SIAM J. SCI. COMPUT.
Vol. 34, No. 6, pp. B909–B924

AN ADAPTIVE, HIGH-ORDER PHASE-SPACE REMAPPING FOR THE TWO DIMENSIONAL VLASOV–POISSON EQUATIONS*

BEI WANG†, GREG MILLER‡, AND PHIL COLELLA§

Abstract. The numerical solution of the high dimensional Vlasov equation is usually performed by particle-in-cell (PIC) methods. However, due to numerical noise, it is challenging to use PIC methods to get a precise description of the distribution function in phase space. To control the numerical error, we introduce an adaptive phase-space remapping which regularizes the particle distribution by periodically reconstructing the distribution function on a hierarchy of phase-space grids with high-order interpolations. The positivity of the distribution function can be preserved using a local redistribution technique. While the one dimensional algorithm has been well established [B. Wang, G. Miller, and P. Colella, *SIAM J. Sci. Comput.*, 33 (2011), pp. 3509–3537], we present the two dimensional algorithm and its parallel implementation in this paper. A performance study of the parallel implementation is included. We discuss the scalability of the algorithm on massively parallel computers.

Key words. particle-in-cell (PIC) methods, adaptive mesh refinement, phase-space remapping, numerical noise, two dimensional Vlasov–Poisson equation, parallel scalability

AMS subject classifications. 35, 65, 76

DOI. 10.1137/120872954

1. Introduction.

The Vlasov equation describes the dynamics of a species of charged particles under electromagnetic fields. In the electrostatic case, the normalized equation reads

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \nabla_{\boldsymbol{x}} f + (-1)^s (\boldsymbol{E} + \boldsymbol{E}^e) \cdot \nabla_{\boldsymbol{v}} f = 0, \tag{1.1}$$

where $f(\boldsymbol{x}, \boldsymbol{v}, t)$ is the distribution function of the species in phase space $(\boldsymbol{x}, \boldsymbol{v}) \in \mathbb{R}^d \times \mathbb{R}^d$ with $d = 1, 2, 3$. s is 0 for positive charges and is 1 otherwise. $\boldsymbol{E}$ and $\boldsymbol{E}^e$ denote the self-consistent and the external electric field, respectively. This equation is the simplest model of collisionless plasmas and beam propagation, which are of importance to controlled thermonuclear fusion and accelerator modeling.

The Vlasov equation is a nonlinear hyperbolic equation in phase space so methods of solution can be guided by the well-established numerical analysis of classical partial differential equations. Accordingly, grid methods in fluid dynamics, such as transform methods, finite-volume methods, and semi-Lagrangian methods, can be employed.

*Submitted to the journal's Computational Methods in Science and Engineering section April 10, 2012; accepted for publication (in revised form) October 18, 2012; published electronically December 18, 2012. This work was supported by the U.S. Department of Energy Office of Advanced Scientific Computing Research under contract number DE-AC02-05CH11231 at Lawrence Berkeley National Laboratory and it used resources of the National Energy Research Scientific Computing center, which is supported by the U.S. Department of Energy Office of Science under contract number DE-AC03-76SF00098.

http://www.siam.org/journals/sisc/34-6/87295.html

†Department of Applied Science, University of California, Davis, CA 95616. Current Address: Princeton Institute for Computational Science and Engineering, Princeton University, Princeton, NJ 08540 (beiwang@princeton.edu).

‡Department of Chemical Engineering and Materials Science, University of California, Davis, CA 95616 (grgmiller@ucdavis.edu). This author's work was supported by DOE contract number DE-SC0001981.

§Applied Numerical Algorithms Group, Lawrence Berkeley National Laboratory, MS 50A-1148, Berkeley, CA 94720 (colella@hpcrdm.lbl.gov).

Operator splitting was successfully applied to the solution of the Vlasov equation by Cheng and Knorr [3] in the 1970s. It reduces the solution of the multidimensional Vlasov equation to a set of one dimensional advection problems, and therefore has become a widely used technique. Even with these well-established algorithms, performing high dimensional simulations using grid methods is still a challenging task. The issue is the computational time and memory cost in dealing with the whole six dimensional phase space. With the advances of today's high performance computers, grids methods have achieved large development in the last decade. In semi-Lagrangian methods, Sonnendrücker et al. [30] introduced the cubic spline method. Nakamura and Yabe [27] introduced the cubic interpolated propagation method. In finite-volume methods, Fijalkow [15] presented the flux balance method. A similar idea is used in a high-order finite-volume method based on mapped coordinate by Colella, Dorr, and Hittinger [6]. Filbet, Sonnendrücker, and Bertrand [16] proposed the positive and flux conservative scheme using limiters.

