HAL open science

Lagrangian/Eulerian solvers and simulations for Vlasov-Poisson

Sebastien Guisset, Philippe Helluy, Michel Massaro, Laurent Navoret, Nhung Pham, Malcolm Roberts

To cite this version:

Sebastien Guisset, Philippe Helluy, Michel Massaro, Laurent Navoret, Nhung Pham, et al.. Lagrangian/Eulerian solvers and simulations for Vlasov-Poisson. ESAIM: Proceedings and Surveys, 2016, 10.1051/proc/201653008 . hal-01239673

HAL Id: hal-01239673

https://hal.science/hal-01239673v1

Submitted on 11 Dec 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ESAIM: PROCEEDINGS AND SURVEYS, Vol. ?, 2015, 1-10
Editors: Will be set by the publisher

LAGRANGIAN/EULERIAN SOLVERS AND SIMULATIONS FOR VLASOV-POISSON

Sebastien Guisset[1,2], Philippe Helluy[3,4], Michel Massaro[4], Laurent Navoret[3,4], Nhung Pham[3,4] and Malcolm Roberts[3,4]

Abstract. We construct a hyperbolic approximation of the Vlasov equation using a method of reduction [10, 14, 22] in which the dependency on the velocity variable is removed. The reduction relies on a semi-discrete finite element approximation in the velocity variable. We apply Gauss-Lobatto numerical integration in velocity space, reducing the hyperbolic system to a system of transport equations for which the transport velocities are the Gauss-Lobatto points. The transport equations are coupled through a zero-order term that represents the electromagnetic forces. We solve the resulting system by a splitting approach: the homogeneous transport equations are solved by a split semi-Lagrangian method and the source term is applied independently. We also present preliminary comparisons with another transport solver based on the discontinuous Galerkin method.

Résumé. Au moyen d'une méthode de réduction décrite dans [10, 14, 22] nous construisons une approximation hyperbolique de l'équation de Vlasov dans laquelle la dépendance en vitesse est supprimée. La réduction repose sur une semi-discrétisation par éléments finis dans la variable de vitesse. Nous appliquons aussi une intégration numérique de Gauss-Lobatto dans l'espace des vitesses. Le sytème hyperbolique se réduit alors à un système d'équations de transport dont les vitesses sont les points de Gauss-Lobatto. Les équations de transport sont couplées à travers un terme source d'ordre zéro qui représente la force électromagnétique. Nous résolvons le système obtenu par une méthode de splitting: les équations de transport homogènes sont résolues par un algorithme semi-Lagrangien splitté et le terme source est appliqué indépendamment. Nous présentons également des comparaisons préliminaires avec un autre solveur de l'équation de transport basé sur une approche Galerkin discontinu.

Introduction

The Vlasov-Poisson system is a popular model for the numerical simulation in plasma physics. Solving the Vlasov-Poisson equation is challenging as it is composed of a time-dependent transport equation in a six-dimensional $(\mathbf{x}, \mathbf{v})$ phase-space coupled with the electric potential equation. Some popular methods for studying this equation are the particle-in-cell (PIC) method [3] or the semi-Lagrangian approach [7].

In a previous work [22] (see also [10,14]), we constructed a reduced Vlasov-Poisson system where the dependency in the velocity variable $\mathbf{v}$ is removed. The principle is to approximate the distribution function $f(\mathbf{x}, \mathbf{v}, t)$ in the velocity variable $\mathbf{v}$ with a finite element interpolation; this semi-discretization transforms the Vlasov equation into a hyperbolic system for which the unknowns system are the values of f at the interpolation nodes in $\mathbf{v}$. The hyperbolic system contains a zero order source term that represents the electric force.

Here, we apply the same strategy as in [22] but we replace the exact numerical integration by Gauss-Lobatto integration in the finite element approximation. This simplifies the nature of the hyperbolic system, reducing it to a system of transport equations for which the velocities are the Gauss-Lobatto points. The spatial transport and velocity source term are solved separately. The transport equation is solved by a semi-Lagrangian approach [6,7]. We

[1] Univ. Bordeaux, CELIA, UMR 5107, F- 33400 Talence, France.
[2] Univ. Bordeaux, IMB,UMR 5251, F- 33400 Talence, France; e-mail: guisset@celia.u-bordeaux1.fr
[3] INRIA Nancy - Grand Est / IRMA - TONUS,
[4] IRMA, UMR 7501, Univ. de Strasbourg et CNRS; e-mail: philippe.helluy@math.unistra.fr, laurent.navoret@math.unistra.fr, pham@math.unistra.fr, michel.massaro@math.unistra.fr, malcolm.i.w.roberts@gmail.com

present numerical results on the classic test cases of Landau damping and the two-stream instability and evaluate the conservation properties of the scheme. Finally, we provide preliminary comparisons between the semi-Lagrangian transport solver and a recently developed discontinuous Galerkin (DG) transport solver.

1. Model reduction of the Vlasov-Poisson equation

We consider the two-dimensional Vlasov equation

$$\frac{\partial f}{\partial t}(\mathbf{x},\mathbf{v},t) + \mathbf{v}\cdot\nabla_{\mathbf{x}} f(\mathbf{x},\mathbf{v},t) + \mathbf{E}\cdot\nabla_{\mathbf{v}} f(\mathbf{x},\mathbf{v},t) = \mathbf{0}. \tag{1}$$

The unknown is the distribution function f, which depends on the position $\mathbf{x}\in\Omega_x\subset\mathbb{R}^2$, the velocity variable $\mathbf{v}\in\Omega_v\subset\mathbb{R}^2$ and the time variable $t\in\mathbb{R}$. The electric field $\mathbf{E}$ depends on $\mathbf{x}$ and t and is given by

$$\mathbf{E} = -\nabla_{\mathbf{x}}\Phi, \text{ with } -\Delta_{\mathbf{x}}\Phi = \rho - \overline{\rho}. \tag{2}$$

The charge ρ and the mean value of the charge $\overline{\rho}$ are given by

$$\rho(\mathbf{x},t) = \int_{\mathbf{v}} f(\mathbf{x},\mathbf{v},t)d\mathbf{v}, \quad \overline{\rho}(t) = \frac{\int_{\mathbf{x},\mathbf{v}} f(\mathbf{x},\mathbf{v},t)d\mathbf{x}d\mathbf{v}}{|\Omega_x|}. \tag{3}$$

The computational domain is $\Omega = \Omega_x\times\Omega_v$ where $\Omega_v = (-V,V)^2$ for some fixed $V>0$. We assume periodic boundary conditions in the space variable. Let $\mathbf{n}_v = (n_v^1, n_v^2)$ be the outward normal vector on the velocity boundary $\partial\Omega_v$. Because the Vlasov equation is a transport equation, it is natural to apply upwind boundary conditions at the velocity boundary

$$\mathbf{E}(\mathbf{x},t)\cdot\mathbf{n}_v(\mathbf{v})|_{\mathbf{v}\in\partial\Omega_v} < 0 \Rightarrow f(\mathbf{x},\mathbf{v},t)|_{\mathbf{v}\in\partial\Omega_v} = 0. \tag{4}$$

