Commun. Comput. Phys.
doi: 10.4208/cicp.080409.051009a

Vol. **8**, No. 1, pp. 159-184
July 2010

Direct Vlasov Solvers with High-Order Spectral Element Method

Jin Xu[1,2,*], Brahim Mustapha[1], Peter Ostroumov[1] and Jerry Nolen[1]

[1] *Physics Division, Argonne National Laboratory, 9700 S. Cass Ave., Argonne, IL 60439, USA.*
[2]*Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Ave., Argonne, IL 60439, USA.*

Received 8 April 2009; Accepted (in revised version) 5 October 2009

Available online 12 February 2010

Abstract. This paper presents the development of parallel direct Vlasov solvers using the Spectral Element Method (SEM). Instead of the standard Particle-In-Cell (PIC) approach for kinetic space plasma simulation, i.e. solving the Vlasov-Maxwell equations, the direct method has been used in this paper. There are several benefits to solve the Vlasov equation directly, such as avoiding noise associated with the finite number of particles and the capability to capture the fine structure in the plasma, etc. The most challenging part of direct Vlasov solver comes from high dimension, as the computational cost increases as N^{2d}, where d is the dimension of the physical space. Recently due to fast development of supercomputers, the possibility of high dimensions becomes more realistic. A significant effort has been devoted to solve the Vlasov equation in low dimensions so far, now more interests focus on higher dimensions. Different numerical methods have been tried so far, such as finite difference method, Fourier spectral method, finite volume method, etc. In this paper SEM has been successfully applied to construct these solvers. SEM has shown several advantages, such as easy interpolation due to local element structure and long time integration due to its high order accuracy. Domain decomposition in high dimensions have been used for parallelization, these include scalable parallel 1D and 2D Poisson solvers. Benchmark results have been shown and simulation results have been reported for two different cases: one dimension (1P1V), and two dimensions (2P2V) in both physical and velocity spaces.

AMS subject classifications: 68U20, 68W10, 65C20, 65Z05

Key words: Vlasov equation, Spectral Element Method, Poisson's equation, interpolation, domain decomposition.

*Corresponding author. *Email addresses:* `jin_xu@anl.gov` (J. Xu), `mustapha@phy.anl.gov` (B. Mustapha), `ostroumov@phy.anl.gov` (P. Ostroumov), `nolen@anl.gov` (J. Nolen)

1 Introduction

Plasma and charged particle simulations have great importance in science. There are three different approaches to simulate plasma: the microscopic model, the kinetic model and the fluid model. The microscopic model is governed by Liouville's equation. Each charged particle is described by 6 variables (x,y,z,v_x,v_y,v_z), for a system of N particles, there are 6N variables in total. This requires to solve the Liouville equation in a phase space with 6N dimensions, which obviously exceeds the capability of current supercomputers. On the other hand, the simplest model is the fluid model, which treats the plasma as a conducting fluid with electromagnetic force exerting on it. This leads to solving the Magnetohydrodynamics (MHD) equations in 3D (x, y and z). MHD solves for the average quantities, such as density and charge, which makes it difficult to describe fine structures in the plasma. Due to computer speed limitations, MHD is a realistic approach in plasma simulation. Between these two models is the kinetic model, which solves the charge density function by solving Boltzmann or Vlasov equations in 6 dimensions (x,y,z,v_x,v_y,v_z). The Vlasov equation describes the evolution of a system of particles under the effects of self-consistent electromagnetic fields. This paper deals with the kinetic model. There are two different ways to solve the kinetic model. The most popular one is to represent the beam bunch by macro particles and push the macro particles along the characteristics of Vlasov equation. This is the so called Particle-In-Cell (PIC) method [3], which utilizes the motion of the particles along the characteristics of the Vlasov-equation using a Lagrange-Euler approach. In principle it simplifies the Partial Differential Equation (PDE) to an Ordinary Differential Equation (ODE). The interaction between charged particles, which is called space charge effect, is accounted for by first distributing charges on a fixed grid, then solving Poisson's equation. Then the electric field from the potential solution can be computed. The PIC method has the advantages of fast speed and easy implementation, but similar to MHD, it is hard to describe fine structures in the plasma. Furthermore, there is noise associated with the finite number of particles used in the simulation, and this noise decreases very slowly as $1/\sqrt{N}$ when the number of particles N is increased. The other way to solve the kinetic model is to solve the Vlasov equation directly. This can overcome the shortcomings of the PIC method, but due to the high dimensional nature of the Vlasov equation, numerical simulations have only been conducted in low dimensions such as 1P1V or the axisymmetic case. Recently, 2P2V simulations have been reported [15]. As the speed and memory of supercomputers increases, 2P2V problems can be easily solved and higher dimensions could be realized in the near future.

Since Cheng and Knorr published their original, ground-breaking paper on a fixed mesh in the phase space [7], many different numerical methods have been adopted to solve the Vlasov equation. Generally, they can be divided into the Fourier spectral method, the finite element method, the finite difference method, the finite volume method, and the semi-Lagrangian method.

The Fourier spectral method has been used for domains with periodic boundary conditions [21, 22]. This method can only be applied in regular domains. In order to handle complex geometry, the finite element method has been used [29, 30]. Since the finite element method usually uses low order basis, the accuracy can only be improved by increasing the number of elements in the domain. The finite difference method can achieve high order accuracy by using high-order schemes. Similar to the finite element method, finite difference method can also be applied to irregular domain using the so-called cut-cell scheme.

The finite volume method [13] shows some advantages under certain conditions. The Piecewise Parabolic Method (PPM) [8] and the Van-Leer-Limited [1] scheme use limiters on the reconstruction of the distribution function to maintain monotonicity and positivity. In order to conserve mass, the flux corrected transport (FCT) method [4, 5] and the flux balance method (FBM) [12] have been designed to solve the average Vlasov equation in each cell of the phase-space. The positive and flux conservation (PCF) [13] scheme was proposed to overcome the non-preservation of the positivity, which brings nonphysical phenomena.

Since the phase-space distribution function is constant along the trajectories of the system, which is the Liouville's theorem, the semi-Lagrangian method has been successfully used in solving the Vlasov equation. The semi-Lagrangian method consists of computing the distribution function at each grid point by following the characteristic curves backward, and interpolating the distribution function at the previous time step. E. Sonnendrücker and F. Filet *et al.* have obtained very good results with the semi-Lagrangian method [2,13–15,17,27]. E. Sonnendrücker proposed the use of cubic spline, Lagrange, or Hermite polynomials for the interpolation, and the results are encouraging. However, using a cubic spline destroys the local character of the reconstruction. Makamura and Yabe proposed the cubic interpolated propagation (CIP) method based on the approximation of the gradients of the distribution function. This method is robust, but very expensive in memory as it needs to store f,$\nabla_x f$ and $\nabla_y f$. The Vlasov equation is hyperbolic, and thus suffers from fine-scale filamentation as time increases. Therefore numerical techniques are needed to make the scheme stable and robust.

In all the works mentioned above, different techniques have been adopted to stabilize the simulation. This inevitably modified the original physical problem to some extent. In this paper, we avoid these artificial effects by using high order method and large scale computing. For the high order method, Spectral Element Method (SEM) has been adopted in this paper. It combines the flexibility of the finite element method and high order accuracy of the spectral method. It also has an advantage for interpolation, since the accuracy can be increased by increasing the polynomial order. For large scale computing, SEM also has the advantage of easy parallelization. Using SEM, we can develop a highly scalable parallel Vlasov solver, especially for high-dimensional problems. Arber *et al.* [1] compared different Vlasov solvers and claimed that high order schemes may bring oscillations and introduce a negative distribution function if no further remedy is used. The high order methods used so far are based on finite volume and finite difference scheme, while for semi-Lagrangian method, high order method show better results.

In this paper, we report our results with SEM on parallel computers. Due to inherent filamentation, people have tried to use different numerical techniques, such as adding artificial viscosity or filtering. These techniques are similar to performing large eddy simulations of turbulence. In this paper large scale computing has been applied, which is similar to performing direct numerical simulations of turbulence. Efficient parallel models have been developed based on domain decomposition. The paper is organized in the following way: the Vlasov equation will be introduced in Section 2. SEM will be briefly presented and the numerical methods explained in Section 3 for both 1P1V and 2P2V problems. The parallel algorithms for these problems are discussed in Section 4. Then benchmarks of parallel solvers and simulation results are given in Section 5. At last, the conclusion is drawn in Section 6.

2 Vlasov equation

The evolution of the distribution function of particles $f(\vec{x},\vec{v},t)$ in the phase space $(\vec{x},\vec{v})\in \mathbf{R}^d\times\mathbf{R}^d$ with $d=1,2,3$ and time t, is given by the dimensionless Vlasov equation:

$$\frac{\partial f(\vec{x},\vec{v},t)}{\partial t}+\vec{v}(\vec{x},t)\cdot\nabla_x f(\vec{x},\vec{v},t)+\vec{F}(\vec{x},\vec{v},t)\cdot\nabla_v f(\vec{x},\vec{v},t)=0, \tag{2.1}$$

where the force field $\vec{F}(\vec{x},\vec{v},t)$ can be coupled to the distribution function f. For the Vlasov-Poisson system, this coupling is done through the macroscopic density ρ

$$\rho(\vec{x},\vec{v},t)=\int_{\mathbf{R}^d} f(\vec{x},\vec{v},t)dv. \tag{2.2}$$

The force field which only depends on t and $\vec{x}$ makes the system nonlinear and is given by solving Poisson's equation as shown in Eq. (2.3):

$$\vec{F}(\vec{x},\vec{v},t)=\vec{E}(\vec{x},t),\quad \vec{E}(\vec{x},t)=-\nabla_{\vec{x}}\phi(\vec{x},t),\quad -\Delta_{\vec{x}}\phi(\vec{x},t)=\rho(\vec{x},t)-1, \tag{2.3}$$

where $\vec{E}$ is the electric field and ϕ the electric potential.