A more widely used approach for the solution of the Vlasov equation is a PIC method [19, 1]. In PIC methods, the particles, a Lagrangian discretization of the distribution function, follow trajectories computed from the characteristic curves given by the Vlasov equation, whereas the self-consistent fields are calculated on a grid. Since the methods employ the fundamental equations without much approximation, it allows us to observe most of the physics in a plasma system with relatively few particles. However, as with all other particle methods, PIC methods suffer from numerical noise such that they have difficulty in simulating some problems, e.g., problems where velocity coordinates have a large dynamic range. To remedy this deficiency, there are usually two approaches. One method is the so-called δf method [13, 28, 22], discretizing only the perturbation δf with respect to an equilibrium state f_0 based on a particle method. The δf method has been successfully used in realistic applications, e.g., microtubulence in magnetic confined plasmas [23]. The limitation of this method is that it can only be applied to the problems which are close to equilibrium. An alternative approach is through periodically reconstructing the distribution function on a grid in phase space. Such remapping techniques have been used in particle methods in fluid dynamics [8, 2], i.e., vortex methods and smoothed particle hydrodynamics (SPH), to maintain regularity of the particle distribution and thereby improve accuracy, but have much more limited use in PIC methods in plasma physics. It is worth mentioning that early work of Denavit [12] and more recent work of Vadlamani [31] and Yang [4] used the idea of remapping for PIC methods. However, they used a low-order interpolation function which resulted in their method being only first-order accurate.

We investigated a high-order and positive remapping scheme to PIC methods for the solution of the one dimensional Vlasov–Poisson equations [32]. The initial numerical experiments on a set of classical plasma problems in one dimension are very encouraging. Remapping significantly reduces numerical noise and results in a more consistent second-order convergence rate in the electric field error. We also studied the effects of integrating mesh refinement to the uniform remapping. This is motivated by the observation that remapping, a numerical diffusive procedure, tends to create a large number of small-strength particles at the low density region of the distribution function. Mesh refinement has the potential to reduce this side effect.

In this paper, we extend the algorithm to the solution of the two dimensional Vlasov–Poisson equations. This includes the use of an efficient Poisson solver with infinite domain boundary conditions for beam problems. High dimensional simulations are very expensive with respect to memory usage. We perform the simulation

on massively parallel computers using domain decomposition. A performance study of the parallel implementation is presented. Two alternative scaling strategies, which are the extension of the current research, are also discussed. We consider two types of numerical tests: plasma problems including linear Landau damping and the two stream instability, and a beam problem based on the paraxial model [17].

The rest of this paper is organized as follows. In section 2, we first review the classical PIC methods for the Vlasov–Poisson equations. An efficient algorithm which solves the Poisson equation with infinite boundary conditions is briefly described. Then we present the high-order and positive remapping algorithm on a hierarchy of locally refined grids in two dimensions. Section 3 discusses the parallel implementation of the algorithm. In section 4, we show numerical results, including a performance study of the parallel implementation in section 4.4.

2. Algorithms

2.1. PIC methods. PIC methods are based on the Lagrangian description of the Vlasov equation

$$\frac{df(\boldsymbol{X},\boldsymbol{V},t)}{dt}=0, \tag{2.1}$$

where the characteristics $(\boldsymbol{X}(t),\boldsymbol{V}(t))$ are the solution of the equation of motion:

$$\frac{d\boldsymbol{X}}{dt}=\boldsymbol{V}(t),\quad \frac{d\boldsymbol{V}}{dt}=(-1)^s(\boldsymbol{E}(\boldsymbol{X},t)+\boldsymbol{E}^e(\boldsymbol{X},t)) \tag{2.2}$$

with initial conditions $\boldsymbol{X}(t=0)=\boldsymbol{x}$ and $\boldsymbol{V}(t=0)=\boldsymbol{v}$.

In the beginning, the distribution function is approximated by a collection of point particles,

$$f(\boldsymbol{x},\boldsymbol{v},t=0)\approx\sum_k q_k\delta(\boldsymbol{x}-\boldsymbol{x}_k)\delta(\boldsymbol{v}-\boldsymbol{v}_k), \tag{2.3}$$

where $(\boldsymbol{x}_k,\boldsymbol{v}_k)$ is a initial particle location at the cell center of a grid in phase space (quiet start). $q_k=f(\boldsymbol{x}_k,\boldsymbol{v}_k,t=0)h_xh_yh_{v_x}h_{v_y}$ is the weight of a particle. Then each particle follows a trajectory described by the equation of motion,

$$\frac{dq_k}{dt}=0,\quad \frac{d\tilde{\boldsymbol{X}}_k}{dt}(t)=\tilde{\boldsymbol{V}}_k(t),\quad \frac{d\tilde{\boldsymbol{V}}_k}{dt}(t)=(-)^s(\tilde{\boldsymbol{E}}_k(t)+\boldsymbol{E}^e_k(t)), \tag{2.4}$$

where $\tilde{\boldsymbol{X}}_k(t=0)=\boldsymbol{x}_k$ and $\tilde{\boldsymbol{V}}_k(t=0)=\boldsymbol{v}_k$.