Using the notation $(\mathbf{E}\cdot\mathbf{n}_v)^- = \min(\mathbf{E}\cdot\mathbf{n}_v, 0)$, condition (4) is equivalent to

$$(\mathbf{E}\cdot\mathbf{n}_v)^- f(\mathbf{x},\mathbf{v},t) = 0 \quad \text{on } \partial\Omega_v. \tag{5}$$

Note that the boundary condition is trivially satisfied when $\mathbf{E}\cdot\mathbf{n}_v \geq 0$. The equations (1)-(2) are supplemented by an initial condition

$$f(\mathbf{x},\mathbf{v},0) = f_0(\mathbf{x},\mathbf{v}).$$

We now recall how to obtain the reduction of the Vlasov equation. One first expands the distribution function f on a basis of functions depending on $\mathbf{v}$, $\{\varphi_j\}_{j=1,\dots,P}$.

$$f(\mathbf{x},\mathbf{v},t) \simeq \sum_{j=1}^{P} w_j(\mathbf{x},t)\varphi_j(v) = w_j(\mathbf{x},t)\varphi_j(\mathbf{v}). \tag{6}$$

(The convention of summation on repeated indices is used.) Several different bases can be used, see for instance [4] and references therein. Multiplying the Vlasov equation (1) by φ_i, integrating with respect to $\mathbf{v}$

$$\partial_t w_j \int_{\Omega_v} \varphi_i\varphi_j + \nabla_{\mathbf{x}} w_j \cdot \int_{\Omega_v} \mathbf{v}\varphi_i\varphi_j + \mathbf{E} w_j \int_{\Omega_v} \varphi_i \nabla_{\mathbf{v}}\varphi_j = 0. \tag{7}$$

Let $(\mathbf{E}\cdot\mathbf{n}_v)^+ = \max(\mathbf{E}\cdot\mathbf{n}_v, 0)$. Using Green's formula two times, and applying the boundary conditions (5), we have

$$\begin{aligned}
\mathbf{E}w_j\int_{\Omega_v}\varphi_i\nabla_{\mathbf{v}}\varphi_j &= -\mathbf{E}w_j\int_{\Omega_v}\nabla_{\mathbf{v}}\varphi_i\varphi_j + w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)\varphi_j\varphi_i && (8)\\
&= -\mathbf{E}w_j\int_{\Omega_v}\nabla_{\mathbf{v}}\varphi_i\varphi_j + w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)^+\varphi_j\varphi_i && (9)\\
&= \mathbf{E}w_j\int_{\Omega_v}\nabla_{\mathbf{v}}\varphi_i\varphi_j - w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)\varphi_j\varphi_i + w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)^+\varphi_j\varphi_i && (10)\\
&= \mathbf{E}w_j\int_{\Omega_v}\nabla_{\mathbf{v}}\varphi_i\varphi_j - w_j\int_{\partial\Omega_v}\big(\mathbf{E}\cdot\mathbf{n}_v - (\mathbf{E}\cdot\mathbf{n}_v)^+\big)\varphi_j\varphi_i && (11)\\
&= \mathbf{E}w_j\int_{\Omega_v}\varphi_i\nabla_{\mathbf{v}}\varphi_j - w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)^-\varphi_j\varphi_i && (12)
\end{aligned}$$

Finally

$$\partial_t w_j\int_{\Omega_v}\varphi_i\varphi_j + \nabla_{\mathbf{x}}w_j\cdot\int_{\Omega_v}\mathbf{v}\varphi_i\varphi_j + \mathbf{E}w_j\int_{\Omega_v}\varphi_i\nabla_{\mathbf{v}}\varphi_j - w_j\int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)^-\varphi_j\varphi_i = 0. \tag{13}$$

In this way, we obtain the following Friedrichs systems with sources:

$$M\partial_t\mathbf{w} + A^1\partial_{x_1}\mathbf{w} + A^2\partial_{x_2}\mathbf{w} + B(\mathbf{E})\mathbf{w} = \mathbf{0}. \tag{14}$$

where $\mathbf{w}$ is the vector of P components $\mathbf{w} = (w_1, w_2, ..., w_P)^T$ and M, A^1, A^2, $B(\mathbf{E})$ are matrices of dimension $P\times P$, whose elements are given by

$$M_{i,j} = \int_{\Omega_v}\varphi_i\varphi_j, \quad A^1_{i,j} = \int_{\Omega_v}v^1\varphi_i\varphi_j, \quad A^2_{i,j} = \int_{\Omega_v}v^2\varphi_i\varphi_j, \tag{15}$$

and

$$B(\mathbf{E})_{i,j} = \mathbf{E}\cdot\int_{\Omega_v}\varphi_i\nabla_{\mathbf{v}}\varphi_j - \int_{\partial\Omega_v}(\mathbf{E}\cdot\mathbf{n}_v)^-\varphi_j\varphi_i. \tag{16}$$

The above procedure applies to any choice of velocity basis. As in [14, 22], we use a finite element interpolation basis, for other choices we refer to eg [4]. To construct the function basis, we first consider a regular square mesh of Ω_v made of finite elements of degree d. The basis functions are continuous on $\overline{\Omega}_v$ and polynomial of degree d in each square finite element. Gauss-Lobatto integration points are used in each element to discretize the variation of f with $\mathbf{v}$. Let N_v be the number of elements in each velocity direction, so that the number of Gauss-Lobatto points in the mesh is $P = (N_v d+1)^2$. We denote these points $\{N_i\}_{i=1}^P$. The basis functions satisfy the interpolation property

$$\varphi_j(N_i) = \delta_{ij},$$

where δ_{ij} is the Kronecker delta. From the finite element construction, we also have access to the velocity mesh connectivity: for a given finite element index k, $0\le k\le N_v^2$ and a local Gauss-Lobatto point index ℓ, $0\le\ell\le(d+1)^2$, we are able to recover the global index i, $0\le i\le P$, of the Gauss-Lobatto point. In this case, we also use the notation $N_i = N_{k,\ell}$. For more details we refer to [22], where the construction of the finite element basis is fully detailed in the one-dimensional case.

A given function h defined on Ω_v can be integrated by splitting $\int_{\Omega_v}h(\mathbf{v})d\mathbf{v}$ into elementary integrals on the square finite elements Q_k, $k = 1\dots N_v^2$,

$$\int_{\Omega_v}h(\mathbf{v})d\mathbf{v} = \sum_{k=1}^{N_v^2}\int_{Q_k}h(\mathbf{v})d\mathbf{v}.$$

Then by using the Gauss-Lobatto quadrature, we obtain

$$\int_{Q_k}h(\mathbf{v})d\mathbf{v} \simeq \sum_{\ell=1}^{(d+1)^2}\omega_{k,\ell}h(N_{k,\ell}), \tag{17}$$

where $\omega_{k,\ell}$ is the weight associated to Gauss-Lobatto point $N_{k,\ell}$ in the quadrature formula. Finally, applying formula (17) we obtain that the matrices M and A^k are diagonal and given by

$$M_{i,i} = \sum_{N_i = N_{k,\ell}} \omega_{k,\ell} > 0, \tag{18}$$

and

$$A^k_{i,i} = M_{i,i} V^k_i, \quad \mathbf{V}_i = (V^1_i, V^2_i). \tag{19}$$

The matrix $B(\mathbf{E})$ is sparse but not diagonal; with the standard numbering of the finite element interpolation points its bandwidth is $N_v d + 1$. We can also rewrite the system (14) as

$$\partial_t \mathbf{w} + M^{-1} A^1 \partial_{x_1} \mathbf{w} + M^{-1} A^2 \partial_{x_2} \mathbf{w} + M^{-1} B(E) \mathbf{w} = \mathbf{0}. \tag{20}$$

where $M^{-1}A^1$ and $M^{-1}A^2$ are diagonal. We call equation (20) the reduced Vlasov model. We observe that thanks to the choice of the basis functions and the numerical quadrature, the reduced Vlasov model is simply a system of transport equations that are coupled through the source term $M^{-1}B(\mathbf{E})\mathbf{w}$.