2.1 Vlasov equation in 1P1V phase-space

In 1P1V phase-space, the non-dimensional Vlasov equation can be written as follow [1]:

$$\frac{\partial f(x,v,t)}{\partial t}+v(x,t)\frac{\partial f(x,v,t)}{\partial x}+E(x,t)\frac{\partial f(x,v,t)}{\partial v}=0, \tag{2.4}$$

$$E(x,t)=-\frac{\partial\phi(x,t)}{\partial x},\quad -\Delta\phi(x,t)=\frac{\partial E(x,t)}{\partial x}=\rho(x,t)-1, \tag{2.5}$$

$$\rho(x,t)=\int_{-\infty}^{\infty} f(x,v,t)dv. \tag{2.6}$$

2.2 Vlasov equation in 2P2V phase-space

In beam dynamics, a simplified model can be developed in 2P2V as a paraxial model [15] based on the following assumptions:

- The beam is in a steady-state: All particle derivatives with respect to time vanish;
- The beam is sufficiently long so that longitudinal self-consistent forces can be neglected;
- The beam is propagating at constant velocity v_b along the propagation axis z;
- Electromagnetic self-forces are included;
- $\vec{p}=(p_x,p_y,p_z), p_z \sim p_b$ and $p_x,p_y \ll p_b$, where $p_b=\gamma m v_b$ is the beam momentum. It follows in particular that

$$\beta \approx \beta_b = v_b/c, \gamma \approx \gamma_b = (1-\beta_b^2)^{-1/2}; \tag{2.7}$$

- The beam is thin: the transverse dimensions of the beam are small compared to the characteristic longitudinal dimension.

The paraxial model can be written as

$$\frac{\partial f}{\partial z}+\frac{\vec{v}}{v_b}\cdot\nabla_{\vec{x}} f+\frac{q}{\gamma_b m v_b}\left(-\frac{1}{\gamma_b^2}\nabla\Phi^s+\vec{E}^e+(\vec{v},v_b)^T\times\vec{B}^e\right)\cdot\nabla_{\vec{v}} f=0 \tag{2.8}$$

coupled with the Poisson's equation

$$-\Delta_{\vec{x}}\Phi^s=\frac{q}{\epsilon_0}\int_{R^2} f(z,\vec{x},\vec{v})d\vec{v}, \tag{2.9}$$

where Φ^s is the self-consistent electric potential due to space charge. $\vec{E}^e$ and $\vec{B}^e$ are external electric and magnetic fields. v_b is the reference beam velocity.

3 Numerical method

In this section, we will first briefly introduce the SEM. Following that, detailed numerical methods for 1P1V and 2P2V problems will be presented.

3.1 Spectral Element Method

SEM originated in 1980's. It combines the flexibility of the finite element method and the high-order accuracy of the spectral method. SEM is characterized by its close relation with orthogonal polynomials and Gaussian quadrature. Many researchers have contributed to this area, successful SEM shows great advantages compared to other methods in many application areas [9, 10, 18, 20, 25]. In this paper, we try to extend it to plasma simulation. As SEM is a high-order method, the interpolation error decreases exponentially as the polynomial order increases. Due to its high accuracy, it is particularly suited to problems that need long time integration.

In this paper, we use modal bases, which are formed by Jacobi polynomials. Jacobi polynomials $P_p^{\alpha,\beta}(r)$ are eigenfunctions of the following Sturm-Liouville equation

$$\frac{d}{dr}\left[(1-r)^{1+\alpha}(1+r)^{1+\beta}\frac{d}{dr}P_p^{\alpha,\beta}(r)\right]=\lambda_p(1-r)^{\alpha}(1+r)^{\beta}P_p^{\alpha,\beta}(r), \tag{3.1}$$

with $\lambda_p=-p(\alpha+\beta+p+1)$. They have the following orthogonal property

$$\int_{-1}^{1}(1+r)^{\beta}(1-r)^{\alpha}P_p^{\alpha,\beta}(r)P_q^{\alpha,\beta}(r)dr=C\delta_{pq}, \tag{3.2}$$

with C depending on α,β,p. The expansion modal bases are tensor products of the following one dimensional basis defined in Eq. (3.3):

$$\phi_p(r)=\begin{cases}\frac{1}{2}(1-r), & p=0,\\ \frac{1}{4}(1-r)(1+r)P_{p-2}^{1,1}(r), & 0<p<P,\\ \frac{1}{2}(1+r), & p=P,\end{cases} \tag{3.3}$$

where $P_p^{1,1}(r)$ are the Jacobi Polynomials $P_p^{\alpha,\beta}(r)$ with $\alpha=\beta=1$ (Legendre Polynomial) in the standard interval $\Omega=\{r\,|\,-1\le r\le 1\}$. The first and last modes are linear in order to satisfy the boundary conditions. All interior modes are zero at the boundaries.

The main advantages of SEM are its flexibility to handle complex geometry and high-order accuracy. Nearly all operations and data are local, including interpolation, integration, solving Poisson's equation, and transformation between physical and spectral spaces, etc. Locality makes it easy for parallelization and is an important motivation to use SEM in this paper.

3.2 Meshes and bases in 1P1V and 2P2V phase-spaces

Both 1P1V and 2P2V simulations use a structured quadrilateral element mesh. For 2P2V simulations, a quadrilateral mesh has been used in both physical (x,y) space and velocity (v_x,v_y) space separately. In each two dimensional space, modal bases have been used. A mesh of 16 elements and bases of order up to (4,4) are shown in Fig. 1. In 2P2V simulations, similar mesh and bases can be used separately in both physical and velocity spaces.

In 1P1V simulations, the solution in each element can be expanded as following (3.4):

$$f(x,v_x,t)=\sum_{e=0}^{N_e-1}\sum_{i=0}^{P_x-1}\sum_{j=0}^{P_{v_x}-1}\hat{f}_{ij}^{e}(t)\phi_i(x)\phi_j(v_x), \tag{3.4}$$

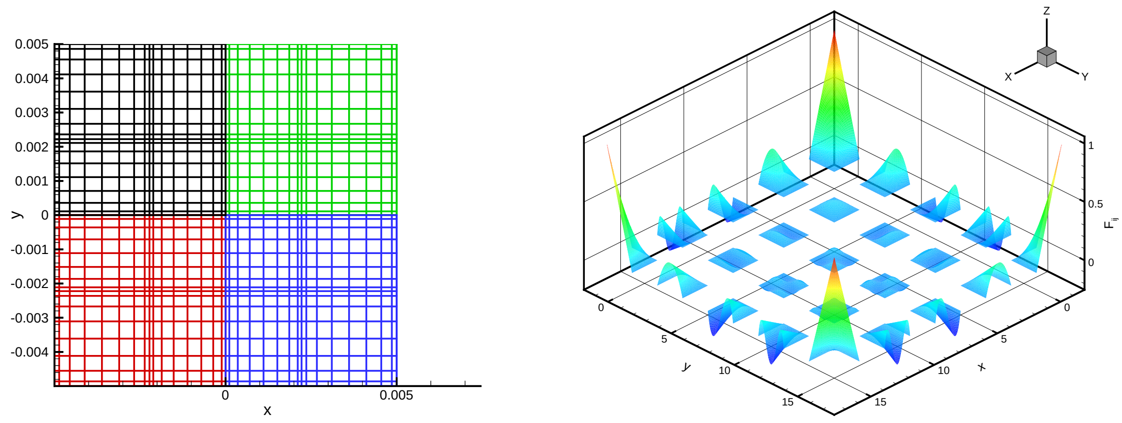

Figure 1: Mesh (left) and modal bases (right).

where

$$\phi_p(x) = P_p^{\alpha,\beta}(r), \quad \phi_p(v_x) = P_p^{\alpha,\beta}(v_r), \quad r = \frac{x - X(e)}{X(e+1) - X(e)}, \quad v_r = \frac{v - (V_e)}{V(e+1) - V(e)}.$$

N_e is the total element number in the (x, v_x) plane, e is the element number, $X(e)$ and $V(e)$ are element boundaries in x and v_x directions, and $-1 \leq r \leq 1$, $-1 \leq v_r \leq 1$, P_x and P_{v_x} are polynomial orders in x and v_x directions respectively.

In 2P2V simulations, the distribution function $f(x,y,v_x,v_y)$ is expanded in the (x,y) and (v_x,v_y) planes separately at different steps in the splitting scheme. Eq. (3.5) shows the spectral element expansion in (x,y) plane, and Eq. (3.6) shows the expansion in (v_x,v_y) plane.

For each $(v_x,v_y) \in \Omega(v_x,v_y)$,

$$f(x,y,v_x,v_y,t) = \sum_{e=0}^{N_x-1} \sum_{i=0}^{P_x-1} \sum_{j=0}^{P_y-1} \hat{f}_{ij}^e(v_x,v_y,t)\phi_i(x)\phi_j(y). \tag{3.5}$$

For each $(x,y) \in \Omega(x,y)$,

$$f(x,y,v_x,v_y,t) = \sum_{e=0}^{N_{v_x}-1} \sum_{i=0}^{P_{v_x}-1} \sum_{j=0}^{P_{v_y}-1} \hat{g}_{ij}^e(x,y,t)\phi_i(v_x)\phi_j(v_y). \tag{3.6}$$

3.3 Semi-Lagrangian method and time splitting scheme

As stated before, the semi-Lagrangian method consists of computing the distribution function at each grid point by following the characteristic curves backward, and interpolating the distribution function at the previous time step. According to Liouville's theorem, the phase-space distribution function is constant along the trajectories of the system.

Therefore, the interpolation at the previous time step equals to the function value at the present time step. In contrast to the Eulerian framework, the semi-Lagrangian scheme allows the use of large time-steps without losing stability. The limitations for stability are that trajectories should not cross and particles should not "overtake" one another. Therefore, the choice of time-step in the semi-Lagrangian scheme is only limited by numerical accuracy.

The time splitting scheme has been used for time integration as proposed by Cheng and Knorr [7]. They are different in 1P1V and 2P2V simulations. Details will be given in the following sections.