At any time that a smooth representation of the distribution function is required, we approximate the function with a collection of finite size particles, where the exact delta function is replaced by a smoothed delta function. That is,

$$f(\boldsymbol{x},\boldsymbol{v},t)\approx\sum_k q_k\delta_{\boldsymbol{\varepsilon_x}}(\boldsymbol{x}-\tilde{\boldsymbol{X}}_k(t))\delta_{\boldsymbol{\varepsilon_v}}(\boldsymbol{v}-\tilde{\boldsymbol{V}}_k(t)),\quad t>0. \tag{2.5}$$

The smoothed delta function satisfies

$$\int_{\mathbb{R}^2}\delta_{\boldsymbol{\varepsilon}}(\boldsymbol{y})d\boldsymbol{y}=1 \tag{2.6}$$

and

$$\delta_{\boldsymbol{\varepsilon}}(\boldsymbol{y})=\prod_{d=0}^{1}\frac{1}{\varepsilon_d}u\left(\frac{y_d}{\varepsilon_d}\right), \tag{2.7}$$

where u is any interpolation function and $\boldsymbol{\varepsilon}$ is the stencil size. Usually, the stencil size for the smoothed delta function in physical space $\boldsymbol{\varepsilon_x}$ is chosen as the same as the mesh spacing of the Poisson solver. The typical interpolation function for PIC methods is the first-order interpolation function

$$u_1(z) = \begin{cases} 1 - |z| & 0 \le |z| \le 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

The flow of a PIC scheme is as follows:

- Assign particle charges on a grid in physical space,

$$\tilde{\rho}(\boldsymbol{x_j}, t) = \sum_k q_k \delta_{\boldsymbol{\varepsilon_x}}(\boldsymbol{x_j} - \tilde{\boldsymbol{X}}_k(t)), \tag{2.9}$$

  where $\boldsymbol{j} \in \mathbb{Z}^2$ are the node index of the Cartesian grid. The grid size is chosen to be the same as the stencil size of the smoothed delta function $\boldsymbol{\varepsilon_x}$. In the case of the first-order interpolation function, for each node $\boldsymbol{j}$, the sum is restricted to the particles with $|\boldsymbol{x_j} - \tilde{\boldsymbol{X}}_k(t)| \le \boldsymbol{\varepsilon_x}$.
- Solve the Poisson equation on the grid with a second-order finite-difference method:

$$-(\triangle^H \phi)_{\boldsymbol{j}} = -\sum_{d=0}^{1} \frac{\phi_{\boldsymbol{j}+\boldsymbol{e}^d} - 2\phi_{\boldsymbol{j}} + \phi_{\boldsymbol{j}-\boldsymbol{e}^d}}{\boldsymbol{\varepsilon}^2_{\boldsymbol{x}_d}} = (-1)^s \tilde{\rho}_{\boldsymbol{j}} + \rho_{\text{background}} \tag{2.10}$$

  and

$$\tilde{\boldsymbol{E}}^d_{\boldsymbol{j}} = \frac{\phi_{\boldsymbol{j}-\boldsymbol{e}^d} - \phi_{\boldsymbol{j}+\boldsymbol{e}^d}}{2\boldsymbol{\varepsilon}_{\boldsymbol{x}_d}}. \tag{2.11}$$

  $\rho_{\text{background}}$ is the background charge density if applicable. With given boundary conditions, the discrete Poisson equation is usually solved by a fast Poisson solver, such as FFTs or multigrid methods.
- Interpolate the calculated field back to the particle locations with the same interpolation function in (2.7). It is worth mentioning that a different interpolation function will introduce self-force errors [7].
- Integrate the equation of motion numerically, for example, using the second-order Runge–Kutta method.

To model beam problems, the Poisson equation with infinite domain boundary conditions needs to be solved. We compute the solution using a modified version of the James–Lackner method [21, 20]. We refer the reader to the work by McCorquodale et al. [24, 25] for details. This method solves two Dirichlet boundary problems plus a boundary-to-boundary convolution and has the potential to scale to a large number of processors.

2.2. Particle remapping. The convergence of particle methods for the one dimensional Vlasov–Poisson equations has been investigated by Cottet and Raviart [9]. Their result shows that particle overlapping and regularization are important for the convergence of the methods. Specifically, the truncation error of a particle method is amplified by a time dependent exponential term. Based on Cottet and Raviart's work, we extend the error analysis to PIC methods [32]. Our result is one order higher in the truncation error. However, as in Cottet and Raviart's analysis, the truncation

error is amplified by a time dependent exponential term. The analysis motivates the use of remapping technique, a widely used strategy in particle methods in fluid dynamics, to control the exponential error. The basic idea of remapping is simple. Since particles will gradually move away from the exact trajectories due to numerical error, we can reduce the displacement by periodically reproducing the distribution function $f(\boldsymbol{x}, \boldsymbol{v}, t)$ on a grid by interpolation. A new set of particles, which is created from the grid representation, then replace the distorted particle distribution. The later step is identical to the initial step of PIC methods that we initialize the particle positions and weights in (2.3). The error due to remapping will depend on the order of the interpolation function.

