{-\infty}^{\infty} \int_{\infty}^{\infty} |\rho_1||\rho_2| f(v+\rho_1\omega_1+\rho_2\omega_2) \mathrm{d}\rho_1 \mathrm{d}\rho_2. \tag{70}$$

Proof. Introduce spherical coordinates in (67) and use the identity

$$4\delta(\omega_1\cdot\omega_2) = \delta(\omega_1\cdot\omega_2) + \delta((-\omega_1)\cdot\omega_2) + \delta(\omega_1\cdot(-\omega_2)) + \delta((-\omega_1)\cdot(-\omega_2))$$

with the appropriate substitutions to extend the radial integration from $(0,\infty)$ to $(-\infty,\infty)$. □

A few remarks are in order:

- First of all, the point is that (69) and (70) are integrals of convolution type and lend themselves well to computation using Fourier transform techniques.

- Next, a quick count reveals that we've grown an extra integration dimension when compared to (6) and (7) (six now, as compared to five before). This is however counteracted by the presence of the δ-function in (68).

- A closer look at (68) shows that the integration is over all pairs of orthogonal unit vectors ω_1, ω_2. Recognizing this special structure should enable us to apply some cleverness in evaluating the integral.

- Expressions like (69) and (70) are not unheard of in mathematics. They resemble the X-ray and the Radon transform, respectively. [6] calls them *generalized X-ray* and *generalized Radon transforms*.

4.4.2 X-Ray to Fourier

To fix notation, we set the Fourier transform to be

$$\varphi(\xi) = \mathcal{F}[f](\xi) = \int_{\mathbb{R}^3} f(v) e^{i(v\cdot\xi)} \mathrm{d}v$$

and its inverse

$$f(v) = \mathcal{F}^{-1}[\varphi](v) = \frac{1}{(2\pi)^3} \int_{\mathbb{R}^3} \varphi(\xi) e^{-i(v\cdot\xi)} \mathrm{d}\xi.$$

We begin by rewriting Φ:

$$\begin{array}{rrl}
 & \Phi(v,\omega) = & \mathcal{F}^{-1}[\mathcal{F}[\Phi](\xi,\omega)](v,\omega) \\
 & = & \mathcal{F}^{-1}\left[\int_{\mathbb{R}^3} \Phi(v,\omega) e^{i(v\cdot\xi)} \mathrm{d}v\right](v,\omega) \\
 & = & \mathcal{F}^{-1}\left[\int_{\mathbb{R}^3} \int_{-\infty}^{\infty} |\rho| f(v+\rho\omega) \mathrm{d}\rho e^{i(v\cdot\xi)} \mathrm{d}v\right](v,\omega) \\
(\tilde{v} := v+\rho\omega) & = & \mathcal{F}^{-1}\left[\int_{\mathbb{R}^3} \int_{-\infty}^{\infty} |\rho| f(\tilde{v}) e^{i((\tilde{v}-\rho\omega)\cdot\xi)} \mathrm{d}\rho \, \mathrm{d}v\right](v,\omega) \\
 & = & \mathcal{F}^{-1}\left[\int_{\mathbb{R}^3} f(\tilde{v}) e^{i(\tilde{v}\cdot\xi)} \mathrm{d}v \int_{-\infty}^{\infty} |\rho| e^{-i(\rho\omega\cdot\xi)} \mathrm{d}\rho\right](v,\omega) \\
 & = & \mathcal{F}^{-1}[\varphi(\xi) d(\xi\cdot\omega)](v,\omega),
\end{array}$$

with the definition

$$d(\zeta) := \int_{-\infty}^{\infty} |\rho| e^{-i\rho\zeta} \mathrm{d}\rho = \mathcal{F}[|\rho|](\zeta) = -\frac{2}{|\zeta|^2}.$$

Similarly for Ψ:

$$\Psi(v,\omega) = \mathcal{F}^{-1}[\varphi(\xi) d(\xi\cdot\omega_1) d(\xi\cdot\omega_2)](v,\omega_1,\omega_2).$$

The function d, however, is singular at $\zeta = 0$, and therefore not very suitable for our calculations. We therefore approximate

$$d(\zeta) \approx d_R(\zeta) := \int_{-R}^{R} |\rho| e^{-i\rho\zeta} \mathrm{d}\rho = 2\frac{\zeta R\sin(\zeta R) + \cos(\zeta R) - 1}{\zeta^2},$$

with $d_R(0) = R^2$. Note that this approximation has the net effect of truncating the integrals for Φ and Ψ to the regions $[-R, R]$ and $[-R, R]^2$. We add the subscripts Q_R, Φ_R and Ψ_R to denote that the approximate function d_R is being used. The modified collision operator Q_R still verifies several analytic properties of the original Q, such as the conservation laws, the H-theorem and the equilibrium solutions.

This, of course, bears the question of how large R should be chosen. We suppose that $\operatorname{supp}(f) \subset \mathcal{B}(0, R_0)$. Then, applying Proposition 4, we find that $|u| < 2R_0$, and $Q(f,f) = Q_R(f,f)$ for $R \geqslant R_0$. Therefore, if we choose a specific R, then the L^∞ in f error can be approximated by finding its maximum value of f on the sphere $|v| = R/2$. Since f tends to evolve into a Maxwellian $\exp(-\alpha|v|^2)$, we simply choose R such that $\alpha R^2 \gg 4$, for the particular equilibrium in the problem to be solved.

4.4.3 Formula to Algorithm

All in all, we have so far (mostly) analytically rewritten (62) into (68). What remains is to accurately approximate the various parts of (68). We proceed as follows:

1. Compute the Fourier transform φ of f. The comments from Section 4.2.1 apply. In particular, [6] uses the grid shift by bulk velocity.

2. Carry out the integration

$$\int_{S^2}\int_{S^2} \delta(\omega_1\cdot\omega_2) F(\omega_1,\omega_2) \mathrm{d}\omega_1 \mathrm{d}\omega_2.$$

   The main difficulty here is finding a suitable quadrature rule on the sphere. After that, the second integral collapses to a path integral over the circle $\{\omega : \omega\cdot\omega_1 = 0\} \subset S^2$, which is easy to carry out.

   At this stage, it is also useful to notice that the function

$$F(\omega_1,\omega_2) := \Phi(v,\omega_1)\Phi(v,\omega_2) - f(v)\Psi(v,\omega_1,\omega_2)$$

   observes the symmetries

$$F(\omega_1,\omega_2) = F(\pm\omega_1, \pm\omega_2) = F(\pm\omega_2, \pm\omega_1),$$

   which can be used for another substantial reduction of computational effort.