3.3.1 1P1V problem

Cheng and Knorr's scheme was proposed for one-dimensional problem. In order to increase the accuracy, we have adopted the algorithm proposed in [26]. Each time step has been divided into three substeps: the first and the third substeps are in velocity space and the second substep is in physical space. The detailed procedure follows:

```
Do istep=1,nstep:
    - Compute $j^n = q\int f^n(x^n,v^n)v^n dv$;
    - Compute $E^{pred} = E^n - j^n\Delta t$ from Ampere's law;
    - Do until $|E^{n+1} - E^{pred}| < \varepsilon$
        · Substep1: $v^{n+1/2} = v^{n+1} - E^{pred}(x^{n+1})\Delta t/2$;
        · Substep2: $x^n = x^{n+1} - v^{n+1/2}\Delta t$;
        · Substep3: $v^n = v^{n+1/2} - E^n(x^n)\Delta t/2$;
        · Interpolate to compute charge density;
        · Solve Poisson's equation for $E^{n+1}$;
        · Update new $E^{pred} = E^{n+1}$.
    - Enddo
Enddo
```

3.3.2 2P2V problem

In 2P2V simulations, as it is very expensive to interpolate in 4 dimensions, the distribution function is updated after each substep. Time splitting is the same as Cheng and Knorr's. Each time step has been divided into three substeps: the first and third substeps are in physical space and the second substep is in velocity space. The detailed procedure follows:

```
Do istep=1,nstep:
    - Substep1: Perform a half time step shift in the (x,y) plane: $f^*(\vec{x},\vec{v}) = f(t^n,\vec{x}-\vec{v}\Delta t/2,\vec{v})$;
    - Compute the electric field at time $t^{n+1/2}$ by substituting $f^*$ in the Poisson's equation;
    - Substep2: Perform a full time step shift in the $(v_x,v_y)$ plane: $f^{**}(\vec{x},\vec{v}) = f^*(\vec{x},\vec{v}-\vec{E}(t^{n+1/2},\vec{x})\Delta t)$;
    - Substep3: Perform a second half time step shift in the (x,y) plane: $f(t^{n+1},\vec{x},\vec{v})=f^{**}(\vec{x}-\vec{v}\Delta t/2,\vec{v})$;
Enddo
```

3.4 Interpolation

For any $(x_0,y_0)\in\Omega(x,y)$, interpolation can be done locally in each element using Jacobi polynomial expansions

$$f(x_0,y_0,v_x,v_y,t)=\sum_{e=0}^{N_x-1}\sum_{i=0}^{P_x-1}\sum_{j=0}^{P_y-1}\hat{g}_{ij}^e(x_0,y_0,t)\phi_i(x_0)\phi_j(y_0), \tag{3.7}$$

$\phi_i(x_0),\phi_j(y_0)$ are the corresponding polynomial values in x and y directions. For comparison, cubic spline interpolation has also been tried. In order to guarantee the positivity of the distribution function in phase space, all negative interpolation values have been set to zero in the simulations. As SEM is a high-order method, increasing polynomial order can reduce the interpolation error. Therefore, SEM can achieve much higher accuracy than cubic splines. Comparisons will be presented in Section 5. This is another motivation to use SEM to directly solve the Vlasov equation. Necessary adjustments are needed to conserve the total mass for long time integration, we have adopted the ideas of Priestley [24] and Gravel and Staniforth [16]. For the simulations in this paper, since a high order SEM has been used, mass loss is small and these adjustments are not necessary.

4 Parallel algorithms

4.1 1P1V simulations

4.1.1 2D domain decomposition

Domain decomposition in the two dimensional plane (x,v_x) has been used. Two communication groups, comx and comvx, have been generated. Comx contains all processors which have the same v_x location, and comvx contains all processors which have the same x location.

4.1.2 Parallel Poisson solvers

Poisson's equation needs to be solved in the x direction, and the zero Dirichlet boundary condition has been used in the 1D Poisson's equation. Therefore each processor will solve Poisson's equation together with other processors in the same comx group. Suppose there are nx and nv processors in comx and comvx, then nv groups solve the same Poisson's equation at the same time. This reduces the parallel efficiency, as it scales with nx, not $nx \times nv$. The scaling of the Vlasov solver will depend on the ratio of Poisson solver time with interpolation time. The scaling with nx will be shown in Section 5.

4.1.3 Pointer array, doubly-linked list and load balancing problem

Back tracking needs to be done for the semi-lagrangian method. Since domain decomposition has been used, the roots of the characteristics from one processor can be located in any other processor. Therefore, a pointer array of $P_x \times P_{v_x}$ has been setup on each processor to store head pointers as shown in Fig. 2. Doubly-linked lists store all grid points on this processor, which has the roots of characteristics located on a corresponding processor. After the pointer array and doubly linked lists have been setup on each processor, a global exchange of data operation is needed to gather all grid points which have roots on a corresponding processor. Then interpolation can be done on each processor locally. At last, exactly opposite scattering operations need to be performed to send all grid points back to the original processor. At the end of each time step, all doubly-linked lists will be cleared and the pointer array will be set to pointer NULL. There is a load balancing problem when using a large number of processors because the roots of the characteristics may not be distributed evenly on all processors, which will influence the final parallel efficiency.

4.2 2P2V simulations

The challenge in 2P2V simulations is that all points in the physical plane need to be back tracked using all possible velocity points in the velocity plane. Similarly, all points in the velocity plane need to be back tracked using all possible points in the physical plane as shown in the right of Fig. 3. This requires an efficient parallel model.

4.2.1 4D domain decomposition

The parallel model performs 2D domain decomposition in both physical (x,y) and velocity (v_x,v_y) planes, therefore totally 4D domain decomposition has been used. This makes it easy to use a large number of processors available on BG/P, and is particularly helpful for direct numerical simulation (DNS) of the Vlasov equation. Similar to 1P1V simulations, two more communicators, comxy and comvxvy, have been generated for operations in different planes.

4.2.2 Parallel Poisson solvers

1. Direct solvers

For comparison, two direct Poisson solvers have been developed for 2P2V simulations. The first one is a global direct Poisson solver where each processor in comxy individu-

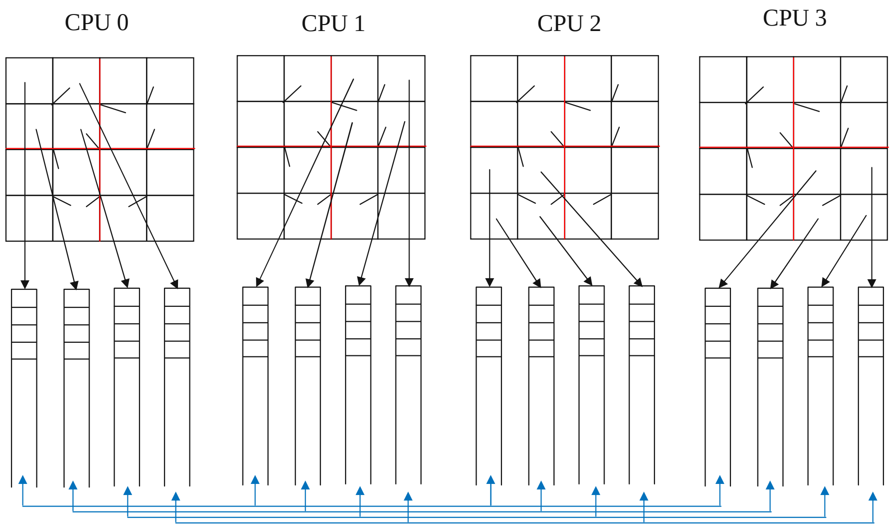

Figure 2: Pointer arrays and load balancing.

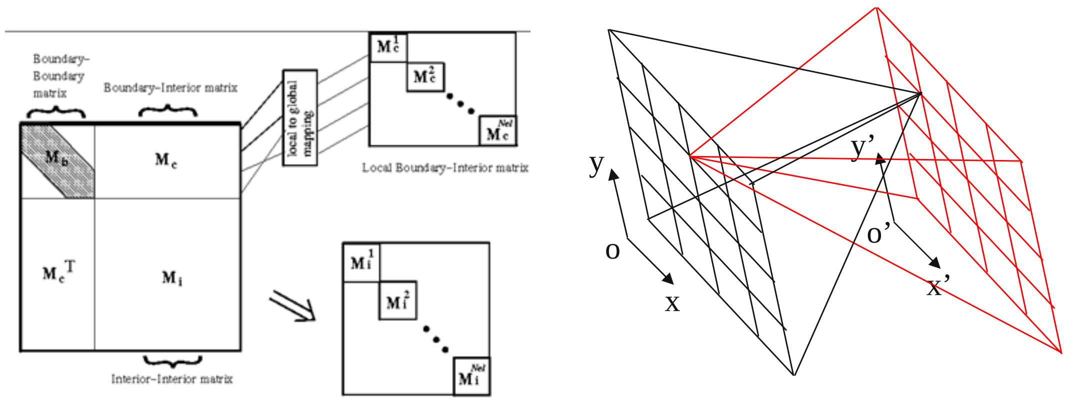

Figure 3: Shur complement (left) and 2P2V data structure (right).

ally solves the same Poisson's equation. This solver is not suitable when the mesh size increases. The second solver makes use of the Shur complement to first solve the boundary matrix then solve interior modes in each quadrilateral as shown in the left of Fig. 3.