2. Numerical methods for solving the reduced Vlasov model

In this section, we present three methods to solve the reduced Vlasov model on Cartesian meshes. The finite volume method (subsection 2.1) and the semi-Lagrangian method (subsection 2.2) rely on a directional-splitting method to replace the $2D$ problem by a pair of one-dimensional problems. Let $\mathbf{w}^n$ be an approximation of $\mathbf{w}$ at time t_n, the time discretization is

$$\frac{\mathbf{w}^* - \mathbf{w}^n}{\Delta t} + M^{-1} A^1 \, \partial_{x_1} \mathbf{w}^n = 0, \tag{21}$$

$$\frac{\mathbf{w}^{**} - \mathbf{w}^*}{\Delta t} + M^{-1} A^2 \, \partial_{x_2} \mathbf{w}^* = 0, \tag{22}$$

$$\frac{\mathbf{w}^{n+1} - \mathbf{w}^{**}}{\Delta t} + M^{-1} B(\mathbf{E}) \mathbf{w}^{**} = 0. \tag{23}$$

where $\mathbf{w}^*$ and $\mathbf{w}^{**}$ are intermediate unknowns. The discontinuous Galerkin method (subsection 2.3) can be applied without having to apply the above splitting technique.

2.1. The finite volume method

A 1D spatial domain of length L is split into N_x cells; the cell C_i is the interval $\left(x_{i-1/2}, x_{i+1/2}\right)$, $i = 1 \ldots N_x$, where $x_{i-1/2} = (i - 1/2)\Delta x$ and $\Delta x = L/N_x$. We consider a piece-wise constant approximation of the vector $\mathbf{w}$ on the spatial mesh

$$\mathbf{w}^n_i \simeq \mathbf{w}(x, t^n), \quad x \in C_i.$$

We obtain the following numerical scheme

$$\frac{\mathbf{w}^{n+1}_i - \mathbf{w}^n_i}{\Delta t} = -\frac{\mathbf{F}(\mathbf{w}_i, \mathbf{w}_{i+1}) - \mathbf{F}(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\Delta x}. \tag{24}$$

Let $\mathbf{F}$ denote the slightly upwinded numerical flux

$$\mathbf{F}(\mathbf{w}_a, \mathbf{w}_b) = M^{-1} A_1 \frac{\mathbf{w}_a + \mathbf{w}_b}{2} - \varepsilon(\mathbf{w}_b - \mathbf{w}_a),$$

where ε is the numerical diffusion coefficient. This numerical scheme is accurate to $\mathcal{O}(\Delta x)$ when the solution is sufficiently smooth. Note that the time step is constrained by the CFL condition

$$\Delta t = \alpha \Delta x / V, \quad 0 < \alpha \leq 1.$$

For more details we refer to [22].

The method is implemented using the library `SELALIB`[1]. The code is parallelized with MPI using a domain decomposition algorithm and the Poisson equation is solved using a FFT.

[1] `selalib.gforge.inria.fr`

2.2. The semi-Lagrangian method

We use the same spatial mesh as in subsection 2.1. Since A^1 is diagonal, system (21) is a time-discretization of P one-dimensional transport equations with constant velocities V_i which evolves according to

$$\partial_t \mathbf{w} + V_i\, \partial_x \mathbf{w} = \mathbf{0}. \tag{25}$$

The semi-Lagrangian scheme is based on the method of characteristics. In our case, the characteristics are straight lines, so

$$\mathbf{w}(t^{n+1}, x) = \mathbf{w}(t^n, x - V_{1,i}\,\Delta t). \tag{26}$$

Then we approximate $\mathbf{w}_i^{n+1}$ as $\mathbf{w}(t^{n+1}, x_i)$ using the equation

$$\mathbf{w}_i^{n+1} = [\Pi \mathbf{w}^n]\,\big(x_i - V_{1,i}\,\Delta t\big), \tag{27}$$

where $\Pi\mathbf{w}^n$ is an interpolation function built from the points $(x_i, \mathbf{w}_i^n)$. Several interpolation methods have been studied (see eg [6]). We perform an interpolation π using either a cubic spline (with accuracy $\mathcal{O}(\Delta x^3)$) or a Lagrange polynomial with $2r+1$ points (with accuracy $\mathcal{O}(\Delta x^{2r+1})$) which we denote

$$[\Pi \mathbf{w}]_{|[x_i, x_{i+1}]} = \pi\big(\big(x_j, \mathbf{w}_j\big)_{i-r \le j \le i+r}\big).$$

We refer to [7, 17] for more details. One of the main advantage of the semi-Lagrangian method is that the size of the time step is not bounded above by a stability condition. Consequently, the time-step size is only limited by the required precision and the stability condition of the finite-element method used in $\mathbf{v}$.

The Strang splitting can be modified in order to obtain second order accuracy in time [7, 17].

This method is implemented using the semi-Lagrangian library `SELALIB`. The stencil of the semi-Lagrangian algorithm is enlarged compared to the finite volume algorithm and it is therefore not possible to adopt a subdomain decomposition. Instead we use the MPI based grid transposition algorithm available in `SELALIB` (based on `MPI_Alltoall` transportation) in order to parallelize the method.

2.3. The discontinuous Galerkin (DG) method for the reduced Vlasov model

The discontinuous Galerkin (DG) method is a generalization of the finite volume method for solving systems of type (20). The computational domain Ω_x is covered with a mesh; this mesh need not be structured or conforming, nevertheless in practice we choose a structured and conforming hexahedron mesh because the geometry of Ω_x is very simple. In general, each cell L of the mesh is obtained from a second order polynomial transformation that maps the reference cube $\hat{L}$ onto L. The cells of our mesh can thus be "curved" hexahedrons, even if this feature is not exploited in this work. In each cell L of the mesh, the field is approximated by polynomial basis functions

$$\mathbf{w}(\mathbf{x}, t) = \mathbf{w}_L^j(t)\psi_j^L(\mathbf{x}), \quad \mathbf{x} \in L. \tag{28}$$