   Now, for each value of (ω_1, ω_2) found by the quadrature, do the following:

   a. Compute

$$F_\Phi(\xi) := \varphi(\xi) d(\xi\cdot\omega), \qquad F_\Psi(\xi) := \varphi(\xi) d(\xi\cdot\omega_1) d(\xi\cdot\omega_2)$$

   from the (pre-computed) Fourier transform φ of f.

   b. Compute the inverse Fourier transform of F_Φ and F_Ψ.

The method obtained in this way achieves second order convergence, and it conserves mass as-is. In order to conserve momentum and energy, it needs an ad-hoc modification to the axial terms of the Fourier transform that works by "making the smallest change" (in L^2 sense) that will conserve both quantities.

4.5 A Spectral Method by Pareschi, Russo and Filbet

In [41], Pareschi et al. propose another Fourier-based scheme for the collision operator, and thus the remarks from Section 4.2.1 apply. For the purposes of their scheme, we assume that our function $f(v)$ has support on $\mathcal{B}(0,R_0)$, and we take the periodicity $L=\pi$ for simplicity. Naturally, we obtain the no-aliasing condition

$$\pi = L \geqslant \frac{3+\sqrt{2}}{2} R_0,$$

as above.

Let $K_N := \{-N, \ldots, N\}$. Not quite unlike 4.2.1, we expand f in Fourier modes, with

$$f_N(v) := \sum_{k\in K_N^3} \hat{f}_k\, e^{i(k\cdot v)}, \tag{71}$$

with the Fourier coefficients $\hat{f}_k$ given by

$$\hat{f}_k = \frac{1}{(2\pi)^3} \int_{[-\pi,\pi]^3} f(v) e^{-i(k\cdot v)} \mathrm{d}v.$$

4.5.1 Derivation

Our method is now defined by asserting that the residual of (61) is L^2-orthogonal to all Fourier modes:

$$\int_{[-\pi,\pi]^3} \left(\frac{\partial f_N}{\partial t} - \left[Q^+(f_N,f_N) - L(f_N) f_N\right]\right) e^{-ik\cdot v} \mathrm{d}v \overset{!}{=} 0 \quad \text{for all } k\in K_N^3. \tag{72}$$

Substituting (71) into (72) gives us (while setting $\mathcal{B} := \mathcal{B}(0, 2R_0)$)