In the previous work [32], we successfully applied the remapped PIC method to the one dimensional Vlasov–Poisson system. The remapping scheme was extended in three aspects compared with the standard scheme. First, we used high-order interpolation functions which improve accuracy but do not preserve positivity. Second, we preserved the positivity of a high-order interpolation by redistributing the excess charge into its local neighborhood. The local redistribution algorithm is based on the mass redistribution idea of Chern and Colella [5], which is first applied to enforce positivity preservation by Hilditch and Colella [18]. Third, instead of reinitializing on a uniform grid, we reproduced the distribution function on a hierarchy of locally-refined grids. Remapping on a hierarchy of locally refined grids significantly reduces the number of small-strength particles located at the tail of the distribution function. The high-order, positive, and adaptive remapping scheme in high dimensional phase space is described below.

2.2.1. High-order remapping. The overall accuracy introduced by remapping will be one order lower than the order of the interpolation function since we lose one order of accuracy in the evolution step. For example, the interpolation function with second-order accuracy only results in a first-order method overall. In this paper, we consider an interpolation function with third-order accuracy derived by Monaghan [26]. The function in one dimension can be expressed as

$$W_4(x,h) = \begin{cases} 1 - \frac{5s^2}{2} + \frac{3s^3}{2} & 0 \le s = \frac{|x|}{h} \le 1, \\ \frac{1}{2}(2-s)^2(1-s) & 1 \le s = \frac{|x|}{h} \le 2, \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

The one dimensional expression can be generalized to four dimensions by tensor product,

$$\boldsymbol{W}_4(\boldsymbol{x_i} - \boldsymbol{x}_k) = \prod_{d=0}^{3} W_4(x_{\boldsymbol{i}}^d - x_k^d, h^d), \tag{2.13}$$

where h^d is the remapping mesh spacing in phase space. $\boldsymbol{i}$ and k denote the index for the cell-centered grid and the particles, respectively.

This function, called a modified B-spline, conserves the total charge and represent a quadratic polynomial exactly. In addition, the first- and the second-order derivative of $W_4(x,h)$ are continuous. The smoothness property of this modified B-spline is particularly good for scattered data interpolation. However, as with all other high-order interpolation functions, W_4 is not positivity preserving. An interpolation function without positivity might create nonphysical negative charge. This should be avoided in simulations.

2.2.2. Positivity. The positivity preserving algorithm is based on the mass redistribution idea of Chern and Colella [5], first applied to enforce positivity preservation by Hilditch and Colella [18]. In the algorithm, we redistribute the undershoot of cell $\boldsymbol{i}$,

$$\delta f_{\boldsymbol{i}} = \min(0, f_{\boldsymbol{i}}^n), \tag{2.14}$$

to its neighboring cells $\boldsymbol{i} + \boldsymbol{\ell}$ in proportion to their capacity ξ,

$$\xi_{\boldsymbol{i}+\boldsymbol{\ell}} = \max(0, f_{\boldsymbol{i}+\boldsymbol{\ell}}^n). \tag{2.15}$$

The distribution function is conserved, which fixes the constant of proportionality

$$f_{\boldsymbol{i}+\boldsymbol{\ell}}^{n+1} = f_{\boldsymbol{i}+\boldsymbol{\ell}}^n + \frac{\xi_{\boldsymbol{i}+\boldsymbol{\ell}}}{\sum_{\boldsymbol{k}\neq 0}^{\text{neighbors}} \xi_{\boldsymbol{i}+\boldsymbol{k}}} \delta f_{\boldsymbol{i}} \tag{2.16}$$

for $\boldsymbol{\ell} \neq 0$ such that cell $\boldsymbol{i} + \boldsymbol{\ell}$ is a neighbor of cell $\boldsymbol{i}$. Superscript n and $n+1$ denote the interpolated value before and after redistribution, respectively.

The drawback of this approach is that positivity is not guaranteed in a single pass. One might have to apply the method iteratively. In practice, however, we find a few iterations are sufficient.

2.2.3. Mesh refinement. Mesh refinement is an attractive option in improving the efficiency of phase-space remapping. The distribution function in phase space is inhomogeneous, e.g., a Maxwellian distribution in velocity space. If the particles in the major body of the distribution function play an important role in the physics we are interested in, a uniform particle distribution may result in a large number of unnecessary particles in the tail of the distribution function. Remapping on a hierarchy of locally refined grids is a good strategy to create a more efficient and optimal particle distribution. From another point of view, remapping through interpolation is a numerically diffusive procedure. This also results in a large number of small-strength particles near the tail of the distribution function. The situation becomes worse as we apply remapping frequently. Remapping on a hierarchy of locally refined grids, with a coarser grid covering the tail of the distribution function, can reduce the number of those small-strength particles. In the following, we present an algorithm of remapping with mesh refinement. In designing the algorithm, we have two guiding principles: the total charge should be conserved and the overall accuracy on the field needs to be maintained.