The numerical solution satisfies the DG approximation scheme

$$\int_L \partial_t \mathbf{w}\psi_i^L - \int_L \mathbf{F}\big(\mathbf{w}, \mathbf{w}, \nabla\psi_i^L\big) + \int_{\partial L} \mathbf{F}(\mathbf{w}_L, \mathbf{w}_R, \mathbf{n}_{LR})\psi_i^L = 0 \tag{29}$$

where R denotes cells adjacent to L, $\mathbf{n}_{LR}$ is the unit normal vector on ∂L oriented from L to R, and $\mathbf{F}(\mathbf{w}_L, \mathbf{w}_R, \mathbf{n})$ is a standard upwind numerical flux satisfying

$$\mathbf{F}(\mathbf{w}, \mathbf{w}, \mathbf{n}) = A^k n^k \mathbf{w}. \tag{30}$$

We apply Gauss-Lobatto numerical integration in $\mathbf{x}$ using nodal basis functions $\psi_i^L(G_k^L) = \delta_{i,k}$. The Gauss-Lobatto integration points and associated weights are noted G_k^L and ω_k^L and the quadrature formula is $\int_L h(X)dX \simeq \sum_k \omega_k^L h(G_k^L)$. The Gauss-Lobatto points are first defined on the reference cube by tensor products of one-dimensional points. They are then mapped to L by the geometric transformation. Similarly, the nodal basis function ψ_i^L are obtained from tensor products of one-dimensional Lagrange polynomials and mapped from $\hat{L}$ to L. In the end, we perform time integration of a system of ordinary differential equations. For a similar approach (but with Gauss-Legendre points instead of Gauss-Lobatto points), we refer to the PhD of Thomas Strub [18]. See also [19].

We have implemented the DG algorithm in the `OpenCL` framework, which is a software environment for programming GPUs or multicore accelerators. Unlike `CUDA`, which is only available for NVIDIA GPUs, `OpenCL` allows access to multicore accelerators of many brands. We will not describe here the philosophy of `OpenCL` and the full details of the DG implementation, but refer readers to previous works [20, 24]. We have also implemented an `OpenMP` version of the same algorithm for use when `OpenCL` is not available and for verification and comparisons. The resulting software `schnaps` can be found at `schnaps.gforge.inria.fr`.

In contrast to our previous works we have also provided additional features, such as a macrocell strategy for managing the DG mesh: the connectivity of the mesh is described at a macro level with H20 quadratic hexahedrons[2] with control points on each vertex and in the middle of each edge, allowing for curved elements in physical space via a non-linear transformation. Then, each macrocell is split into subcells that share the same macro geometry data. In the implementation of the DG algorithm, a macrocell is associated to a single `OpenCL` kernel, subcells are associated to `OpenCL` work-groups, and finally the Gauss points are associated to `OpenCL` work-items. This organization reduces memory access to the mesh connectivity. The development of `schnaps` is still in progress. In this paper, we shall only present preliminary results.

3. Numerical results

3.1. Convergence rate

We first verify that the convergence orders of transport solvers corresponds to theory. Equation (14) is solved with an electric field equal to zero leading a vanishing source term $B(\mathbf{E})w = 0$. We compare the exact and the approximate solution by translating the initial conditions using the equation

$$f(\mathbf{x}, \mathbf{v}, t) = f(\mathbf{x} - t\mathbf{v}, \mathbf{v}, 0). \tag{31}$$

It is also possible to perform the same kind of test in the velocity variable $\mathbf{v}$. In this case, we assume a constant electric field and we cancel the transport in the $\mathbf{x}$ direction, which is equivalent to setting $A^1 = A^2 = 0$. The exact solution is then given by

$$f(\mathbf{x}, \mathbf{v}, t) = f(\mathbf{x}, \mathbf{v} - t\mathbf{E}, 0). \tag{32}$$

3.1.1. *SELALIB SL solver*

The semi-Lagrangian / finite-element element solver was tested for both spatial and velocity translation. In both cases the computational domain was $\Omega = \Omega_x \times \Omega_v = (-1, 1)^2$. For the spatial case, the initial condition was

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \sin(\pi x). \tag{33}$$

We used a second-order time-stepper with $\Delta t = 0.01$ and advanced the system to $t = 0.5$ and $N_v = 17$ grid points in v_1. The spatial convergence agreed well with the theoretical convergence rate for a cubic spline interpolation. Results are shown in Figure 1.

The initial condition for the velocity transport case was

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \begin{cases} (1-2v)^3(1+2v)^3 & \text{if } v < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{34}$$

where $v = \sqrt{v_1^2 + v_2^2}$. The electrical field was set to $\mathbf{E} = \mathbf{v}_1$. We used a second-order time-stepper with $\Delta t = 0.01$ and advanced the system to $t = 0.2$. We observe a convergence close to the theoretical value of 2 for second order finite elements. Results are shown in Figure 1.

3.1.2. *schnaps DG solver*

The initial condition used for this test was

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{v^2}{\sigma}} \begin{cases} e^{\frac{-1}{1-4r^2}} & \text{if } r < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{35}$$

[2]The "Hexahedron20" type in gmsh: see `geuz.org/gmsh/doc/texinfo/gmsh.html#Node-ordering`.

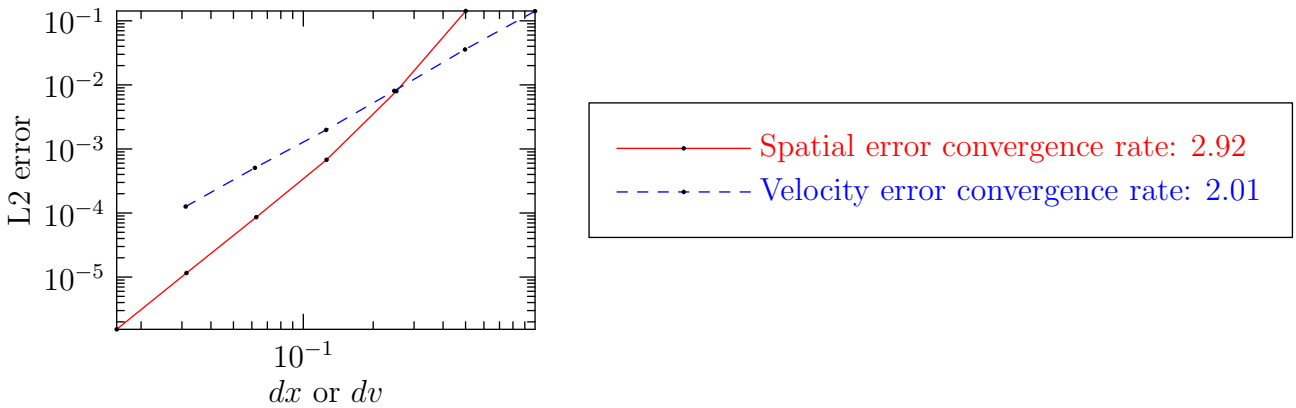

FIGURE 1. The relative L^2 error between the numerical solution and the analytic solution as a function of grid spacing for `SELALIB`. An interpolation spline of order 3 is used in space and a second order finite element method is used for the velocity transport.