$$\begin{aligned}
& 0 \\
\overset{!}{=}\ & \int_{[-\pi,\pi]^3} \left(\frac{\partial f_N}{\partial t} - \left[Q^+(f_N,f_N) - L(f_N) f_N\right]\right) e^{-i(k\cdot v)} \mathrm{d}v \\
=\ & \int_{[-\pi,\pi]^3} \left(\frac{\partial f_N}{\partial t} - \int_{\mathcal{B}} \int_{S^2} B(|v-v_*|, \cos\theta)\left[\, f_N(v')\, f_N(v_*') - f_N(v) f_N(v_*)\right] \mathrm{d}\omega\, \mathrm{d}v_*\right) e^{-i(k\cdot v)} \mathrm{d}v \\
=\ & \int_{[-\pi,\pi]^3} \left(\frac{\partial f_N}{\partial t} - \int_{\mathcal{B}} \int_{S^2} B(|g|, \cos\theta)\left[\, f_N(v')\, f_N(v_*') - f_N(v) f_N(v-g)\right] \mathrm{d}\omega\, \mathrm{d}g\right) e^{-i(k\cdot v)} \mathrm{d}v \\
=\ & \int_{[-\pi,\pi]^3} \Bigg(\left(\sum_{k''\in K_N^3} \frac{\partial \hat{f}_{k''}}{\partial t} e^{i(k''\cdot v)}\right) - \int_{\mathcal{B}} \int_{S^2} B(|g|, \cos\theta) \Bigg[\left(\sum_{k'\in K_N^3} \hat{f}_{k'}\, e^{i(k'\cdot v')}\right)\left(\sum_{k_*'\in K_N^3} \hat{f}_{k_*'}\, e^{i(k_*'\cdot v_*')}\right) - \\
& \left(\sum_{k''\in K_N^3} \hat{f}_{k''}\, e^{i(k''\cdot v)}\right)\left(\sum_{k_*\in K_N^3} \hat{f}_{k_*}\, e^{i(k_*\cdot(v-g))}\right)\Bigg] \mathrm{d}\omega\, \mathrm{d}g \Bigg) e^{-i(k\cdot v)} \mathrm{d}v \\
=\ & \int_{[-\pi,\pi]^3} \Bigg(\left(\sum_{k''\in K_N^3} \frac{\partial \hat{f}_{k''}}{\partial t}\ e^{i(k''\cdot v - k\cdot v)}\right) - \int_{\mathcal{B}} \int_{S^2} B(|g|,\ \cos\ \theta) \Bigg[\sum_{k',k_*'\in K_N^3} \hat{f}_{k'} \hat{f}_{k_*'}\ e^{i(k'\cdot v' + k_*'\cdot v_*' - k\cdot v)} \ - \\
& \sum_{k'',k_*\in K_N^3} \hat{f}_{k''}\, \hat{f}_{k_*}\, e^{i(k''\cdot v + k_*\cdot v - k_*\cdot g - k\cdot v)} \Bigg] \mathrm{d}\omega\, \mathrm{d}g \Bigg) \mathrm{d}v
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{k''\in K_N^3} \frac{\partial \hat{f}_{k''}}{\partial t} \int_{[-\pi,\pi]^3} e^{i(k''\cdot v - k\cdot v)}\mathrm{d}v - \int_{[-\pi,\pi]^3}\int_{\mathcal{B}}\int_{S^2} B(|g|,\cos\theta)\bigg[\sum_{k',k'_*\in K_N^3} \hat{f}_{k'}\hat{f}_{k'_*}\, e^{i(k'\cdot v' + k'_*\cdot v'_* - k\cdot v)} - \\
&\quad \sum_{k'',k_*\in K_N^3} \hat{f}_{k''}\,\hat{f}_{k_*}\, e^{i(k''\cdot v + k_*\cdot v - k_*\cdot g - k\cdot v)}\bigg]\mathrm{d}\omega\,\mathrm{d}g\,\mathrm{d}v \\
&= \sum_{k''\in K_N^3} \frac{\partial \hat{f}_{k''}}{\partial t}\ \delta_{k'',k} - \int_{[-\pi,\pi]^3}\int_{\mathcal{B}}\int_{S^2} B(|g|,\ \cos\ \theta)\bigg[\sum_{k',k'_*\in K_N^3} \hat{f}_{k'}\hat{f}_{k'_*}\ e^{i(k'\cdot v' + k'_*\cdot v'_* - k\cdot v)} - \\
&\quad \sum_{k,k_*\in K_N^3} \hat{f}_{k''}\,\hat{f}_{k_*}\, e^{i(k''\cdot v + k_*\cdot v - k_*\cdot g - k\cdot v)}\bigg]\mathrm{d}\omega\,\mathrm{d}g\,\mathrm{d}v \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{k',k'_*\in K_N^3} \hat{f}_{k'}\hat{f}_{k'_*} \int_{\mathcal{B}}\int_{S^2}\int_{[-\pi,\pi]^3} B(|g|,\ \cos\ \theta)\ e^{i(k'\cdot v' + k'_*\cdot v'_* - k\cdot v)}\mathrm{d}v\ \ \mathrm{d}\omega\ \ \mathrm{d}g - \\
&\quad \sum_{k'',k_*\in K_N^3} \hat{f}_{k''}\,\hat{f}_{k_*}\int_{\mathcal{B}}\int_{S^2}\int_{[-\pi,\pi]^3} e^{i(k''\cdot v + k_*\cdot v - k_*\cdot g - k\cdot v)}\mathrm{d}v\,\mathrm{d}\omega\,\mathrm{d}g\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{l,m\in K_N^3} \hat{f}_l\hat{f}_m \int_{\mathcal{B}}\int_{S^2}\int_{[-\pi,\pi]^3} B(|g|,\cos\theta)\, e^{i\left[l\cdot\left(\frac{1}{2}(v+v_*+|g|\omega)\right)+m\cdot\left(\frac{1}{2}(v+v_*-|g|\omega)\right)-k\cdot v\right]}\mathrm{d}v\,\mathrm{d}\omega\,\mathrm{d}g - \\
&\quad \sum_{l,m\in K_N^3} \hat{f}_l\,\hat{f}_m \int\int_{S^2}\int_{[-\pi,\pi]^3} e^{i(l\cdot v + m\cdot v - m\cdot g - k\cdot v)}\mathrm{d}v\,\mathrm{d}\omega\,\mathrm{d}g\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{l,m\in K_N^3} \hat{f}_l\hat{f}_m \int_{\mathcal{B}}\int_{S^2}\int_{[-\pi,\pi]^3} B(|g|,\theta)\, e^{i\left[l\cdot\left(\frac{1}{2}(v+v-g+|g|\omega)\right)+m\cdot\left(\frac{1}{2}(v+v-g-|g|\omega)\right)-k\cdot v\right]}\mathrm{d}v\,\mathrm{d}\omega\,\mathrm{d}g - \\
&\quad \sum_{l,m\in K_N^3} \hat{f}_l\,\hat{f}_m \int_{\mathcal{B}}\int_{S^2} e^{i(-m\cdot g)}\delta_{l+m,k}\mathrm{d}\omega\,\mathrm{d}g\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m \int_{\mathcal{B}}\int_{S^2} B(|g|,\ \cos\ \theta)\ e^{i\left[l\cdot\left(\frac{1}{2}(-g+|g|\omega)\right)+m\cdot\left(\frac{1}{2}(-g-|g|\omega)\right)\right]}\mathrm{d}\omega\ \ \mathrm{d}g - \\
&\quad \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\,\hat{f}_m \int_{\mathcal{B}}\int_{S^2} e^{i(-m\cdot g)}\mathrm{d}\omega\,\mathrm{d}g\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m \int_{\mathcal{B}}\int_{S^2} B(|g|,\ \cos\ \theta)\ e^{-ig\cdot\frac{l+m}{2}-i|g|\frac{m-l}{2}}\mathrm{d}\omega\ \ \mathrm{d}g - \\
&\quad \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\,\hat{f}_m \int_{\mathcal{B}}\int_{S^2} e^{i(-m\cdot g)}\mathrm{d}\omega\,\mathrm{d}g\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m\hat{B}(l,m) - \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\,\hat{f}_m\,\hat{B}(m,m)\bigg] \\
&= \frac{\partial \hat{f}_k}{\partial t} - \bigg[\sum_{m\in K_N^3} \hat{f}_{k-m}\hat{f}_m\hat{B}(k-m,m) - \sum_{m\in K_N^3} \hat{f}_{k-m}\,\hat{f}_m\,\hat{B}(m,m)\bigg],
\end{aligned}
$$

where we have defined the *kernel modes* $\hat{B}(l,m)$ as

$$\hat{B}(l,m) := \int_{\mathcal{B}(0,2R_0)}\int_{S^2} B(|g|,\cos\theta)\, e^{-ig\cdot\frac{(l+m)}{2}-i|g|\omega\cdot\frac{(m-l)}{2}}\mathrm{d}\omega\,\mathrm{d}g. \tag{73}$$

We repeat the scheme for clarity:

$$\frac{\partial \hat{f}_k}{\partial t} = \sum_{m\in K_N^3} \hat{f}_{k-m}\hat{f}_m\hat{B}(k-m,m) - \sum_{m\in K_N^3} \hat{f}_{k-m}\,\hat{f}_m\,\hat{B}(m,m). \tag{74}$$

Again, now is the time to note down a few properties of this scheme:

- $\hat{B}(l,m)$ is a function of $|l-m|$, $|l+m|$, $(l-m)\cdot(l+m)$. (Proposition 3.2 in [41])
- For the VHS kernel, $\hat{B}(l,\,m)$ is a function of $|l\,-\,m|$, $|l\,+\,m|$. For this important case, there are only two parameters to $\hat{B}$, allowing it to be precomputed and stored in a two-dimensional array.
- Because we can pre-evaluate $\hat{B}$, evaluation of (74) takes $O(N^6)$ operations.

  In contrast, a straightforward method would take $O(N^6 N_a)$, where N_a is the number of directions accounted for in a discretization of the unit sphere.
- The loss term is actually a convolution sum, and may therefore be evaluated in $O(N^3\log\,N)$, making the gain term (sensibly) the largest consumer of computational resources.

It turns out that for the VHS kernels in two and three dimensions, the evaluation of (73) can be simplified drastically. We follow the two-dimensional derivation as an example.