Before explaining the algorithm, we introduce the definition of a composite grid. We define a hierarchy of cell-centered grids Ω_ℓ, where $0 \leq \ell \leq \ell_{\max}$. $\Omega_{\ell=0}$ is the coarsest grid that covers the whole problem domain. The finer grids $\Omega_{\ell>0}$ are constructed as a union of cell-centered rectangles (see Figure 2.1). The mesh spacing of each level is $\boldsymbol{h}_\ell = \boldsymbol{h}_{\ell-1}/\boldsymbol{r}_{\ell-1}$, where $\boldsymbol{r}_{\ell-1}$ is the refinement ratio of level $\ell - 1$. In four dimensions, $\boldsymbol{h}_\ell \in \mathbb{R}^4$ and $\boldsymbol{r}_\ell \in \mathbb{Z}^4$. The composite grid consists of valid grids at all levels, where a valid grid is defined as a region not overlain by a finer grid. That is,

$$\Omega_c = \sum_{\ell=0}^{\ell_{\max}} \left(\Omega_\ell \setminus P_{\ell+1}^\ell(\Omega_{\ell+1})\right), \tag{2.17}$$

where $P_{\ell+1}^\ell$ is the operator projecting from level $\ell + 1$ to level ℓ.

At the beginning, a set of particles is created from the cell center of the composite grid. In the remapping step, each particle first finds the valid cell in the composite

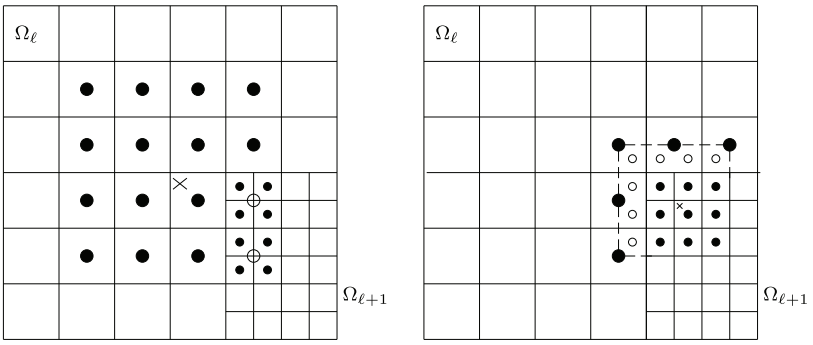


FIG. 2.1. *Cross signs denote the particle locations. The valid and the invalid deposited cells are denoted by filled circles and open circles, respectively. The refinement ratio is* $\mathbf{r}_0 = (2, 2)$ *in the plots. Left: Particle is at the coarser level side. Right: Particle is at the finer level side. Cross signs denote the particle locations.*

grid it belongs to. One particle can only belong to a single valid cell. If the cell is far enough away from a coarse fine-interface, the charge can be interpolated on the grid as in (2.12). If the cell is near a coarse-fine interface such that the interpolation stencil intersects the coarse-fine interface, special care must be taken. First, we interpolate the charge on the surrounding cells as usual. After deposition, a correct is made so change transfer occurs only to valid cells. There are two cases depending on where the invalid cell is located. If the invalid cell is in a coarser level and it is covered by the valid cells of a finer level, we transfer the deposited charge from the coarser level to the finer level through interpolation. On another hand, if the invalid cell is outside the grid of the current level, the charge is transferred by projection. Figure 2.1 shows the algorithm in two dimensions. The four dimensional case can be generalized easily. It is worth mentioning that we lose one order of accuracy in interpolating the coarser level charge into the finer level. However, since the coarse-fine interface is in co-dimension one, the expected accuracy in the field, e.g., second-order, will be preserved in L_∞ norm error. Our current implementation doesn't have time-dependent adaptivity. This feature can be incorporated by selecting some refinement criterion, for example, that all cells phase space have a similar number of particles. Each cell in phase space has a similar number of particles.

3. Parallel implementation. The parallel implementation of the algorithm is based on domain decomposition in physical space. The physical space is decomposed into M disjoint patches. Given a parallel machine with N processors, each patch is assigned to a processor cyclically. Particles are assigned to patches according to their physical space positions. Using MPI, patches communicate with each other through ghost-cells and particles move between patches. The current implementation is designed to be scalable in the weak sense; that is, for the case where the number of processor increases as the problem size increases. This scaling strategy has widely used in particle in cell simulations for fusion plasmas and has obtained excellent scaling to a large number of processors [14, 33]. We present a parallel performance study of the current implementation in section 4.4.

Domain decomposition in physical space has potential limitation when the grid size in phase space increases, for example, when we need to increase the resolution for a simulated problem. The restriction is that the computational time and memory usage will be increasing proportionally to the grid size in velocity space, since the

number of MPI processes is scaled with the grid size in physical space only. In the worst case, the processor will be out of memory. We show this issue in section 4.4.