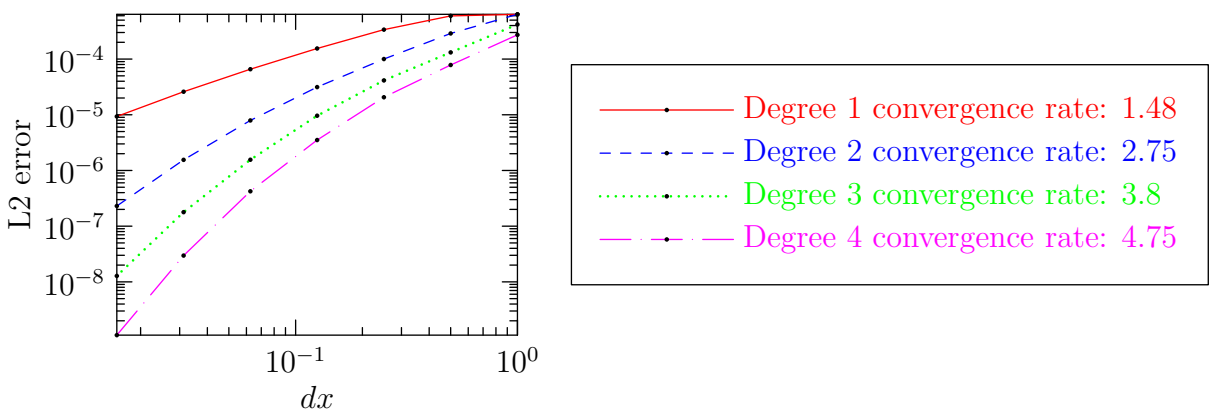

FIGURE 2. The relative L^2 error between the numerical solution and the analytic solution as a function of grid spacing for `schnaps` using Gauss-Lobatto interpolation with polynomial degree between 1 and 4.

where $r = \sqrt{x_1^2 + x_2^2}$, $v = \sqrt{v_1^2 + v_1^2}$, and $\sigma = 0.2$. This initial condition consists of a Gaussian in the velocity multiplied by a C^∞ function with support in $B_{\mathbf{x}}(\mathbf{0}, 1/2)$. The tests were done with 30 velocity components in direction $\mathbf{v}_1$ and $\mathbf{v}_2$, the computational domain was $\Omega = \Omega_v \times \Omega_x = (-1,1)^4$, and the maximum time was $t = 0.4$. For each combination of degree and spatial resolution the fourth-order Runge-Kutta time step was decreased by a factor of two until subsequent error values differed by no more than 1%, thereby guaranteeing that the error was independent of the choice of time-step. The results are shown in Figure 2 and we observe that the convergence rate in the L^2 norm is slightly larger than to the polynomial order d. This results again complies with the theory. Let us observe that if we had used Gauss-Legendre integration, we would have reached a convergence rate of $d+1$. Gauss-Lobatto points were chosen so as to allow better performance with respect to memory access and to reduce the implementation cost.

3.2. **Performance of the SL and DG transport solvers**

In this section we compare the performance of the semi-Lagrangian and the DG transport solvers. We consider only the transport in $\mathbf{x}$, ie the source term in (23) is deactivated. At first sight, the semi-Lagrangian method should have a clear advantage; the scheme is universally stable and there is no error due to time discretization. However, memory access is more complicated for the semi-Lagrangian method than for the DG algorithm, and the parallelization of the semi-Lagrangian method relies on the transposition of the computational grid (`MPI_Alltoall`). We compare the speed of the two codes using parameters which could be considered realistic for research simulations. Velocity was discretized with 32 points in both dimensions (30 for `schnaps` due to a current technical limitation) and between 32 and 256 points in both spatial directions. The initial condition was a Gaussian in the velocity multiplied by a 6th degree C^2 piecewise defined polynomial with compact support in $B_{\mathbf{x}}(\mathbf{0}, 1/2)$. The particular

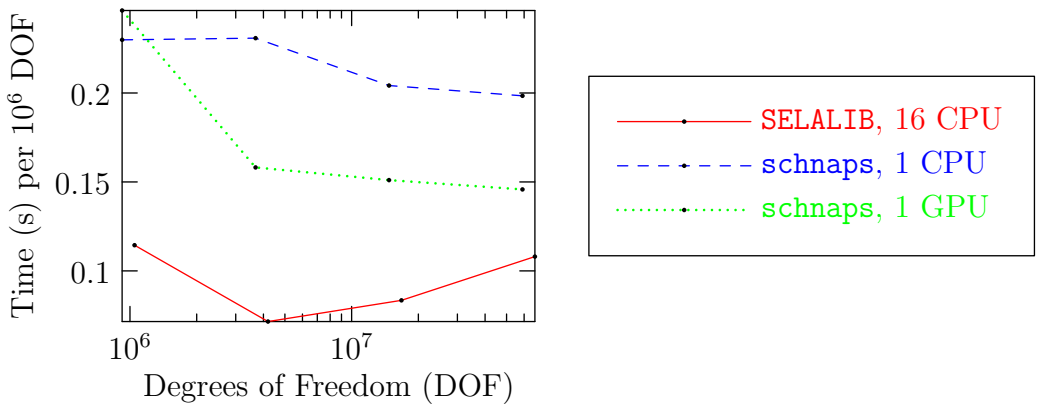

FIGURE 3. Computation time per second-order time-step per 10^6 degrees of freedom for `SELALIB` with 16 CPUs and `schnaps` with 1 CPU or 1 graphics card.

form of the initial condition is

$$f(\mathbf{x},\mathbf{v},t)|_{t=0} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{v^2}{\sigma}} \begin{cases} \frac{35}{16}(1-2r)^3(1+2r)^3 & \text{if } r < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

where $r = \sqrt{x_1^2 + x_2^2}$, $v = \sqrt{v_1^2 + v_2^2}$, and $\sigma = 0.2$. Since the initial condition is only C^2, we chose a degree-3 DG method in order to attain the expected convergence rate.

The computational domain for this test is $\Omega = \Omega_x \times \Omega_v = (-1,1)^4$. We evolved the system until time $t = 0.4$ to be consistent with both the homogeneous Dirichlet boundary conditions imposed in `schnaps` and the periodic boundary conditions imposed in `SELALIB`. The comparison of computational time per second-order time-step per million degrees of freedom is shown in Figure 3. The three implementations, namely `SELALIB`, `schnaps-C`, and `schnaps-OpenCL`, demonstrate roughly similar computation time per second-order time-step for the case studied. Since the time-step of the DG code is bounded by a stability condition, a semi-Lagrangian method may be a better choice if it can be shown that the error due to time discretization using a large value of dt does not significantly alter the results.

3.3. Comparison of the SL and FV methods

In this section we benchmark the spatial semi-Lagrangian / velocity finite-element method with an upwind finite-volume method developed previously [14, 22]. We consider the full Vlasov system as given in equation (20).

3.3.1. *Precision*

We used $N_{x_1} \times N_{v_1} = 128 \times 129$ grid points in the $\mathbf{x}_1$ and $\mathbf{v}_1$ directions and set $\Delta t = 7 \times 10^{-3}$ for the precision tests below. A Lagrange interpolation of degree 3 was used in space for the SL/FE method, and the FV method was first order in space. A degree 2 finite element method was used in velocity for both the SL/FE and FV methods.