Poisson's equation in 2D can be written as follows:

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -\frac{\rho}{\epsilon} = f, \tag{4.1}$$

$$\int_V \nabla^2 u \cdot \phi dv = \int_V f \cdot \phi dv, \tag{4.2}$$

$$\int_V \nabla u \cdot \nabla \phi dv = \int_{\partial V} \vec{n} \cdot (\nabla u \phi) dS - \int_V f \cdot \phi dv. \tag{4.3}$$

This leads to a linear system

$$\begin{pmatrix} A_{bb} & C_{bi} \\ C_{ib} & A_{ii} \end{pmatrix} \begin{pmatrix} u_b \\ u_i \end{pmatrix} = \begin{pmatrix} f_b \\ f_i \end{pmatrix},$$

where A_{bb} is the boundary coefficient, C_{bi} is the boundary interior coefficient, A_{ii} is the interior coefficient, and C_{ib} is the interior boundary coefficient. u_b is the boundary unknown coefficient and u_i is the interior unknown coefficient. u_b is the boundary right-hand side and f_i is the interior right-hand side. Applying the Shur complement, boundary and interior modes are solved separately as following:

$$(A_{bb} - C_{bi}A_{ii}^{-1}C_{bi}^T)u_b = f_b - C_{bi}A_{ii}^{-1}f_i, \tag{4.4}$$

$$u_i = A_{ii}^{-1}(f_i - C_{bi}^T u_b), \tag{4.5}$$

the zero Dirichlet boundary condition has been used to solve Poisson's equation in 2D.

2.Iterative solver

As the mesh size becomes larger, direct solvers cannot be used due to memory limitation on each processor. Therefore, an iterative solver is necessary. The standard conjugate gradient method has been used in this paper. Iterative solvers can make use of solutions at previous time steps, therefore, only a few steps of iteration are needed to reach the required accuracy. Since the Poisson equation is solved in the physical plane, only those processors in the same comxy will contribute to the parallel Poisson solver. It is similar to the situation in the 1P1V case, which reduces the parallel efficiency. The total parallel efficiency depends on the ratio of time for Poisson solver and time for interpolation. Benchmark results will be shown in Section 5.

4.2.3 Pointer array, doubly-linked list and load balancing

Two pointer arrays need to be setup for both the physical and velocity planes, and doubly-linked lists also need to be generated for both planes. A similar load balancing problem exists as in 1P1V case, which reduces the overall parallel efficiency. After each substep, the pointer array needs to be set to NULL pointer, and all doubly-linked lists are cleared.

5 Benchmark results

5.1 Verification and comparison

In order to verify the scheme, we will show the convergence of derivatives and interpolation. Interpolation with a Jacobi polynomial and cubic spline will also be presented.

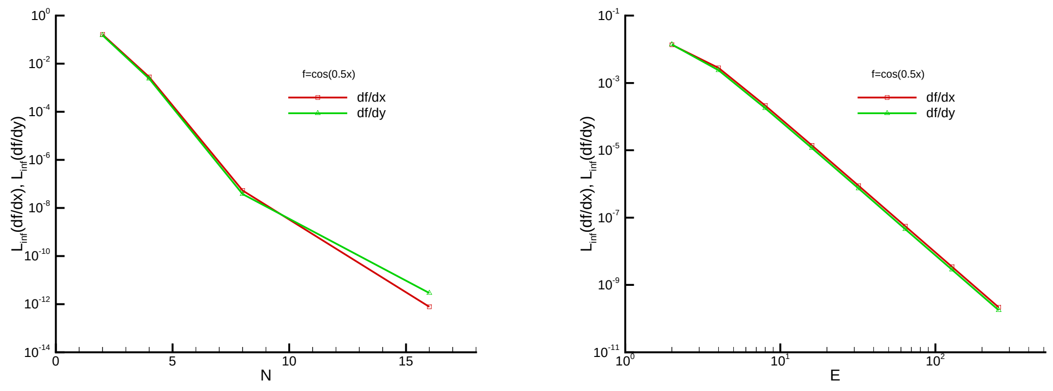

Figure 4: Derivative convergence with polynomial order (left) and element number (right).

5.1.1 Convergence of derivative

In Fig. 4, the errors of derivatives have been shown to decrease with increasing polynomial order and element number. The computation domain is $[0,4\pi]\times[-6,6]$. The function $f(x,y)$ equals $\cos(x)$ or $\cos(y)$ when testing in x or y direction. As it is shown, the error decreases faster by increasing the polynomial order more than the element number.

5.1.2 Convergence of interpolation

In Fig. 5, the errors of 1D interpolation are shown to decrease with increasing polynomial order and element number. The computation domain is $[0,4\pi]\times[-6,6]$. The function $f(x,y)$ equals $\exp(-0.5x^2)/2\pi$ or $\exp(-0.5y^2)/2\pi$ when testing in x or y direction. The red line shows $\frac{\partial f}{\partial x}$, and the green line shows $\frac{\partial f}{\partial y}$. Increasing polynomial order is more efficient to reduce the interpolation error, which is a benefit of SEM. 1D interpolation are used to interpolate electric fields.

In Fig. 6, the errors of 2D interpolation are shown to decrease with increasing polynomial order and element number. The computation domain is $[0,4\pi]\times[-6,6]$. The function $f(x,y)$ equals $\exp(-0.5y^2)(1+0.5\cos(x))/\sqrt{2\pi}$. In 1P1V simulations, 2D interpolation are used to update the distribution function at $t=n+1$.

5.1.3 Comparison of interpolation with Jacobi polynomials and cubic splines

In Fig. 7, we compare the interpolation errors with Jacobi polynomials and cubic splines. Polynomial order N=4 has been used for this test. Similar domain and function have been used as in previous tests. As the polynomial order increases, the distance between mesh points are smaller. From the left plot, it is clear that when using a Jacobi polynomial, the interpolation errors decrease much faster than using a cubic spline. On the right plot, two polynomial orders $N=2,4$ have been compared with cubic spline interpolation. Similar results have been obtained, and even using $N=2$ the errors converges faster than when a cubic spline is used.

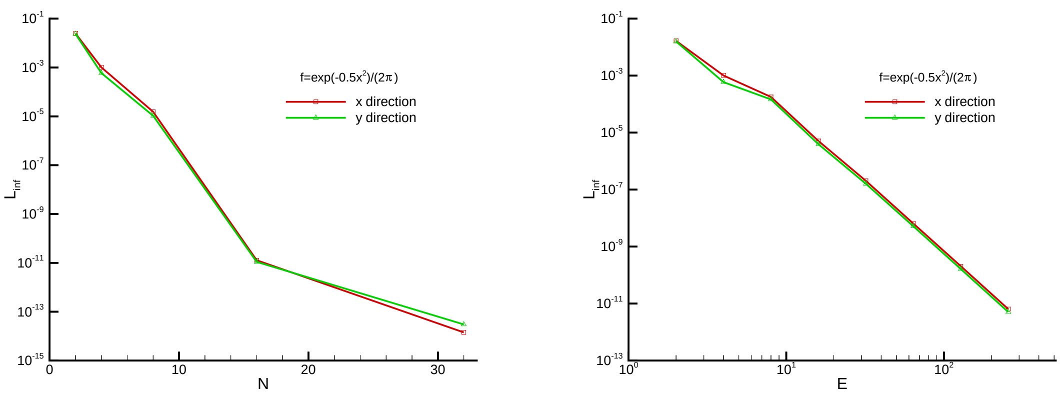

Figure 5: 1D Interpolation convergence with polynomial order (left) and element number (right).

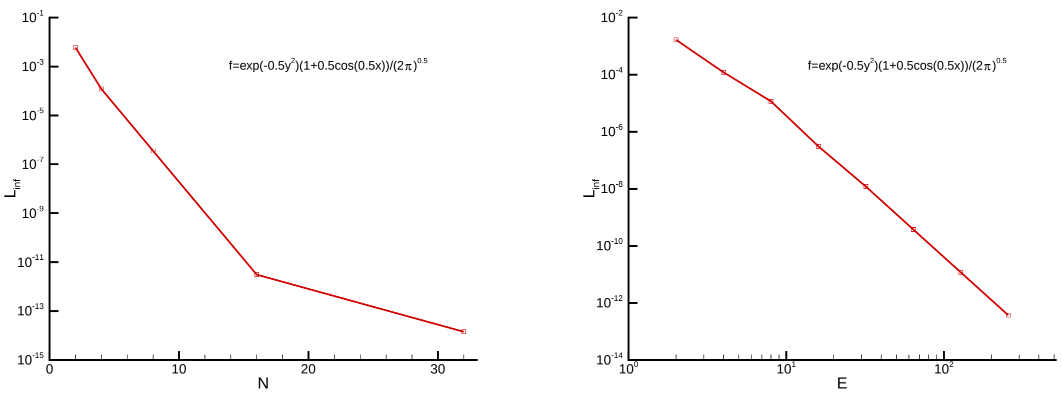

Figure 6: 2D Interpolation convergence with polynomial order (left) and element number (right).

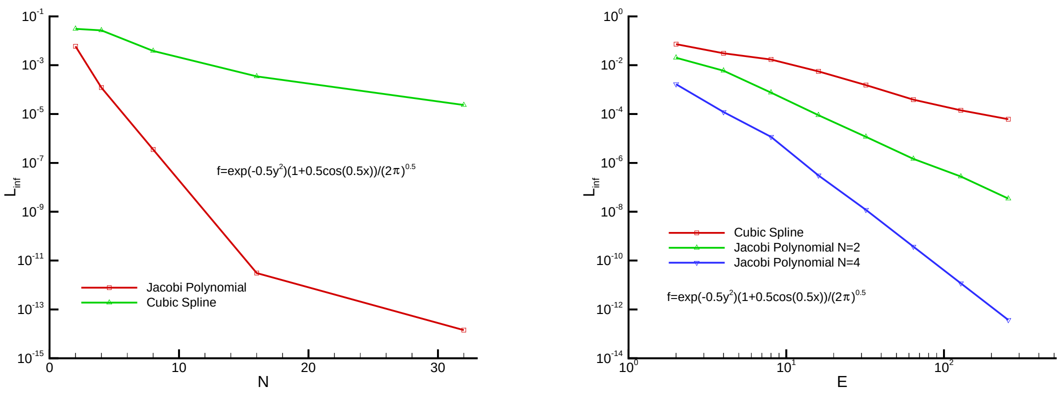

Figure 7: Comparisons of interpolation with Jacobi polynomial and cubic spline as function of polynomial order (left) and element number (right).