4.5.2 Kernel Modes

Recall that for the VHS kernel, we are seeking a more explicit formula for the expression

$$\hat{B}_{\mathrm{VHS}}(l,m)=C^{\alpha}\int_{\mathcal{B}(0,2R_0)}|g|^{\alpha}\,e^{-ig\cdot\frac{(l+m)}{2}}\underbrace{\int_{S^2}e^{-i|g|\omega\cdot\frac{(m-l)}{2}}\mathrm{d}\omega}_{I_2(|g|,l-m):=}\,\mathrm{d}g. \tag{75}$$

For the two-dimensional problem, we obtain

$$\begin{aligned}
I_2(|g|,l-m) &= \int_{S^1}\exp\Big(-i|g|\omega\cdot\frac{(m-l)}{2}\Big)\mathrm{d}\omega\\
&= \int_{S^1}\exp\Big(-i|g|\omega\cdot e_1\frac{|l-m|}{2}\Big)\mathrm{d}\omega\\
\Big(r:=|g|\frac{|l-m|}{2}, e_1:=(1,0)^T\Big)\qquad &= \int_0^{2\pi}\exp(i\,r\cos\theta\,)\mathrm{d}\theta\\
&= 2\int_0^{2\pi}\cos(r\cos\theta\,)\mathrm{d}\theta\\
&= 2\pi J_0(r).
\end{aligned}$$

Plugging this result into (75) and introducing polar coordinates yields

$$\begin{aligned}
\hat{B}_{\mathrm{VHS}}(l,m) &= 2\pi C^{\alpha}\int_{\mathcal{B}(0,2R_0)}|g|^{\alpha}\exp\Big(-i\,g\cdot\frac{(l+m)}{2}\Big)J_0\Big(|g|\frac{|l-m|}{2}\Big)\,\mathrm{d}g\\
&= 2\pi C^{\alpha}\int_0^{2R_0}\rho^{1+\alpha}\int_0^{2\pi}\cos\Big(|l+m|\frac{\rho}{2}\Big)\cos\theta\mathrm{d}\theta\,J_0\Big(|g|\frac{|l-m|}{2}\Big)\,\mathrm{d}\rho\\
&= 4\pi^2 C^{\alpha}\int_0^{2R_0}\rho^{1+\alpha}J_0\Big(|l+m|\frac{\rho}{2}\Big)J_0\Big(|g|\frac{|l-m|}{2}\Big)\,\mathrm{d}\rho\\
&= 4\pi^2(2R_0)^{2+\alpha}\int_0^1 r^{1+\alpha}J_0(\xi r)J_0(\eta r)\mathrm{d}r,
\end{aligned}$$

with $\xi=|l+m|R_0$ and $\eta=|l-m|R_0$, allowing a very simple evaluation of $\hat{B}_{\mathrm{VHS}}$.

A similar analysis works in three dimensions.

4.5.3 Numerical Analysis

The question for the accuracy of the scheme is (quite positively) answered by the following

Theorem 8. *(Theorem 5.3 in [41])*

Let $f \in L^2([-\pi, \pi]^3)$. Then

$$\left\| Q^{\mathrm{tr}}(f, f) - \mathcal{P}_N Q^{\mathrm{tr}}(f, f) \right\|_2 \leqslant C \left(\| f - f_N \|_2 + \frac{\left\| Q^{\mathrm{tr}}(f_N, f_N) \right\|_{H^r_{\mathrm{per}}}}{N^r} \right)$$

for all $r \geqslant 0$, where H^r_{per} is the Sobolev space of order r containing only periodic functions, and Q^{tr} represents the truncated collision operator.

Stability can be proven once a certain smoothing (i.e. filtering, in Fourier space) operator is incorporated in the scheme. We refer to [41] for the details.

4.6 The Sub-N^6 Mouhot-Pareschi Method

In 2006, Mouhot and Pareschi introduced another method [19, 35] that cleverly combines the ideas from Sections 4.4 and 4.5. In doing so, it does achieve the important milestone of computing the entire collision operator in less than $O(N^6)$ operations, i.e. less than "any velocity coupled with any other".

Recall the statement of Lemma 6, slightly generalized for arbitrary dimensions:

Lemma. *The collision operator can be represented as*

$$Q(f, f) = \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \tilde{B}(y, z) \delta(z \cdot y) [f(v + z) f(v + y) - f(v) f(v + y + z)] \mathrm{d}y \, \mathrm{d}z, \tag{76}$$

with

$$\tilde{B}(y, z) := \frac{2^{d-1}}{|y + z|^{d-2}} B\left(|y + z|, \frac{y \cdot (y + z)}{|y||y + z|} \right).$$

The idea of this method is to use (76) instead of the "untreated" collision operator to derive a spectral method. Looking at the operator in this way yields better decoupling properties between the arguments of the operator, as we will see later.

The next job is to appropriately truncate the integration domains. Once again, we consider the periodic domain $[-L, L]^d$. If $\operatorname{supp}(f) \subset \mathcal{B}(0, R_0)$, then similar arguments as above show that we may truncate both integrations above to $\mathcal{B}(0, R)$, with $R = \sqrt{2} R_0$, while keeping $L \geqslant (3 + \sqrt{2})/2 \, R_0$ as the period. We obtain

$$Q^R(f, f) := \int_{\mathcal{B}(0, R)} \int_{\mathcal{B}(0, R)} \tilde{B}(y, z) \delta(z \cdot y) [f(v + z) f(v + y) - f(v) f(v + y + z)] \mathrm{d}y \, \mathrm{d}z.$$

Once truncated, we can derive the spectral scheme. The symbols f_N, $\hat{f}_k$, and K_N are as in Section 4.5. As we did there, we assume $L = \pi$. Again, we start by assuming that the residual of the evolution equation (61) is orthogonal to the space $\{v \mapsto e^{-ik \cdot v} : k \in K_N\}$.