In order to compare the precision of the two methods, we consider the 1D Landau damping test case. The initial distribution function is given by

$$f_0(\mathbf{x},\mathbf{v}) = (1 + \epsilon\cos(kx_1))\frac{1}{\sqrt{2\pi}} e^{-\frac{v_1^2}{2}}, \tag{37}$$

where $k = 0.5$ and $\epsilon = 5 \times 10^{-3}$. The spatial domain is periodic with length $L = 2\pi/k$. An analytic solution for the linearized problem is given in [16]. We plot the electric energy

$$\mathcal{E}(t) = \sqrt{\int_0^L (E_1(x_1,t))^2\, dx_1}$$

of the analytic solution, of the classic finite volume method, and of the semi-Lagrangian method on Figure 4. Both numerical methods capture the theoretical damping rate well. We observe that the solution obtained by the semi-Lagrangian method is closer to the analytic solution than the solution of the classic finite volume method. This

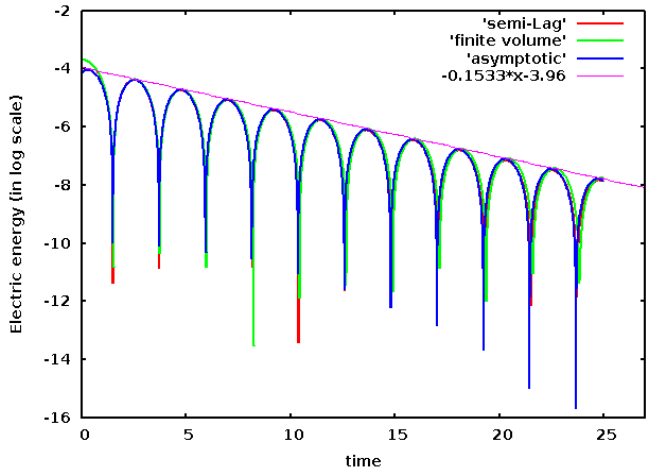

FIGURE 4. The electric energy of the Landau damping 1D test as a function of time. The blue curve labelled "asymptotic" is the exact solution of the linearized equation, the red curve is from the semi-Lagrangian/finite element method and the green curve is from the finite volume method.

is due to the fact that semi-Lagrangian method (order 3 in space) is more precise than the finite volume method (order 1 in space).

We now consider the two-stream instability one-dimensional test case in order to better compare the difference in the distribution function. In this test case, the initial distribution function is

$$f_0(\mathbf{x},\mathbf{v}) = (1+\epsilon\cos(kx_1))\frac{1}{2\sqrt{2\pi}}\left(e^{-\frac{(v_1-v_0)^2}{2}} + e^{-\frac{(v_1+v_0)^2}{2}}\right), \tag{38}$$

where $k = 0.2$, $\epsilon = 5\times 10^{-3}$ and $v_0 = 3$. The distribution function at time $t = 25$ and $t = 50$ for the two methods is compared in Figure 5. We observe that both methods do a good job at capturing the filamentation in phase-space, with the semi-Lagrangian / finite-element method being slightly more precise. The results are quite similar since they both use the same discretization in velocity.

3.3.2. *Stability*

The time-step of semi-Lagrangian / finite element method is only constrained by a stability condition in the **v**-direction, since the semi-Lagrangian method in the **x**-direction does not restrict the time-step. In order to compare the stability of the SL and the FV methods, we can fix the number of cells and then vary the time step. We consider a 4D Landau damping test case. The initial condition is

$$f_0(\mathbf{x},\mathbf{v}) = \frac{1}{2\pi}(1+\epsilon\cos(k_1x_1)\cos(k_2x_2))e^{-\frac{(v_1^2+v_2^2)}{2}}, \tag{39}$$

where $\epsilon = 5\times 10^{-3}$, the wave numbers $k_1 = k_2 = 0.5$ and $L_1 = L_2 = 4\pi$. The theoretical damping rate of the energy field is 0.394 [8]. We fix the number of cells at 32×32 in space and the degree 2 with 32 elements in each dimension of velocity. With the finite volume method, we have to use the time step small enough to ensure the stability of the energy field, for example here Δt is 1.47×10^{-2}. For this test-case, the electric field is very weak, so the stability condition in the **v**-direction does not significantly restrict the time step. Large time step can thus be used for the semi-Lagrangian / finite element method and this produces qualitatively reasonable results even with $\Delta t = 0.1$.

3.3.3. *Conservation*

The Vlasov equation conserves the L^1 and L^2 norms of the distribution function f. To test the conservation properties of the semi-Lagrangian/finite element and the finite-volume methods, we consider the strong Landau damping 1D test case with initial condition given by (37) but with the parameter $\epsilon = 0.5$ and $k_x = 0.5$. The number of grid points in x_1 is 32 and the number of grid points in v_1 is 65. We take the time step $\Delta t = 0.125$ for

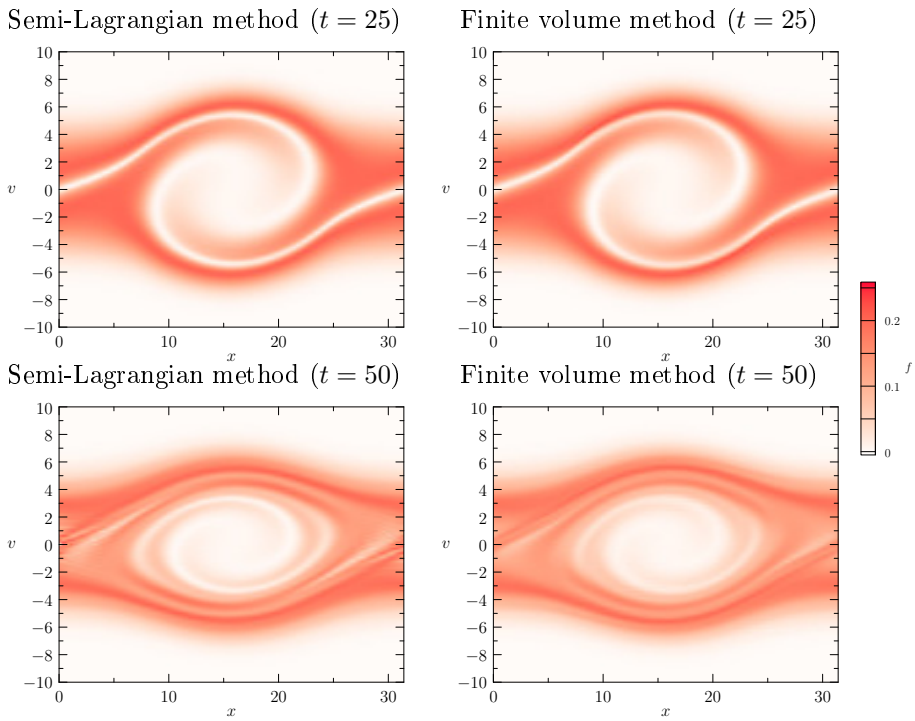

FIGURE 5. The distribution function $f(\mathbf{x}, \mathbf{v}, t)$ for the double-stream instability test-case at time $t = 25$ (on the top) and at $t = 50$ (in the bottom) in the (x_1, v_1) phase space. Results from the semi-Lagrangian method (left) and the classic finite volume method (right with slightly upwinded flux $\varepsilon = 0.008$) are shown.