Table 1: Strong scaling of 1D iterative Poisson solver (E=32, P=32).

CPU	4	8	16	32
Time (s)	2.91	1.53	7.38e-1	4.22e-1
Parallel Efficiency	1.0	0.951	0.986	0.86

Table 2: Strong scaling of 2D iterative Poisson solver (E=64, P=4).

CPU	16	64	256	1024	4096
Time (s)	286	68	17.2	4.08	1.66
Parallel Efficiency	1.0	1.0	1.0	1.0	0.673

Table 3: Benchmark results for 1P1V Vlasov solver on BG/P at ANL (E=128, P=20).

(CPU_x,CPU_{vx})	Poisson time	Ratio	Total Time	Speedup	Parallel Efficiency
(16,1)	3.36e-02	0.033%	101.37	1.0	1.0
(16,4)	4.78e-02	0.20%	24.2	4.19	1.0
(16,16)	4.79e-02	0.72%	6.655	15.23	0.952
(16,64)	4.53e-02	1.68%	2.7	37.54	0.587

5.2 Scalability

As large scale computing is necessary, we have benchmarked the parallel Poisson and Vlasov solvers for both 1P1V and 2P2V simulations. Details are given in the following.

5.2.1 1D Poisson solver

We have tested the 1D Poisson solver by choosing one processor in the velocity space. The iterative solver relies heavily on matrix-vector multiplication. As can be seen from Table 1, good scaling has been achieved. Since we only increase the processor number in physical space in these benchmarks, the total number of processors will be squared in the real 1P1V simulations if the same number of processors is used in velocity space. The scaling is closely related to the polynomial order being used on each element. The higher the order used, the better scaling can be achieved.

5.2.2 2D Poisson solver

In 2D Poisson solver benchmarks, we set the total processor number in the velocity plane to be one. From Table 2, very good scaling has been achieved up to 1024 processors. Similar to the 1D Poisson solver, the scaling is closely related to the polynomial order used on each element, and the higher the order used, the better scaling can be achieved.

5.2.3 1P1V Vlasov solver

In order to see the scaling of 1P1V Vlasov solver, the number of processors in the physical plane was kept constant at 16. Only the number of processors in the velocity plane

Table 4: Benchmark results for 2P2V Vlasov solver on BG/P at ANL (E=24, P=4).

Total CPU	16	64	256	1024
CPU_x	2	4	4	8
CPU_y	2	4	4	8
CPU_{vx}	2	2	4	4
CPU_{vy}	2	2	4	4
Poisson solver	2.74	1.16	1.33	4.46e-02
1st substep	78.1	22.9	5.84	2.70
2nd substep	120.2	31.5	9.54	2.6
3rd substep	74.8	22.1	5.66	2.66
Total time	275.9	77.7	22.4	8.01
Speedup	1.0	3.55	12.32	34.4
Parallel Efficiency	1.0	0.89	0.77	0.54

was changed. From Table 3, the processor times for the Poisson solvers are nearly the same, and good scaling was achieved. The time to solve the Poisson's equation is closely related to the accuracy parameter ϵ. For these benchmarks, $\epsilon = 1.0 \times 10^{-8}$. The algorithm was optimized so that there are no nested loops or "if" statements in the loops. These contribute to nearly linear scaling for the interpolation.

5.2.4 2P2V Vlasov solver

In the 2P2V case, the interpolation time dominates the Poisson solver time. Therefore, good scaling has been shown on the right plot of Fig. 8 for up to 256 processors. But due to four-dimensional simulation, the mesh size that can be used on 16 processors is relatively small. Therefore, the scaling becomes worse when using 1024 processors. And using 1k processors in both physical and velocity spaces will be one million processors total, which is more than the processor number currently available on BG/P. From Fig. 8, the Poisson solver time stays the same when the number of processors is the same in the physical space as when using 16 or 64 processors total. From Table 4, the Poisson solver time is less than a few percent of total time in one step, and the scaling of the three substeps is better than the Poisson solver because there are less communications in the three substeps than the Poisson solver.

5.3 Stability issue

Due to inherent instability, the beam filamentation becomes smaller and smaller. The instability issue appears if the mesh size is not small enough in our simulations. This is consistent with other researchers. Since we have adopted high order SEM and large scale computing, stability can be sustained for all simulations in this paper without additional stabilizing techniques.

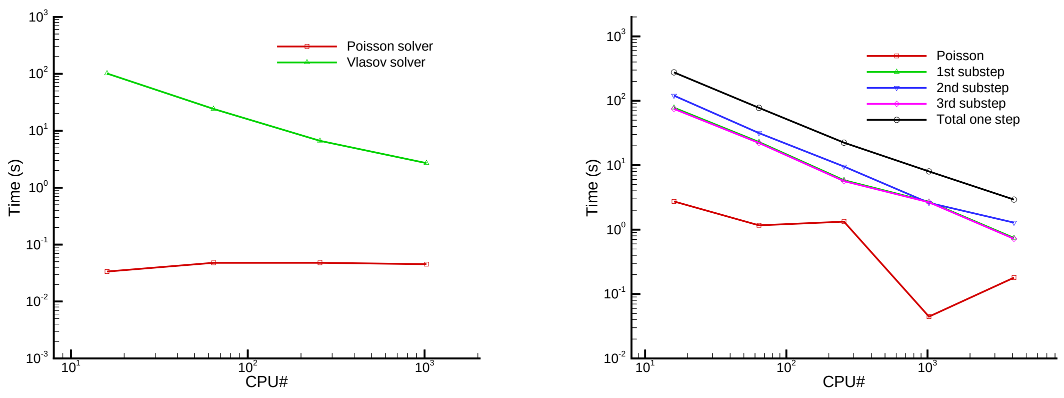

Figure 8: Strong scaling of Vlasov solvers for 1P1V (left) and 2P2V (right).

6 Simulation results

6.1 1P1V results

Several classic problems have been investigated in order to verify the parallel SEM solvers. These include linear and strong Landau damping, and two stream instability. These simulations will be presented in the following sections of the paper.

6.1.1 Linear Landau damping

The initial distribution function is

$$f(0,x,v)=\frac{1}{\sqrt{2\pi}}\exp(-v^2/2)(1-\alpha\cos(kx)),\quad (x,v)\in[0,2\pi/k]\times R. \tag{6.1}$$

where k is the wave number and α=0.01 is the amplitude of the perturbation. We consider linear regimes here. In the x direction, the periodic boundary condition has been used. In the v direction, zero Dirichlet boundary condition has been used. $v_{\max}$ has been set to 6, so the velocity space spans from -6 to 6. $\Delta t=1/8$, P=16, E=64.

The evolution of the square root of the electric energy is shown on the left of Fig. 9. It is supposed to decrease exponentially with a rate of -0.154. This value matches Cheng and Knorr's results very well. Linear Landau damping simulation is not difficult, as the initial variation just damps out.

6.1.2 Strong Landau damping

The initial distribution function and boundary conditions are similar to the linear Landau damping problem, but with $\alpha=0.5$. All other parameters are the same as in the linear Landau damping.

The evolution of the square root of the electric energy is shown on the right of Fig. 9. The slopes of electric energy can also be measured at two periods, and the values -0.281

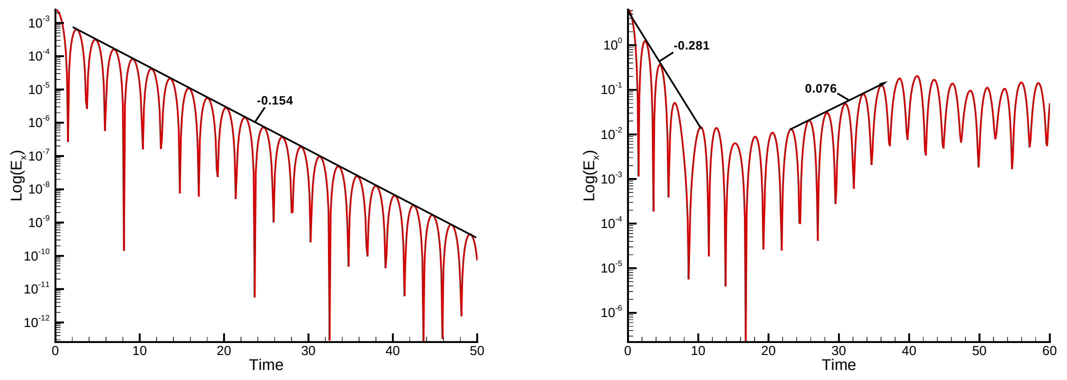

Figure 9: Electric field energy history: Linear Landau Damping (left); Strong Landau Damping (right).