In modern high performance computing platforms with multicore processors, the memory limitation issue could be alleviated to some extent by using a lesser number of MPI processes in a single node while exploring the rest parallelism in the node with shared memory OpenMP. Meanwhile, we discuss two alternative implementations which have the potential to scale well in both device scaling and phase-space grid scaling as the extension of the current research. One method is based on domain decomposition in phase space. The phase-space domain is decomposed into patches. Particles are assigned to patches according to their phase-space positions. Another method is to introduce an additional layer of parallelization, a particle decomposition, as a supplement to the domain decomposition in physical space. In this case, each patch in physical space can have more than one processor associated with it, whereas each processor holds a fraction of the total number of particles in that subdomain. In both methods, it might be the case that particles belong to the same cell in physical space are distributed on different processors. Since the Laplacian operator is linear, we can choose to solve the Poisson equation separately on different processors. The total fields are then obtained by MPI_Allreduce. Although we pay the price of solving the Poisson equation repeatedly, the overheads will be small since the Poisson solver only takes a very small fraction of the total computational time (less than several percent).

4. Numerical tests. We demonstrate PIC methods with adaptive phase-space remapping on a set of classical plasma and beam problems in two dimensions, including linear Landau damping, the two stream instability, and beam propagation in the paraxial model. In the simulations, there are two sets of grids, a two dimensional grid in physical space for the solution of Poisson equation and a four dimensional grid in phase space for remapping, which we will refer to as Poisson grid and remapping grid, respectively. For satisfying the overlapping condition, we choose $\varepsilon_x/h_x = \varepsilon_y/h_y = 2$, where $\boldsymbol{\varepsilon_x} = (\varepsilon_x, \varepsilon_y)$ and $\boldsymbol{h} = (h_x, h_y, h_{v_x}, h_{v_y})$. Here $\boldsymbol{\varepsilon_x}$ and $\boldsymbol{h}$ are the mesh spacings for the Poisson grid and the remapping grid. For all tests with remapping, we apply remapping every five PIC step. The tests without remapping is equivalent to apply remapping at infinite PIC step.

We use Richardson extrapolation for error estimation. If $\tilde{\boldsymbol{E}}^h$ is the electric field computed with the initial phase-space discretization $\boldsymbol{h}$ and integration step size Δt, and $\tilde{\boldsymbol{E}}^{2h}$ computed with $2\boldsymbol{h}$ and $2\Delta t$, the relative solution error in direction d is defined as

$$e_d^h = |\tilde{E}_d^h - \tilde{E}_d^{2h}|. \tag{4.1}$$

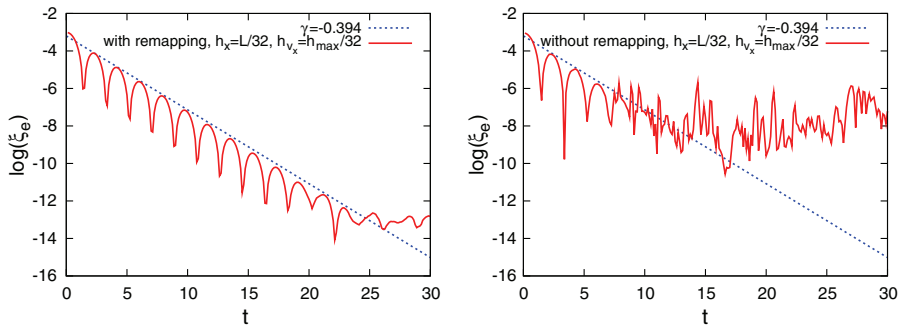
q is the order of the method and is calculated by

$$q = \min_d \log_2 \left(\frac{\|e_d^{2h}\|}{\|e_d^h\|} \right). \tag{4.2}$$

4.1. Linear Landau damping. The initial distribution for linear Landau damping is

$$f_0(x, y, v_x, v_y) = \frac{1}{2\pi} \exp(-(v_x^2 + v_y^2)/2)(1 + \alpha \cos(k_x x) \cos(k_y y)), \tag{4.3}$$

where $\alpha = 0.05$, $k_x = k_y = 0.5$, and $v_{\max} = 6.0$. The physical domain is $(x, y) \in [0, L = 2\pi/k_x] \times [0, L = 2\pi/k_y]$ with periodic boundary conditions. In loading the



(a) with remapping (b) without remapping

FIG. 4.1. *The amplitude of the electric field for two dimensional linear Landau damping problem. Scales* (h_x, h_{v_x}) *above denote the remapping grid mesh spacing at the base level, where* $h_x = h_y$ *and* $h_{v_x} = h_{v_y}$. *With remapping, the computed damping rate is very close to the theoretical value* $\gamma = -0.394$ *until time* 25. *Without remapping, the simulation fails to track the exponential decay after a few damping circles.*

particles from the initial distribution, the particles with function value less than 1.0×10^{-9} are ignored.

In the first test, we are interested in the evolution of the amplitude of the electric field. According to Landau's theory, the electric field is expected to decrease exponentially with damping rate $\gamma = -0.394$. The behavior of exponential decay has been observed by many other authors, mostly simulated by grid methods [27, 16, 10, 11].