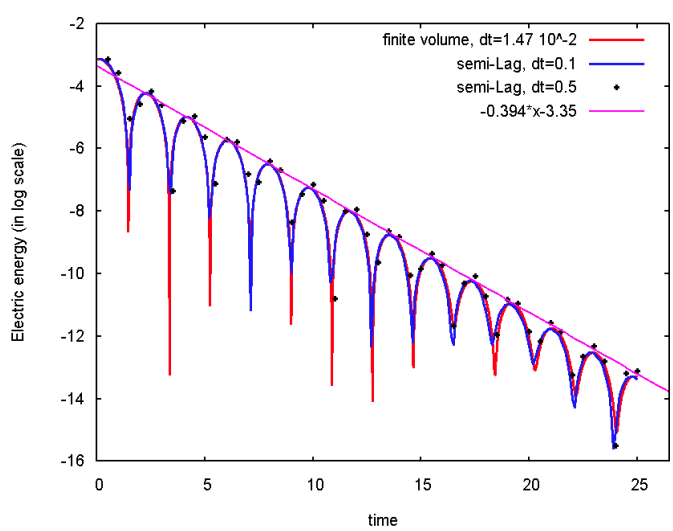

FIGURE 6. The electric energy of the Landau damping 2D test case as a function of time. The red curve is computed using the classic finite volume method with $\Delta t = 1.47 \times 10^{-2}$, the blue curve is computed with the semi-Lagrangian/finite element method with $\Delta t = 0.1$, and the black points are computed with computed with the semi-Lagrangian/finite element method with $\Delta t = 0.5$.

the semi-Lagrangian method and $\Delta t = 0.025$ for the finite volume method. The mass is defined as

$$\int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} f(\mathbf{x}, \mathbf{v}, t)\, d\mathbf{x}\, d\mathbf{v} \tag{40}$$

and its time evolution is shown in Figure 7. Due to the boundary condition in velocity, global mass is a priori not a conserved quantity, though the initial density quickly decays to zero as v increases, so the mass loss should not be

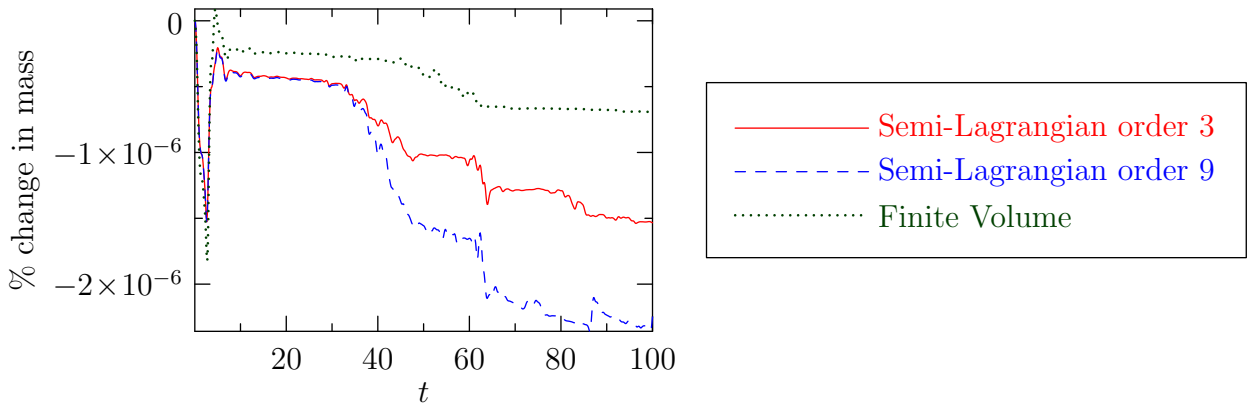

FIGURE 7. Time evolution of the change in mass as function of time for the strong Landau damping 1D test case. The semi-Lagrangian schemes (order 3: cubic spline interpolation, order 9: Lagrange interpolation) and the finite volume scheme ($\varepsilon = 0.02$) are compared.

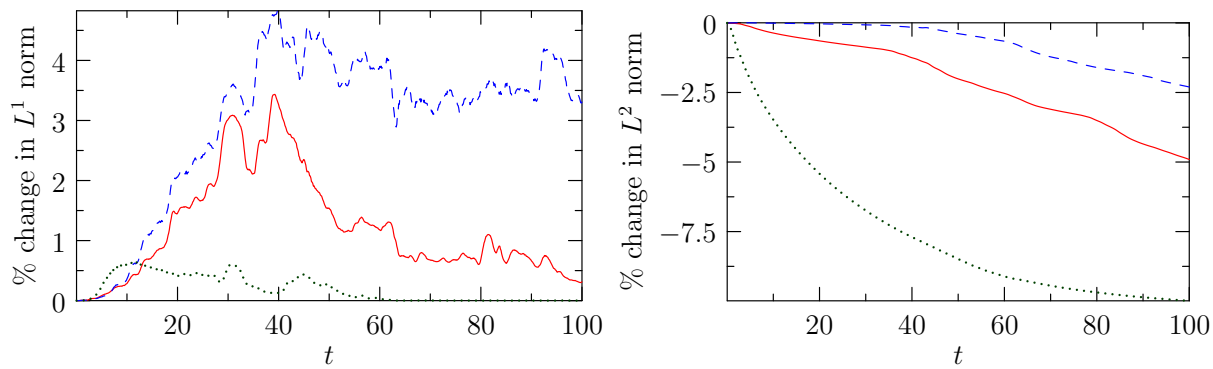

FIGURE 8. Time evolution of the change of the L^1 (left) and L^2 (right) norms of f as a function of time for the strong Landau damping 1D test case. Results from the semi-Lagrangian/finite element method are shown using interpolation using cubic spline of order 3 (red, solid) and Lagrange interpolation of order 9 (blue, dashed), and the finite volume method with $\varepsilon = 0.02$ (green, dotted).

very large. For this test-case, the three method (semi-Lagrangian cubic spline, semi-Lagrangian Lagrange order 9, finite volume) leads to conservation of mass up to a relative error of $\mathcal{O}(10^{-8})$. The time evolution of the L^1 and L^2 norms of f are shown in Figure 8. The finite volume method has better conservation of the L^1 norm for the cases studied here. Indeed, as the finite volume scheme is more diffusive (already observed in section 3.3.1), it better preserves the positivity of the distribution function. The L^2 norm is better conserved by the semi-Lagrangian/finite element scheme than by the finite volume scheme. The large numerical dissipation of the finite volume scheme leads to a relative error of $\mathcal{O}(0.1)$ in the L^2 norm. As observed in [7], the Lagrange interpolation of order 9 should be used in order to obtain numerical dissipation lower than the cubic spline interpolation.

4. Conclusion and Future Work

4.1. Conclusion

In this paper, we presented and compare several high order methods to efficiently solve the reduced Vlasov model (20) in space.