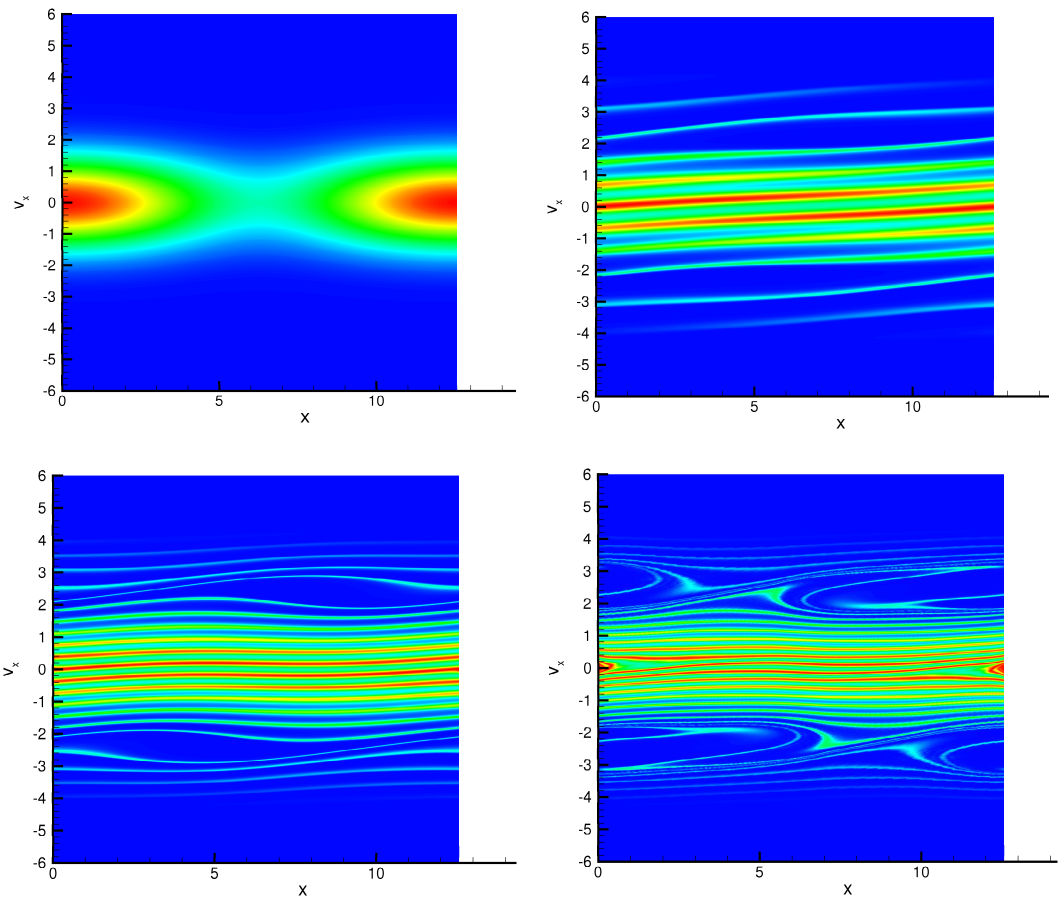

Figure 10: Strong Landau damping: Contour for distribution function $f(x,y)$ in 2D phase space at $t=0$ (top left), $t=16$ (top right), $t=32$ (bottom left) and $t=48$ (bottom right).

and 0.076 are also very close to Cheng and Knorr's. Strong Landau damping simulation is more challenging, as a more complicated structure has been generated. In order to see the details of the distribution function in the phase space, contours of the distribution function f at $t=0,16,32$ and 48 are shown in Fig. 10. These contours clearly show the Vlasov equation transforming an initial smooth distribution function into thinner and thinner ridges. This is the nature and the challenge of solving the Vlasov equation. In order to capture these ridges, larger mesh needs to be used. Otherwise some artificial smoothing needs to be added to make the scheme stable for long time integration. Previous researchers have obtained similar results using small meshes and additional techniques. These inevitably modified the Vlasov equation, similar to performing a large eddy simulation (LES) of turbulence. In this work, we rely on large scale computing and high-order SEM. The Vlasov equation has been simulated directly without any smoothing, which is similar to direct numerical simulation (DNS) of turbulence. These DNS results could be used to check other LES simulations on a small grid.

6.1.3 Two stream instability

The initial distribution function is

$$f(0,x,v)=\frac{1}{\sqrt{2\pi}}v^2\exp(-v^2/2)(1-\alpha\cos(kx)),\quad (x,v)\in[0,2\pi/k]\times R. \tag{6.2}$$

where $\alpha=0.05$, $k=0.5$ and $v_{\max}=6$. $\Delta t=1/8$, P=16, E=64. The boundary condition is similar to the Landau damping simulations.

The simulation starts with initial two ridges with variation, then a vortex is generated. The vortex rotates and generates a hole in the center, which corresponds to trapped particles. This result is also close to that in [2]. With large scale computing a detailed structure of the vortex can be captured, which cannot be done with a small grid. Similar phenomena occur as in strong Landau damping simulations. As time increases, the ridges become thinner and thinner. Larger and larger mesh is needed to accurately simulate these sharp ridges. Contours of the distribution function at $t=0,16,32$ and 48 are shown in Fig. 11.

With the success of these 1P1V simulations, other interesting problems can also be easily studied, such as bump-on-tail instability and ion-acoustic turbulence, etc.

6.2 2P2V results

In order to verify our 2P2V Vlasov solver, we compared it with another solver (TRACK) using a PIC model which has been developed in the physics division at Argonne National Lab. First, we show results of TRACK for KV and Gaussian beams under periodic alternating magnetic force field.

As shown in Fig. 12, exact and nearly periodic $X_{rms}, Y_{rms}, X'_{rms}$ and Y'_{rms} envelopes have been obtained, which verify that TRACK produces correct results for KV and Gaussian beams with periodic magnetic force.

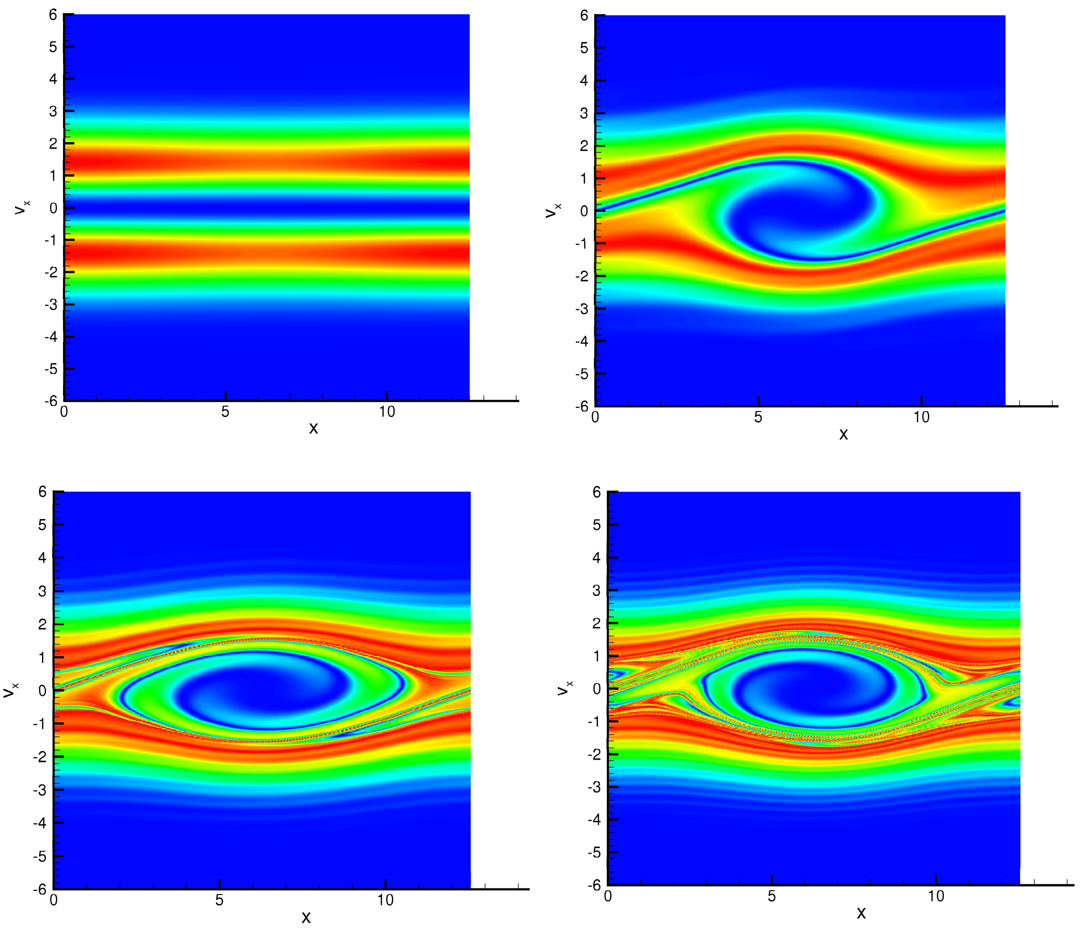

Figure 11: Two stream instability: Contour for distribution function $f(x,y)$ in 2D phase space at $t=0$ (top left), $t=16$ (top right), $t=32$ (bottom left) and $t=48$ (bottom right).

Next, we compare results of our new Vlaosv solver with TRACK for a Gaussian beam. As shown in Fig. 13, the statistics matches well with each other except for small differences which can be removed by using larger mesh. From the phase contours in Fig. 17, beam halos can be clearly seen in XX′ and YY′ planes. These halos are not shown clearly in the PIC simulations, which means that the Vlasov solver is more sensitive in predicting halos. From these comparisons, we can validate our new Vlasov solver, and they can be further applied to simulate beams with complex structures.

In 2P2V simulation, a proton beam has been simulated through a periodic alternating electric field as shown in Fig. 14. Initial emittance $\epsilon = 200\pi$ mm mrad, and energy W=0.2 MeV. The current of the beam is 0.1 A, and reference velocity $v_b = 6.19\times 10^6 m/s$. The physical space is $[-0.12,0.12]\times[-0.12,0.12]$, and the velocity space is $[-8\times 10^5, 8\times 10^5]\times$

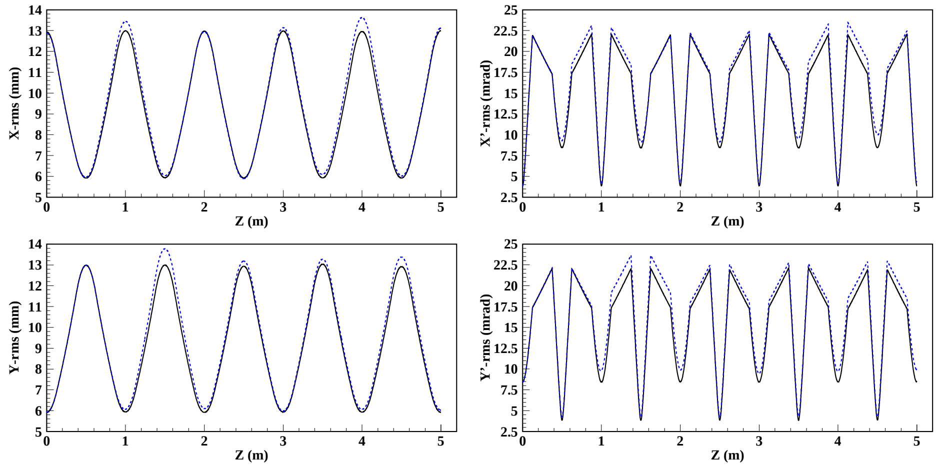

Figure 12: Comparison of TRACK (solid) and Vlasov (dotted) simulations using a KV beam: $X_{rms}(upper-left)$,$Y_{rms}(lower-left)$;$X'_{rms}(upper-right)$,$Y'_{rms}(lower-right)$ for a 100 mA proton beam.