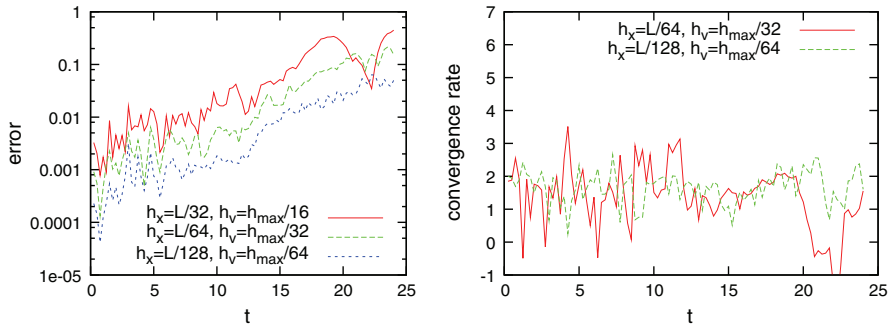
We initialize the problem on two levels of remapping grids with base level at $h_x = h_y = L/32, h_{v_x} = h_{v_y} = v_{\max}/32$. The velocity space is refined on subdomain $\boldsymbol{v} \in [-4.5, 4.5] \times [-4.5, 4.5]$ with a refinement ratio 2. The PIC step size is $dt = 1/8$. We compare the simulation with and without remapping in Figure 4.1. Remapping has successfully captured the damping rate until time 25. The case without remapping fails to track the exponential decay at a very early stage of the simulation, around time 7. Compared with the results from grid-based methods [11], we see that grid-based methods are able to capture the damping rate until time 30. After time 30, the field drops below machine accuracy (around 1.0×10^{-15} in double precision).

4.2. The two stream instability. The initial distribution for the two stream instability is

$$f_0(x, y, v_x, v_y) = \frac{1}{12\pi} \exp(-(v_x^2 + v_y^2)/2)(1 + \alpha \cos(k_x x))(1 + 5v_x^2), \tag{4.4}$$

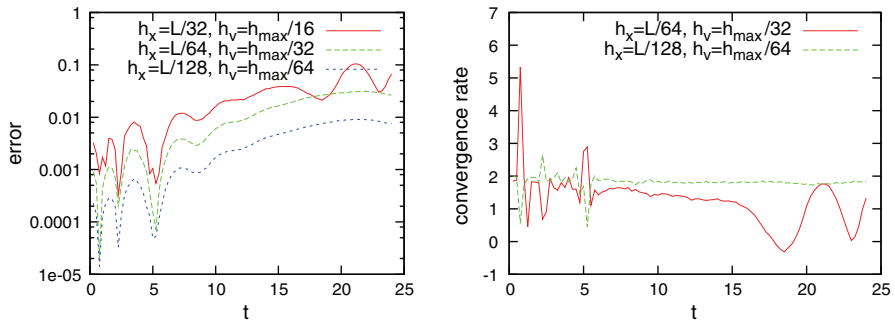
where $\alpha = 0.05$, $k_x = 0.5$, and $v_{\max} = 9.0$. The physical domain is $(x, y) \in [0, L = 2\pi/k_x] \times [0, L = 2\pi/k_y]$ with periodic boundary conditions. In loading the particles from the initial distribution, the particles with function value less than 1.0×10^{-5} are ignored. The velocity space is refined on subdomain $\boldsymbol{v} \in [-4.5, 4.5] \times [-4.5, 4.5]$ with a refinement ratio of 2.

We compare the electric field errors and their convergence rates with and without remapping as before. Figure 4.2 shows the L_∞ norm of the errors at the case without remapping in three different resolutions. The corresponding convergence rates are shown on the right of the error plots. Second-order convergence rates are lost at the early time of the simulation. Comparing with the results with remapping in Figure 4.3, we see that remapping extends the second-order convergence rates to longer times.



(a) L_∞ errors of $\tilde{E}_x$ (b) Convergence rates of the errors on the left

FIG. 4.2. *Error and convergence rate plots for the two stream instability without remapping. Scales* (h_x, h_v) *above denote the remapping grid mesh spacing at the base level, where* $h_x = h_y$ *and* $h_{v_x} = h_{v_y}$*. The PIC step size is* $dt = 1/8$ *at the lowest resolution.* (a) *the* L_∞ *norm of the electric field errors on three different resolutions.* (b) *the convergence rate for the errors on plot* (a).