$$\begin{aligned}
& 0 \\
\overset{!}{=} \; & \int_{[-\pi, \pi]^d} \left(\frac{\partial f_N}{\partial t} - \left[Q^+(f_N, f_N) - L(f_N) f_N \right] \right) e^{-ik \cdot v} \mathrm{d}v \\
= \; & \int_{[-\pi, \pi]^d} \left(\sum_{k' \in K_N^d} \frac{\partial \hat{f}_{k'}}{\partial t} e^{i(k' \cdot v)} - Q^R(f_N, f_N) \right) e^{-ik \cdot v} \mathrm{d}v \\
= \; & \frac{\partial \hat{f}_k}{\partial t} - \int_{[-\pi, \pi]^d} \left[Q^R(f_N, f_N) \right] e^{-ik \cdot v} \mathrm{d}v
\end{aligned}$$

$$
\begin{aligned}
&= \frac{\partial \hat{f}_k}{\partial t} - \int_{[-\pi,\pi]^d} \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y)[f_N(v+z)f_N(v+y) - f_N(v)f_N(v+y+z)]e^{-ik\cdot v}\mathrm{d}y\,\mathrm{d}z\,\mathrm{d}v\\
&= \frac{\partial \hat{f}_k}{\partial t} - \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y) \int_{[-\pi,\pi]^d} \sum_{l,m\in K_N^3} \hat{f}_l\hat{f}_m\, e^{i[l\cdot(v+z)+m\cdot(v+y)-k\cdot v]} - \sum_{l,m\in K_N^3} \hat{f}_l\hat{f}_m\, e^{i[l\cdot v+m\cdot(v+y+z)-k\cdot v]}\mathrm{d}v\,\mathrm{d}y\,\mathrm{d}z\\
&= \frac{\partial \hat{f}_k}{\partial t} - \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y) \sum_{l,m\in K_N^3} \hat{f}_l\hat{f}_m\delta_{l+m,k}\Big[e^{i[l\cdot z+m\cdot y]} - e^{im\cdot(y+z)}\Big]\mathrm{d}y\,\mathrm{d}z\\
&= \frac{\partial \hat{f}_k}{\partial t} - \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y)\Big[e^{i[l\cdot z+m\cdot y]} - e^{im\cdot(y+z)}\Big]\mathrm{d}y\,\mathrm{d}z\\
&= \frac{\partial \hat{f}_k}{\partial t} - \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m\,\hat{\beta}(l,m),
\end{aligned}
$$

where

$$\hat{\beta}(l,m) := \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y)\Big[e^{i[l\cdot z+m\cdot y]} - e^{im\cdot(y+z)}\Big]\mathrm{d}y\,\mathrm{d}z.$$

We repeat the scheme for clarity:

$$\frac{\partial \hat{f}_k}{\partial t} = \sum_{l,m\in K_N^3, l+m=k} \hat{f}_l\hat{f}_m\,\hat{\beta}(l,m) \tag{77}$$

and notice that the $\hat{\beta}(l,m)$ can be simplified further by writing

$$\hat{\beta}(l,m) = \beta(l,m) - \beta(m,m)$$

with

$$\beta(l,m) := \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \tilde{B}(y,z)\delta(z\cdot y)\Big[e^{i[l\cdot y+m\cdot z]}\Big]\mathrm{d}y\,\mathrm{d}z = \beta(m,l). \tag{78}$$

We have a few preliminary remarks before we carry on:

- Recall that the scheme (74) that we derived in Section 4.5 had exactly the same shape. So how do we expect to save computational time here with a scheme that looks exactly the same? Recall that in Section 4.5 we truncated the integration in $g = y + z$, so our truncated operator there would look like this in our current representation:

  $$Q_{\text{old}}^R(f,f) = \int_{\mathcal{B}(0,R)} \int_{\mathcal{B}(0,R)} \underbrace{\mathbf{1}_{|y+z|\leqslant R}}_{\text{extra term}} \tilde{B}(y,z)\delta(z\cdot y)[f(v+z)f(v+y) - f(v)f(v+y+z)]\mathrm{d}y\,\mathrm{d}z.$$

  It is exactly this extra characteristic function term that causes the variables y and z to couple unnecessarily, and this is exactly where we save (a lot of) time here.

- These savings obviously have to come from a careful analysis of the kernel modes $\beta(l,m)$ to allow simplification of (77). And that is exactly what we will tackle next.

We introduce spherical coordinates:

$$\beta(l,m) = \int_{S^{d-1}} \int_{S^{d-1}} \delta(\omega\cdot\omega') \int_0^R \int_0^R |\rho|^{d-2}|\rho'|^{d-2}\tilde{B}(|\rho|,|\rho'|)\Big[e^{i[l\cdot\rho\omega+m\cdot\rho'\omega']}\Big]\mathrm{d}\rho\,\mathrm{d}\rho'\,\mathrm{d}\omega\mathrm{d}\omega',$$

where we note that

a) due to the orthogonality enforced by the δ-function, $\tilde{B}$ only depends on the moduli ρ and ρ',

b) the same peculiarity with the δ-function as encountered in the proof of Lemma 5 reduces the exponent from the usual ρ^{d-1} to ρ^{d-2}.

Now we extend the inner integrals from $(0,R)$ to $(-R,R)$, dividing by four to take care of the three extra terms introduced.

$$\beta(l,m)=\frac{1}{4}\int_{S^{d-1}}\int_{S^{d-1}}\delta(\omega\cdot\omega')\int_{-R}^{R}\int_{-R}^{R}|\rho|^{d-2}|\rho'|^{d-2}\tilde{B}(|\rho|,|\rho'|)\Big[e^{i[l\cdot\rho\omega+m\cdot\rho'\omega']}\Big]\mathrm{d}\rho\,\mathrm{d}\rho'\,\mathrm{d}\omega\mathrm{d}\omega'. \tag{79}$$

The last key ingredient in the efficient evaluation of (79) plugged into (77) is the "*decoupling assumption*"

$$\tilde{B}(y,z)=a(|y|)b(|z|)\quad\text{for}\quad y\perp z,$$

which, for us, boils down to

$$\tilde{B}(\rho,\rho')=a(\rho)b(\rho').$$

Obviously, Maxwellian molecules in 2D and Hard Spheres in 3D satisfy this assumption, because for these $\tilde{B}(y,z)$ is just a constant. Other cases where this can be made to happen are enumerated in the appendix of [35]. We specialize to three dimensions and find

$$\begin{aligned}\beta(l,m) &= \frac{1}{4}\int_{S^2}\int_{S^2}\delta(\omega\cdot\omega')\int_{-R}^{R}\rho a(\rho)e^{il\cdot\rho\omega}\mathrm{d}\rho\int_{-R}^{R}\rho' b(\rho')e^{im\cdot\rho'\omega'}\mathrm{d}\rho\,\mathrm{d}\rho'\,\mathrm{d}\omega\mathrm{d}\omega'\\ &= \frac{1}{4}\int_{S^2}\int_{-R}^{R}\rho a(\rho)e^{il\cdot\rho\omega}\mathrm{d}\rho\int_{S^2\cap\omega^{\perp}}\int_{-R}^{R}\rho' b(\rho')e^{im\cdot\rho'\omega'}\mathrm{d}\rho'\mathrm{d}\omega'\,\mathrm{d}\rho\,\mathrm{d}\omega\\ &= \frac{1}{4}\int_{S^2}\varphi_a(l\cdot\omega)\int_{S^2\cap\omega^{\perp}}\varphi_b(m\cdot\omega')\mathrm{d}\omega'\,\mathrm{d}\rho\,\mathrm{d}\omega\end{aligned}$$