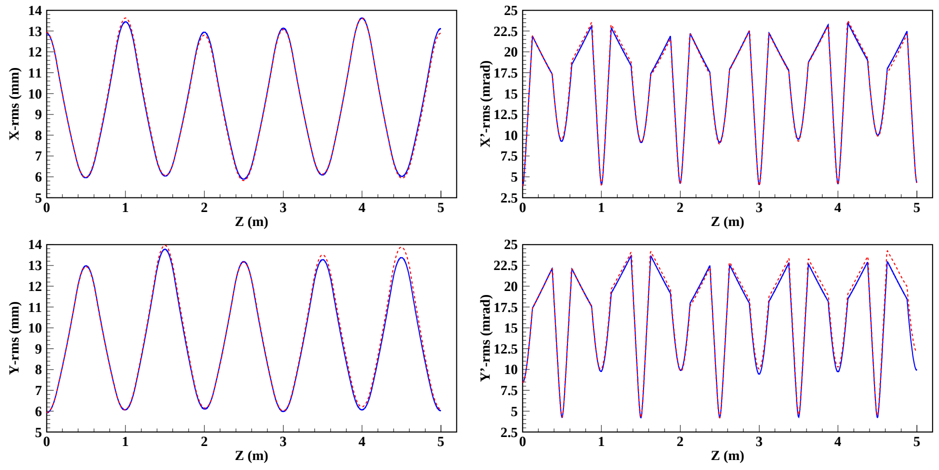

Figure 13: Comparison of TRACK (solid) and Vlasov (dotted) simulations using a Gaussian beam: $X_{rms}(upper-left)$,$Y_{rms}(lower-left)$;$X'_{rms}(upper-right)$,$Y'_{rms}(lower-right)$ for a 100 mA proton beam.

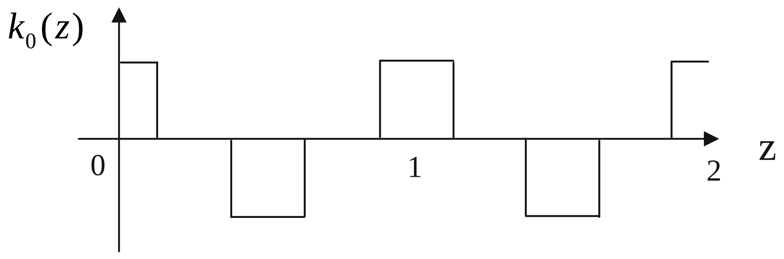

Figure 14: External alternating electric field been used in 2P2V simulation.

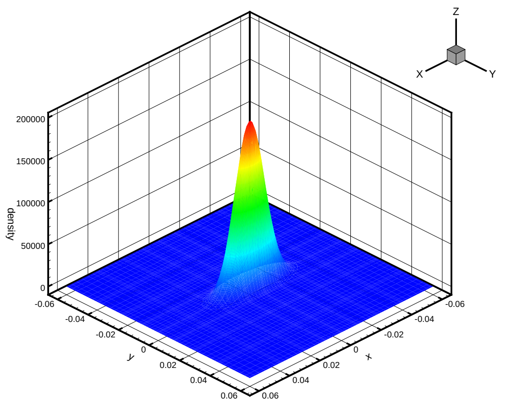
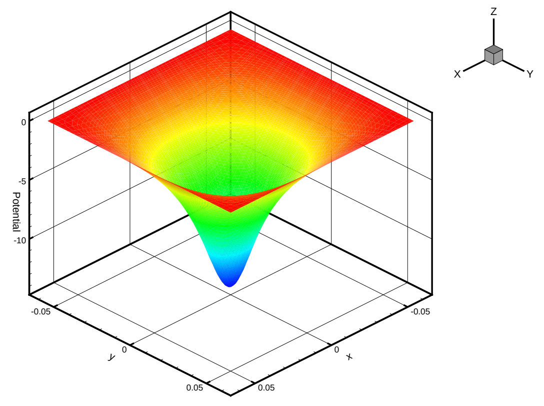

Figure 15: Charge density in physical space (x,y) (left); Electric potential distribution obtained by solving 2D Poisson's equation (right).

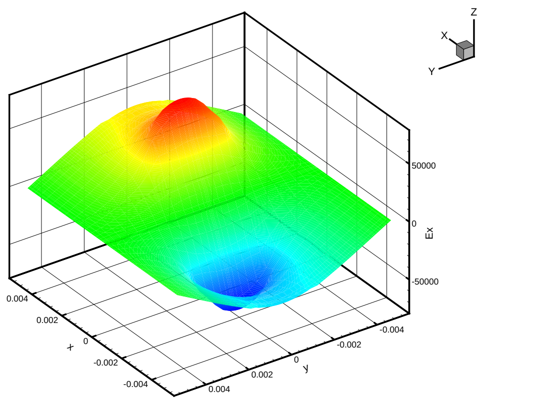
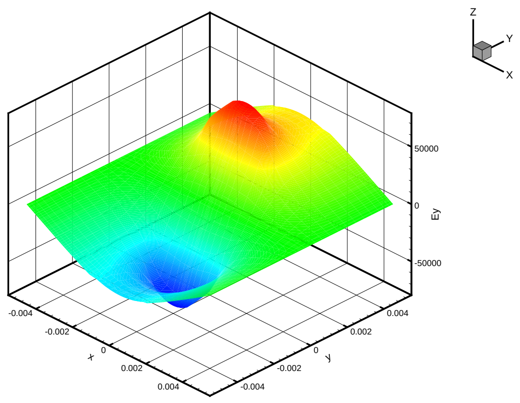

Figure 16: X and Y components of the electric field E_x (left); E_y (right).

$[-8\times10^5, 8\times10^5]$. The periodic alternating electric field is defined as

$$\vec{E}(x,y,z)=\begin{pmatrix} k_0(z)x \\ -k_0(z)y \end{pmatrix},$$

where

$$k_0(z)=\begin{cases} V_b, & 0<z<1/8, \\ 0, & 1/8<z<3/8, \\ -V_b, & 3/8<z<5/8, \\ 0, & 5/8<z<7/8, \\ V_b, & 7/8<z<1, \end{cases}$$

$k_0(z)$ is shown in Fig. 14.

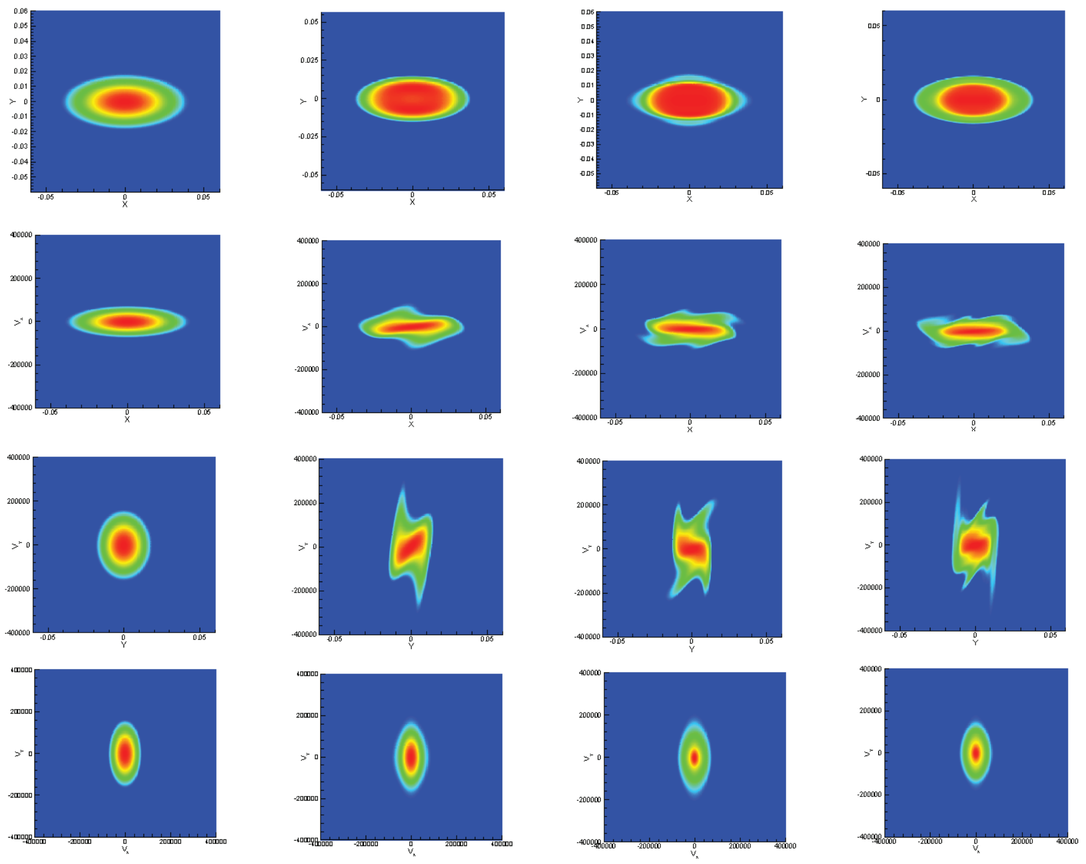

Figure 17: Contour plots of the distribution function $f(x,y,v_x,v_y)$ in 2D phase space at $t=0,1,2,3$ from left to right.