The semi-Lagrangian / finite element (SL/FE) `SELALIB` implementation was compared with the discontinuous Galerkin (DG) method implemented in `schnaps` using a test case which consisted of transport of a multi-valued velocity field. The SL/FE and DG method showed the appropriate convergence rates. The speeds of the two implementations were compared. `schnaps` had better performance per CPU, making it a priori faster, but `SELALIB` has less stringent stability requirements on the time step, which will allow it to simulations faster for certain configurations. The SL/FE method was compared with a finite volume method using the 1D double-stream instability

test-case and the 2D Landau damping test case for the Vlasov-Poisson equation. This allowed us to successfully validate the Vlasov-Poisson implementation in `SELALIB`. The SL/FE method was able to use much larger time-steps than the finite volume method, making the SL/FE method more efficient for the test cases considered.

The `schnaps` DG solver, which we present at an early stage of development, has more stringent time-step stability constraints, but is able to accommodate more complex geometry than a standard SL/FE method. `schnaps` also makes use of the `OpenCL` programming language to make use of `GPU`s and co-processor boards, which may allow for a higher degree of parallelization.

Complex geometries and unstructured, high order meshes can be more easily handled with the DG method.

4.2. **Future Work**

The SL/FE method, implemented in the `SELALIB` library, is part of an established software project. Future work for the SL/FE method includes implementation in 3D and the addition of a gyrokinetic (eg drift kinetic) model in order to capture complex flow regimes while mitigating the effects of the curse of dimensionality.

The DG method, implemented in `schnaps`, is in an earlier stage of development. In it current state, it can be used to solve general hyperbolic conservation equations in two and three dimensions, and is parallelized with `OpenMP` and `OpenCL`. However, the parallelization is optimal when the number of conserved quantities (eg the number of velocities) is small, and we expect to be able to achieve much better performance by modifying the parallelism and improving the coalescence of memory access when dealing with the Vlasov equations with a large number of degrees of freedom in velocity. We plan to implement the velocity source term using either FFTs, finite elements, or perhaps the semi-Lagrangian grid.

The Vlasov equation is a starting-point for more complicated systems which better represent the physical situation which we wish to understand. For example, particle collision models would be a worthwhile additions to both `SELALIB` and `schnaps`, as well as adding physically realistic geometries and boundary conditions when possible.

Acknowledgements

We would like to thank Michel Mehrenberger and Pierre Navaro of the Université de Strasbourg for their help with the `SELALIB` codebase. This project was also supported by the LABEX of the Université de Strasbourg.

References

[1] C. Altmann, T. Belat, M. Gutnic, P. Helluy, H. Mathis, E. Sonnendrücker, W. Angulo, J.-M. Hérard. A local time-stepping discontinuous Galerkin algorithm for the MHD system. CEMRACS 2008—Modelling and numerical simulation of complex fluids, 33–54, ESAIM Proc., 28, EDP Sci., Les Ulis, 2009.

[2] T. Barth. On discontinuous Galerkin approximations of Boltzmann moment systems with Levermore closure. Comput. Methods Appl. Mech. Engrg. 195 (2006), no. 25-28, 3311–3330.

[3] C. K. Birdsall, A. B. Langdon. Plasma Physics via Computer Simulation. Institute of Physics (IOP), Series in Plasma Physics, 1991.

[4] S. Le Bourdiec, F. de Vuyst, L. Jacquet. Numerical solution of the Vlasov-Poisson system using generalized Hermite functions. Comput. Phys. Comm. 175 (2006), no. 8, 528–544.

[5] A. Crestetto. Optimisation de méthodes numériques pour la physique des plasmas. Application aux faisceaux de particules chargées, October 2012.

[6] N. Crouseilles, M. Mehrenberger, E. Sonnendrücker. Conservative semi-Lagrangian schemes for Vlasov equations. J. Comput. Phys. 229 (2010), no. 6, 1927–1953.

[7] F. Filbet, E. Sonnendrücker. Comparison of Eulerian Vlasov solvers Comput. Phys. Commun., 150 (2003), pp. 247–266.

[8] F. Filbet, E. Sonnendrücker, P. Bertrand, Conservative Numerical Schemes for the Vlasov Equation, J. Comp. Phys., 172 (2001), pp. 166–187.

[9] E. Godlewski, P.-A. Raviart. Numerical approximation of hyperbolic systems of conservation laws. Applied Mathematical Sciences, 118. Springer-Verlag, New York, 1996.

[10] Reduced Vlasov-Maxwell simulations P Helluy, L Navoret, N Pham, A Crestetto Comptes Rendus Mécanique 342 (10-11), 619-635

[11] C. Johnson, J. Pitkäranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. Math. Comp. 46 (1986), no. 173, 1–26.

[12] G. Latu, N. Crouseilles, V. Grandgirard, E. Sonnendrücker. Gyrokinetic Semi-Lagrangian Parallel Simulation using a Hybrid OpenMP/MPI Programming, Recent Advances in PVM an MPI, Springer, LNCS, pp 356-364, Vol. 4757, (2007).

[13] B. Perthame. Boltzmann type schemes for gas dynamics and the entropy property. SIAM J. Numer. Anal. 27 (1990), no. 6, 1405–1421.

[14] N Pham, P Helluy, L Navoret. Hyperbolic approximation of the Fourier transformed Vlasov equation ESAIM: Proc., Congrés SMAI, Seignosse, 27-31 mai 2013, 45, 379-389

[15] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery: Numerical Recipes in C.

[16] E. Sonnendrücker. Approximation numerique des equations de Vlasov-Maxwell, Notes du cours de M2, 18 mars 2010.

[17] E. Sonnendrücker, J. Roche, P. Bertrand, A. Ghizzo. The semi-Lagrangian method for the numerical resolution of the Vlasov equation. J. Comp. Phys., Vol. 149, no. 2, pp. 201–220 (1999).

[18] Thomas Strub. Résolution numérique des équations de Maxwell tridimensionnelles instationnaires sur architecture massivement multicoeur. PhD thesis, University of Strasbourg, 2015.

[19] Thomas Strub, Philippe Helluy. Multi-gpu numerical simulation of electromagnetic waves. ESAIM: Proceedings and Surveys, 45:199–208, 2014.

[20] Thomas Strub, Nathanaël Muot, Philippe Helluy. Méthode Galerkin discontinue appliquée à l'électromagnétisme en domaine temporel. 17e Colloque International et Exposition sur la Compatibilité Electromagnetique, Jul 2014, Clermont-Ferrand, France. `https://hal.archives-ouvertes.fr/hal-01108010`

[21] D. Tskhakaya, R. Schneider, Optimization of PIC codes by improved memory management, Journal of Computational Physics, Volume 225, Issue 1, 1 July 2007, Pages 829-839

[22] N. Pham, P. Helluy and A. Crestetto. Space-only hyperbolic approximation of the Vlasov equation. CEMRACS 2012, ESAIM: PROCEEDINGS, December 2013, Vol. 43, p. 17-36.

[23] Eliasson, Outflow boundary conditions for the Fourier transformed one-dimensional Vlasov-Poisson system, J. Sci. Comput. , 2001.

[24] P Helluy, M Massaro, L Navoret, N Pham, T Strub. Reduced Vlasov-Maxwell modeling - PIERS Proceedings, August 25-28, Guangzhou, 2014, 2014