where

$$\varphi_f(s):=\int_{-R}^{R}|\rho|f(|\rho|)e^{i\rho s}\mathrm{d}\rho,$$

and we notice $\varphi_f(s)=\varphi_f(-s)$. Therefore, it is sufficient to integrate over a hemisphere $(S^2)^+$, which gives us

$$\beta(l,m)=\int_{(S^2)^+}\varphi_a(l\cdot\omega)\int_{S^2\cap\omega^{\perp}}\varphi_b(m\cdot\omega')\mathrm{d}\omega'\,\mathrm{d}\rho\,\mathrm{d}\omega.$$

As a last step, we notice that the function

$$\omega\quad\mapsto\quad\varphi_a(l\cdot\omega)\int_{S^2\cap\omega^{\perp}}\varphi_b(m\cdot\omega')$$

is periodic on $(S^2)^+$. Therefore, even the simple rectangle rule quadrature with a parametrization

$$\theta:=p\pi/M,\quad\varphi:=q\pi/M$$

of the hemisphere has infinite order:

$$\beta(l,m)\approx\tilde{\beta}(l,m):=\frac{\pi^2}{M^2}\sum_{p,q=0}^{M}\underbrace{\varphi_a(l\cdot\omega_{\theta,\varphi})}_{\alpha_{p,q}(l):=}\underbrace{\int_{S^2\cap\omega_{\theta,\varphi}^{\perp}}\varphi_b(m\cdot\omega')}_{\alpha'_{p,q}(m):=}.$$

The specific structure of $\beta(l, m)$ is the key to the speed of the method. Observe:

$$\begin{aligned}\frac{\partial \hat{f}_k}{\partial t} &= \frac{\pi^2}{M^2} \sum_{l, m \in K_N^3, l + m = k} \hat{f}_l \hat{f}_m \sum_{p, q = 0}^{M} \alpha_{p, q}(l) \alpha'_{p, q}(m) \quad - \quad \text{loss}\\ &= \frac{\pi^2}{M^2} \sum_{m \in K_N^3} \hat{f}_{k - m} \hat{f}_m \sum_{p, q = 0}^{M} \alpha_{p, q}(k - m) \alpha'_{p, q}(m) \quad - \quad \text{loss}\\ &= \frac{\pi^2}{M^2} \sum_{p, q = 0}^{M} \sum_{m \in K_N^3} \hat{f}_{k - m} \alpha_{p, q}(k - m) \hat{f}_m \alpha'_{p, q}(m) \quad - \quad \text{loss.}\end{aligned}$$

Now this is a sum of M^2 convolutions, which can be carried out in $M^2 N^3 \log N$ operations, using an FFT, giving, to the best of our knowledge, the currently smallest asymptotic complexity of any deterministic method for the collision operator.

Again, we close with a few remarks:

- As shown in [35], the method is spectrally accurate.

- The only “sophisticated” numerical technique required in the implementation of this method is the FFT. Therefore, the the net implementation effort is still reasonably limited.

- To be able to prove the usual conservation properties, the effective quadrature $(\mathcal{A}, \mathrm{d}\mathcal{A}(\omega, \omega'))$ on the set

  $$\{(\omega, \omega') \in S^{d-1} \times S^{d-1} : \omega \cdot \omega' = 0\}$$

  needs to be even, i.e. satisfy

  $$(\omega, \omega') \in \mathcal{A} \quad \Rightarrow \quad (\omega, -\omega'), (-\omega, \omega'), (-\omega, -\omega') \in \mathcal{A},$$

  and the discrete measure $\mathrm{d}\mathcal{A}(\omega, \omega')$ also needs to be even.

4.7 Summary

Method	Ref.	Maxw.	HS	VHS	Other	Order	Complexity	C.Mass	C.Momt.	C.Energy
DSMC	[5]	✓	✓	✓	✓	“$N^{-1/2}$”	N^3	✓	✓	✓
Ibragimov-R.	[29]	✓	✓	✓	–	N^{-2}	N^6	✓	(✓) [1]	(✓) [1]
general case [3]	[29]	–	–	–	✓	N^{-2}	$N^6 \log N$	✓	(✓) [1]	(✓) [1]
Bobylev-R.	[6]	–	3D	–	–	N^{-2}	$N^6 \log N$	✓	(✓) [1]	(✓) [1]
Pareschi-Russo	[41]	✓	✓	✓	–	spectral	N^6	✓	(✓) [2]	(✓) [2]
Mouhot-P.	[35]	2D	3D	–	–	spectral	$M^2 N^3 \log N$	✓	(✓) [2]	(✓) [2]

Legend:

***N*.** N is the number of velocity discretization points in one dimension.

***M*.** M is a fixed number of discretization angles.

Order. Implicitly in $O(\,\cdot\,)$-notation.

Complexity. Implicitly in $O(\,\cdot\,)$-notation.

Recall that the collision operator is local in space, i.e. its value at x only depends values at x. "Complexity" here is the number of operations required to calculate the collision operator at one point in space. To simulate the whole Boltzmann equation, one would need to do this P^3 times, where P is the number of points in the one-dimensional spatial discretization.

If we assume $P=M=N$, the fastest of these methods bring the naïve N^{11} complexity down to $N^8 \log N$. Even if we assume only a measly 10 points per dimension, that's a win by almost three orders of magnitude!

[1]. These conservations only hold if an extra correction is applied.

[2]. Conservation holds up to the order of the scheme.

[3]. The Ibragimov-Rjasanow scheme is split into two rows: The (faster) VHS case and the (slower) general case.

4.8 Time Discretization for the Collision Operator

Standard time discretizations, such as multi-stage Runge-Kutta or multi-step methods, while sometimes used to advance (61) in time, often fail to keep f positive, while having the added drawback of restricting the timestep to rather small values.

Therefore, one is often better served taking advantage of the considerable freedom in choosing time discretizations afforded by the shape of (61). In particular, since each spatial point x is independent of its surroundings, we are free to choose separate timesteps or even separate timestepping *methods* at each point in space.