The initial distribution function is Gaussian

$$f_0(x,y,v_x,v_y)=\frac{n_0}{4\pi^2abcd}\exp\left(-\frac{r^2}{2}\right), \tag{6.3}$$

$$r^2=\left(\frac{x}{a}\right)^2+\left(\frac{y}{b}\right)^2+\left(\frac{v_x}{c}\right)^2+\left(\frac{v_y}{d}\right)^2, \tag{6.4}$$

$$a=\frac{a_0}{2},\quad b=\frac{b_0}{2},\quad c=\frac{c_0}{2}=\frac{\epsilon_x}{2a_0},\quad d=\frac{d_0}{2}=\frac{\epsilon_y}{2b_0}, \tag{6.5}$$

where $a_0=0.00925$, $b_0=0.00341$, $c_0=33480$, $d_0=90520$ are derived from the Kapchinky-Vladimirsky (KV) distribution. Currently, the largest mesh that can be used for the simulation is 256^4, and it has been tested with 4096 processors on BG/P at ANL.

Fig. 15 shows the distribution function in physical space and electric potential solved by Poisson's equation. A zero boundary condition has been imposed.

Fig. 16 shows the electric field components E_x and E_y obtained by derivation of the electric potential. They are not zero on all boundaries.

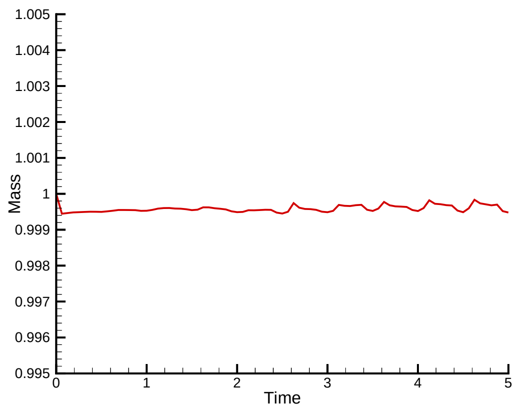

Figure 18: Total mass change with time.

Fig. 17 shows the phase contours at $t=0,1,2,3$ for a 100 mA proton beam. In (x,y) and (x',y') planes, they are nearly the same, as the beam is nearly periodic. In (x,x') and (y,y') planes, the beam halos can be clearly seen.

Fig. 18 shows the total mass with time. Since we have adopted mass conservation technique, which corrects the total mass at each time step. This makes the total mass nearly constant during the entire simulation process.

Based on these simulations, statistical quantities can be calculated easily, such as X_{rms}, Y_{rms}, XX'_{rms} and YY'_{rms}. They are defined as follows:

$$X_{rms}=\sqrt{\frac{\int X(x,y,x',y')^2 f(x,y,x',y')dxdydx'dy'}{\int f(x,y,x',y')dxdydx'dy'}}, \tag{6.6}$$

$$Y_{rms}=\sqrt{\frac{\int Y(x,y,x',y')^2 f(x,y,x',y')dxdydx'dy'}{\int f(x,y,x',y')dxdydx'dy'}}, \tag{6.7}$$

$$XX'_{rms}=\sqrt{\frac{\int X(x,y,x',y')X'(x,y,x',y')f(x,y,x',y')dxdydx'dy'}{\int f(x,y,x',y')dxdydx'dy'}}, \tag{6.8}$$

$$YY'_{rms}=\sqrt{\frac{\int Y(x,y,x',y')Y'(x,y,x',y')f(x,y,x',y')dxdydx'dy'}{\int f(x,y,x',y')dxdydx'dy'}}, \tag{6.9}$$

where $x'=v_x$, $y'=v_y$.

7 Summary

This paper presented our first efforts to develop parallel direct Vlasov solvers with high-order SEM. The advantages and effectiveness of the SEM have been demonstrated.

Both 1P1V and 2P2V Vlasov solvers have been successfully developed using the Semi-Lagrangian method. Domain decomposition has been used for parallelization of these solvers. Scalable Poisson solvers have been developed in them. Benchmarks of the parallel models have shown good scaling on BG/P at ANL. SEM shows its advantages in these direct Vlasov solvers, such as local interpolation, easy parallelization and long time integration. These first explorations are encouraging, and higher dimensional problems are under investigation and will be reported in the near future. Numerical techniques can be added to current solvers on small grids as other researchers, and current DNS results can be used to calibrate them. The results are promising, and with the upcoming peta-scale supercomputers, these direct Vlasov solvers can be applied to real beam dynamics and plasma simulations soon. Many challenges still need to be overcome and many valuable results are expected.

Acknowledgments

We would like to thank Dr. Paul Fischer of the Mathematics and Computer Science (MCS) Division at Argonne (ANL) for his help and suggestions, and extend a special thanks to ALCF support team at ANL. Claire Hillig proofread the paper. This work was completed at ANL, but also used machines at other supercomputing centers, such as NERSC, ARSC and PSC. This work was supported by the U.S. Department of Energy, Office of Nuclear Physics, under Contract No. DE-AC02-06CH11357.

References

[1] T.D. Arber and R.G.L. Vann, A critical comparison of Eulerian-grid-based Valsov solvers, J. Sci. Comp., 180, 339-357 (2002).

[2] N. Besse, E. Sonnendrucker, Semi-Lagrangian schemes for the Vlasov equation on an unstructured mesh of phase space, J. Sci. Comp., 191, 341-376 (2003).

[3] C.K. Birdsall and A.B. Langdon, Plasma Physics via Computer Simulation Inst. of Phys. Publishing, Bristol/Philadelphia (1991).

[4] J.P. Boris, F. Golse and M. Pulvirenti, Kinetic Equation and Asymptotic Theory, Gauthier-Villars, Paris, 2000.

[5] J.P. Boris and D.L. Book, Flux-corrected transport. I: SHASTA, a fluid transport algorithm that works, J. Comput. Phys., 11, 38 (1973).

[6] C. Canuto, M.Y. Hussaini, A. Quarteroni and T.A. Zang, Spectral Methods in Fluid Mechanics, Springer-Verlag, New York (1987).

[7] C.Z. Cheng and G. Knorr, The integration of the Vlasov equation in configuration space, J. Sci. Comp., 22, 330 (1976).

[8] P. Colella and P.R. Woodward, The piecewise parabolic method (PPM) for gas-dynamical simulations, J. Comput. Phys., 54, 174 (1984).

[9] M.O. Deville, P.F. Fischer and E.H. Mund, High-Order Methods for Incompressible Fluid Flow, Cambridge University Press, Cambridge (2002).

[10] M. Dubiner, Spectral Methods on triangles and other domains, J. Sci. Comp., 6, 345 (1991).

[11] B. Eliasson, Numerical modelling of the Fourier transformed two-dimensional Vlasov-Maxwell system, J. Sci. Comp., 190, 501-522 (2003).

[12] E. Fijalkow, A numerical solution to the Vlasov equation, Comput. Phys. Comm., 116, 319 (1999).

[13] F. Filbet, E. Sonnendrücker, P. Bertrand, Conservative numerical schemes for the Vlasov equation, J. Sci. Comp., 172, 166-187 (2001).

[14] F. Filbet, E. Sonnendrücker, Comparison of Eulerian Vlasov solvers, Comput. Phys. Comm., 151, 247-266 (2003).

[15] F. Filbet, E. Sonnendrücker, Modeling and Numerical Simulation of Space Charge Dominated Beams in the Paraxial Approximation Mathematical Models and Methods in Applied Sciences, Vol.16, No.5, 763-791 (2006).

[16] S. Gravel, A. Staniforth, A mass-conserving semi-lagrangian scheme for the Shallow-Water equations, Mon. Weather Review. 243-248 (1994).

[17] M. Gutnic, M. Haefele, I. Paun and E. Sonnendrücker, Vlasov simulations on an adaptive phase-space grid, Comput. Phys. Comm., 164, 214-219 (2004).

[18] J.S. Hesthaven and T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis and Applications, Springer, New York (2008).

[19] R.W. Hockney, J.W. Eastwood, Computer simulation using particles, Adam Hilger, New York, 1988.

[20] G.E. Karniadakis and S.J. Sherwin, Spectral/hp element methods for CFD, Oxford University Press, London (1999).

[21] A.J. Klimas, A method for overcoming the velocity space filamentation problem in collisionless plasma model solutions, J. Comput. Phys., 68, 202 (1987).

[22] A.J. Klimas and W.M. Farrell, A splitting algorithm for Vlasov simulation with filamentation filtration J. Comput. Phys., 110, 150 (1994).

[23] P.N. Ostroumov, J.A. Nolen and B. Mustapha, Computational needs for the RIA accelerator systems, Nucl. Instr. and Meth. A, 558, 25-31 (2006).

[24] A. Priestley, A quasi-conservative version of the semi-lagrangian advection scheme, MOn. Weather Review. 621-629 (1992).

[25] Jie Shen and Tao Tang, Spectral and High-Order Methods with Applications, Science Press of China, (2006).

[26] E. Sonnendrücker, F. Filbet, A. Friedman, E. Oudet and J.-L. vay Vlasov simulations of beams with a moving grid, Comput. Phys. Comm., 164, 2390-395 (2004).

[27] E. Sonnendrücker, J. Roche, P. Bertrand and A. Ghizzo, The semi-Lagrangian method for the numerical resolution of Vlasov equations, J. Comput. Phys., 149, 201 (1998).

[28] J. Xu, B. Mustapha, V.N. Aseev and P.N. Ostroumov, Parallelization of a Beam Dynamics Code and First Large Scale RFQ Simulations, Phys. Rev. ST Accel. Beams 10, 014201 (2007).

[29] S.I. Zaki, L.R. Gardner and T.J.M. Boyd, A finite element code for the simulation of one-dimensional Vlasov plasma I. Theory, J. Comput. Phys., 79, 184 (1988).

[30] S.I. Zaki, L.R. Gardner and T.J.M. Boyd, A finite element code for the simulation of one-dimensional Vlasov plasma II. Applications, J. Comput. Phys., 79, 184 (1988).