Among just the standard methods, Filbet and Russo [16] favor multi-step over multi-stage methods for improved efficiency. In general however, they advocate the use of a class of special-purpose schemes, called a *time-relaxed (TR) schemes* (see [43]). These schemes are designed to maintain the positivity of f while permitting large timesteps.

To construct a TR scheme, we begin by assuming that we want to advance in time an equation of the shape

$$\frac{\partial f}{\partial t} = \frac{1}{k_n}[P(f,f) - \mu f], \tag{80}$$

where $P(f,f)$ is a positive bilinear operator, and $\mu > 0$ is a constant. The point of this representation is that the constant μ puts an upper bound on the "negative part" (i.e. the loss term) of the collision operator.

For a Maxwellian gas, the kernel B does not depend on velocity and therefore Q already has the form (80). For more general kernels, we would need to find a μ to satisfy

$$\mu \geqslant L[f](v) \quad \text{for all } v \in \mathbb{R}^3. \tag{81}$$

Then we could rewrite (61) in the form (80) by setting

$$P(f,f) = Q^+(f,f) + f(v)(\mu - L[f]).$$

Such a μ might not exist as $L[f]$ is in general unbounded. Even if we assume f to have compact support, choosing a μ to satisfy (81) would lead to excessive numerical viscosity. Therefore, one seeks a μ just big enough to keep P positive. Various heuristics (see [43, 16]) exist to achieve that.

We begin by introducing a change of variables:

$$\tau := 1 - e^{-\mu t/k_n}, \qquad F(\tau, v) := f(t,v) e^{\mu t/k_n}. \tag{82}$$

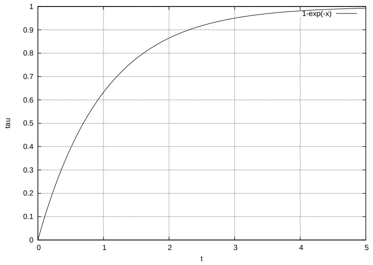

Figure 9. τ vs. t in (82).

It is not hard to see that (82) changes (80) into

$$\frac{\partial F}{\partial \tau} = \frac{1}{\mu} P(F, F). \tag{83}$$

Formally expanding F in a power series in t yields

$$F(\tau, v) = \sum_{k=0}^{\infty} \tau^k \varphi_k(v), \tag{84}$$

and we immediately see that $\varphi_0(v) = f(x, v, 0)$. Substituting (84) into (83) gives a recurrence formula for φ_{k+1}:

$$\varphi_{k+1}(v) = \frac{1}{k+1} \sum_{l=0}^{k} \frac{1}{\mu} P(\varphi_l, \varphi_{k-l}). \tag{85}$$

It turns out that $\varphi_k \rightarrow f_\infty$ as $k \rightarrow \infty$, where f_∞ is the Maxwellian. Truncating (84) and adding the "limit term" of the series back on gives

$$f^{n+1}(v) = (1-\tau) \sum_{k=0}^{m} \tau^k \varphi_k^n(v) + \tau^{m+1} f_\infty(v), \tag{86}$$

where φ_k^n denotes φ_k as in (85), but calculated from f^n, the value of f at time step n. Note that the factor $(1-\tau)$ is added to ensure that

$$1 = (1-\tau) \sum_{k=0}^{m} \tau^k + \tau^{m+1}.$$

Other choices of the coefficients of φ_k and f_∞ are conceivable, see [16]. We examine the properties of this scheme:

Proposition 9. *(cobbled together from [16] and [43])*

The scheme (86) has the following properties:

1. *It is consistent with (80) of order m.*

2. *For $m \in \{1, 2\}$, these schemes are A-stable.*

3. *If f^n is nonnegative, then f^{n+1} is also nonnegative.*

4. *Certain integrals are preserved: If g is a function such that*

$$\int_{\mathbb{R}^3} P(f, f) g \mathrm{d}v = \mu \int_{\mathbb{R}^3} f g \mathrm{d}v,$$

then

$$\int_{\mathbb{R}^3} f^{n+1} g \mathrm{d}v = \int_{\mathbb{R}^3} f^n g \mathrm{d}v.$$

5. *The scheme preserves the asymptotic behavior of (80):*

$$\lim_{\mu\Delta t/k_n \to \infty} f^{n+1}(v) = f_\infty(v).$$

Bibliography

[1] A. Arakawa. Computational Design for Long-Term Numerical Integration of the Equations of Fluid Motion: Two-Dimensional Incompressible Flow. Part I. *Journal of Computational Physics*, 1(1):119–143, 1966.

[2] U. Ascher, S. Ruuth, and J. Spiteri. Implict-explicit Runge-Kutta methods for time dependent Partial Differential Equations. *Applied Numerical Mathematics*, 25:151–167, 1997.

[3] U. Ascher, S.J. Ruuth, and B.T.R. Wetton. Implicit-Explicit Methods for Time-Dependent Partial Differential Equations. *SIAM Journal on Numerical Analysis*, 32(3):797–823, 1995.

[4] N. Besse and E. Sonnendruecker. Semi-Lagrangian schemes for the Vlasov equation on an unstructured mesh of phase space. *Journal of Computational Physics*, 191:341–376, 2003.

[5] G. A. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows*. Oxford University Press, USA, 2rev ed edition, June 1994.

[6] A. V. Bobylev and S. Rjasanow. Fast deterministic method of solving the boltzmann equation for hard spheres. *European Journal of Mechanics - B/Fluids*, 18:869–887, September 1999.

[7] S. Boscarino. Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems. *SIAM Journal on Numerical Analysis*, 45(4):1600–1621, 2007.

[8] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, 2002.

[9] J. Butcher. *The numerical analysis of Ordinary differential equations. Runge-Kutta and general linear methods*. John Wiley & Sons, Chirchester and New York, 1987.

[10] J.A. Carrillo and F. Vecil. Nonoscillatory interpolation methods applied to Vlasov-based models. *SIAM Journal of Scientific Computing*, 29:1179–1206, 2007.

[11] C.Z. Cheng and G. Knorr. The Integration of the Vlasov Equation in Configuration Space. *Journal of Computational Physics*, 22(3):330–351, 1976.

[12] B. Cockburn, S.-Y. Lina, and C.-W. Shu. TVB Runge-Kutta Local Projection Discontinuous Galerkin finite element method for conservation laws III: One-dimensional Systems. *Journal of Computational Physics*, 84:90–113, 1989.

[13] B.O. Dia and M. Schatzman. Commutateur de certains semi-groupes holomorphes et applications aux directions alternées. *Mathematical Modelling and Numerical Analysis*, 30:343–383, 1996.

[14] E. Fijalkow. A numerical solution to the Vlasov equation. *Computer Physics Communications*, 116:319–328, 1999.

[15] F. Filbet and G. Russo. High order numerical methods for the space non-homogeneous Boltzmann equation. *Journal of Computational Physics*, 186:457–480, 2003.

[16] F. Filbet and G. Russo. *Modeling and Computational Methods for Kinetic Equations*, pages 117–145. Modeling and Simulation in Science, Engineering and Technology. Birkhäuser, Boston, 2004.

[17] F. Filbet and E. Sonnendruecker. Comparison of Eulerian Vlasov solvers. *Computer Physics Communications*, 150:247–266, 2003.

[18] F. Filbet, E. Sonnendruecker, and P. Bertrand. Conservative Numerical Schemes for the Vlasov Equation. *Journal of Computational Physics*, 172:166–187, 2001.

[19] Francis Filbet, Clément Mouhot, and Lorenzo Pareschi. Solving the Boltzmann equation in N log N. *math/0607573*, July 2006. SIAM Journal on Scientific Computing 28 (2006) 1029-1053.

[20] I. Gamba and J. Proft. Stable discontinuous Galerkin schemes for linear Vlasov-Boltzmann transport equations. *To appear*, 2007.

[21] A. Gerisch and R. Weiner. The Positivity of Low-Order Explicit Runge-Kutta Schemes Applied in Splitting Methods. *Computers and Mathematics with Applications*, 45:53–67, 2003.

[22] T. Gjesdal. Implicit-explicit methods based on strong stability preserving multistep time discretizations. *Applied Numerical Mathematics*, 57:911–919, 2007.

[23] S. Gottlieb and C.-W. Shu. Total variation diminishing Runge-Kutta schemes. *Mathematics of Computation*, 67(221):73–85, 1998.

[24] B. Gustafsson, H.-O. Kreiss, and J. Oliger. *Time Dependent Problems and Difference Methods*. Pure and Applied Mathematics: A Wiley-Interscience Series of Texts, Monographs and Tracts, 1996.

[25] E. Hairer. Order Conditions for Numerical Methods for Partitioned Ordinary Differential Equations. *Numerische Mathematik*, 36:431–445, 1981.

[26] J.S. Hesthaven, S. Gottlieb, and D. Gottlieb. *Spectral Methods for Time-Dependent Problems*. Cambridge University Press, 2007.

[27] J.S. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*. Springer, 2007.

[28] E. Hofer. A Partially Implicit Method for Large Stiff Systems of ODE's with Only Few Equations Introducing Small Time-Constants. *SIAM Journal on Numerical Analysis*, 13(5):645–663, 1976.

[29] Ibragimov and Rjasanow. Numerical solution of the boltzmann equation on the uniform grid. *Computing*, 69:163–186, October 2002.

[30] R. Illner and H. Neunzert. On Simulation methods for the Boltzmann Equation. *Transport Theory and Statistical Physics*, 16:141–154, 1987.

[31] S. Jin. Runge-Kutta methods for hyperbolic systems with stiff relaxation terms. *Journal of Computational Physics*, 122:51–67, 1995.

[32] A. Klimas and W.M. Farrell. A splitting algorithm for Vlasov simulation with filamentation filtration. *Journal of Computational Physics*, 110:150–163, 1994.

[33] A.J. Klimas. A method for overcoming the velocity space filamentation problem in collisionless plasma model solutions. *Journal of Computational Physics*, 68:202–226, 1987.

[34] R.J. Leveque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.

[35] Clément Mouhot and Lorenzo Pareschi. Fast Algorithms for Computing the Boltzmann Collision Operator. *Math. Comp*, 75:1833–1852, 2006.

[36] T. Nakamura and T. Yabe. Cubic interpolated propagation scheme for solving the hyper-dimensional vlasov-poisson equation in phase space. *Computer Physics Communications*, 120:122–154, 1999.

[37] K. Nanbu. Direct Simulation Scheme derived from the Boltzmann Equation. I. monocomponent gases. *Journal of the Physical Society of Japan*, 52:2042–2049, 1983.

[38] Taku Ohwada. High Order Approximation Methods for the Boltzmann Equation. *Journal of Computational Physics*, 139:1–14, 1998.

[39] L. Pareschi and B. Perthame. A Fourier Spectral Method for homogeneous Boltzmann Equation. *Transport Theory and Statistical Physics*, 25(3):369–382, 1996.

[40] L. Pareschi and G. Russo. Implicit-explicit Runge-Kutta schemes for stiff systems of differential equations. *Advances In The Theory Of Computational Mathematics*, 3:269–288, 2000.

[41] L. Pareschi and G. Russo. Numerical Solution of the Boltzmann Equation I: Spectrally Accurate Approximation of the Collision Operator. *SIAM Journal of Numerical Analysis*, 37(4):1217–1245, 2000.

[42] L. Pareschi and G. Russo. Implicit-Explicit Runge-Kutta Schemes and Applications to Hyperbolic Systems with Relaxation. *Journal of Scientific Computing*, 25:129–155, 2005.

[43] Lorenzo Pareschi and Giovanni Russo. Time Relaxed Monte Carlo Methods for the Boltzmann Equation. *SIAM Journal on Scientific Computing*, 23:1253–1273, 2002.

[44] K. Sebastian and C.-W. Shu. Multidomain WENO Finite Difference Method with Interpolation at Subdomain Interfaces. *Journal of Scientific Computing*, 19:405–438, 2003.

[45] Xiaowen Shan and Xiaoyi He. Discretization of the Velocity Space in the Solution of the Boltzmann Equation. *Physical Review Letters*, 80:65, 1998.

[46] C.-W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, 1697:325–432, 1998.

[47] E. Sonnendruecker, J. Roche, P. Bertrand, and A. Ghizzo. The Semi-Lagrangian Method for the Numerical Resolution of the Vlasov Equation. *Journal of Computational Physics*, 149(2):201–220, 1999.

[48] G. Strang. On the construction and comparison of different schemes. *SIAM Journal of Numerical Analysis*, 5(3):506–517, 1968.

[49] Alexander J. Wagner. A Practical Introduction to the Lattice Boltzmann Method. http://www.physics.ndsu.nodak.edu/wagner/LB.html, March 2006.