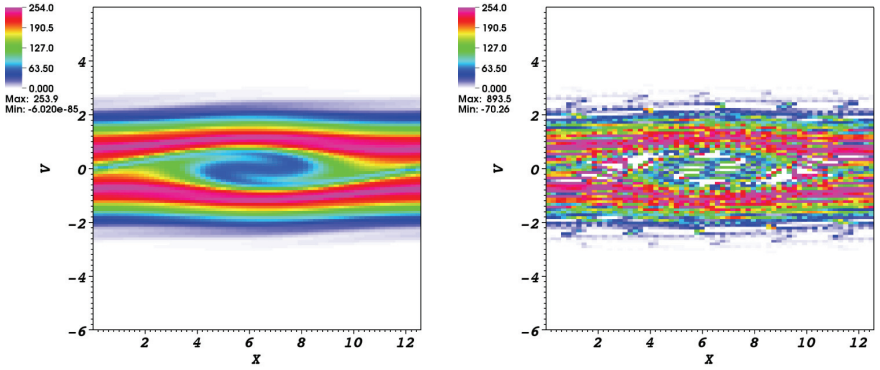
(a) L_∞ errors of $\tilde{E}_x$ (b) Convergence rates of the errors on the left

FIG. 4.3. *Error and convergence rate plots for the two stream instability with remapping. Scales* (h_x, h_v) *above denote the remapping grid mesh spacing at the base level, where* $h_x = h_y$ *and* $h_{v_x} = h_{v_y}$*. The PIC step size is* $dt = 1/8$ *at the lowest resolution.* (a) *the* L_∞ *norm of the electric field errors on three different resolutions.* (b) *the convergence rate for the errors on plot* (a).

We also compare the projected distribution function on plane (x, v_x) at the same instant time $t = 20$ by both methods in Figure 4.4. For visualization purposes, in the case without remapping, we interpolate the particle-based distribution function to a grid in phase space. We see that the classical PIC method results in a noisy solution (see Figure 4.4(b)). Figure 4.4(a) shows the distribution function computed by the PIC method with remapping.

4.3. Semi-Gaussian beam. The paraxial model is an approximation to the steady-state Vlasov–Maxwell equation in three dimensions. The Kapchinky-Vladimirsky (K-V) distribution is a measure solution of the paraxial model. Given an arbitrary initial distribution, we can focus a beam with the same matching forces for the K-V beam using the concept of equivalent beam. Here, we consider an initial semi-Gaussian beam focused by an uniform electric field using the concept of equivalent beam. The model has been considered by many authors [29, 10, 11].

In the test, the beam is composed of ionized potassium. The physical parameters are the following: current $I = 0.2A$, beam velocity $v_b = 0.63 \times 10^6 m/s$, and the radius of the beam $a = 0.02m$. We choose the tune depression $\eta = 1/2$. For the normalization



(a) with remapping at time $t = 20$ (b) without remapping at time $t = 20$

FIG. 4.4. *Comparison of $F(x, v_x)$, the distribution function projected on space (x, v_x), at the same instant of time $t = 20$ with (left) and without remapping (right) for the two stream instability problem. The projected value is $F(x, v_x) = \int_0^L \int_{-\infty}^{\infty} f(x, y, v_x, v_y) dy dv_y$. The grid-based distribution function is obtained by reproducing the particle-based distribution function through a second-order interpolation. We initialize the distribution function on two levels of remapping grids, with base level at $h_x = h_y = L/64$, $h_{v_x} = h_{v_y} = v_{\max}/32$. The grid is refined by factor of 2 in velocity space on subdomain $\boldsymbol{v} \in [-4.5, 4.5] \times [-4.5, 4.5]$. The classical PIC method results in a noisy solution with large errors in maximum. Both numerical noise and errors in maximum are significantly reduced by using remapping. The negative minimum in the case without remapping is a superficial effect due to project four dimensional data to two dimensional using a high-order interpolation for visualization purpose.*

of the paraxial model, we refer to the work of Filbet and Sonnendrücker [17]. We use normalization parameters $(x_0, v_0) = (a, \frac{\epsilon_x v_h}{2a})$. This results in the normalized Poisson system

$$-\triangle \phi = \int_{\mathbb{R}^2} f d\boldsymbol{v}, \quad -\nabla \phi = \boldsymbol{E}, \tag{4.5}$$

with initial semi-Gaussian distribution

$$f_0(x, y, v_x, v_y) = \begin{cases} \frac{4(1-\eta^2)}{\pi \eta^2} \exp(-(v_x^2 + v_y^2)/2), & x^2 + y^2 \leq 1, \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

and the external matching field

$$E^e(x, y, t) = -\frac{4}{\eta^2}(x\boldsymbol{e}_x + y\boldsymbol{e}_y). \tag{4.7}$$

As for the numerical parameters, we choose $(h_x = |L_x|/128, h_y = |L_y|/128)$ and $(h_{v_x} = v_{\max}/128, h_{v_y} = v_{\max}/128)$, where $(L_x, L_y) = (-10, 10)$ and $v_{\max} = 10$. The PIC time step is $dt = 0.00052925$.

Figure 4.5 shows the projection of the distribution function on planes (x, v_x) with and without remapping, respectively. The simulation with remapping gives a well-resolved result which preserves the positivity of the distribution function. Meanwhile, we show the root mean square (RMS) quantities of the semi-Gaussian beam in Figure 4.6. Although the RMS quantities of the semi-Gaussian beam are oscillatory, they remain close to the quantities of the associated K-V beam and they converge as the resolution increases. As mentioned by the other authors [10], the oscillatory behavior is due to the fact that the semi-Gaussian is not exactly a steady state distribution.

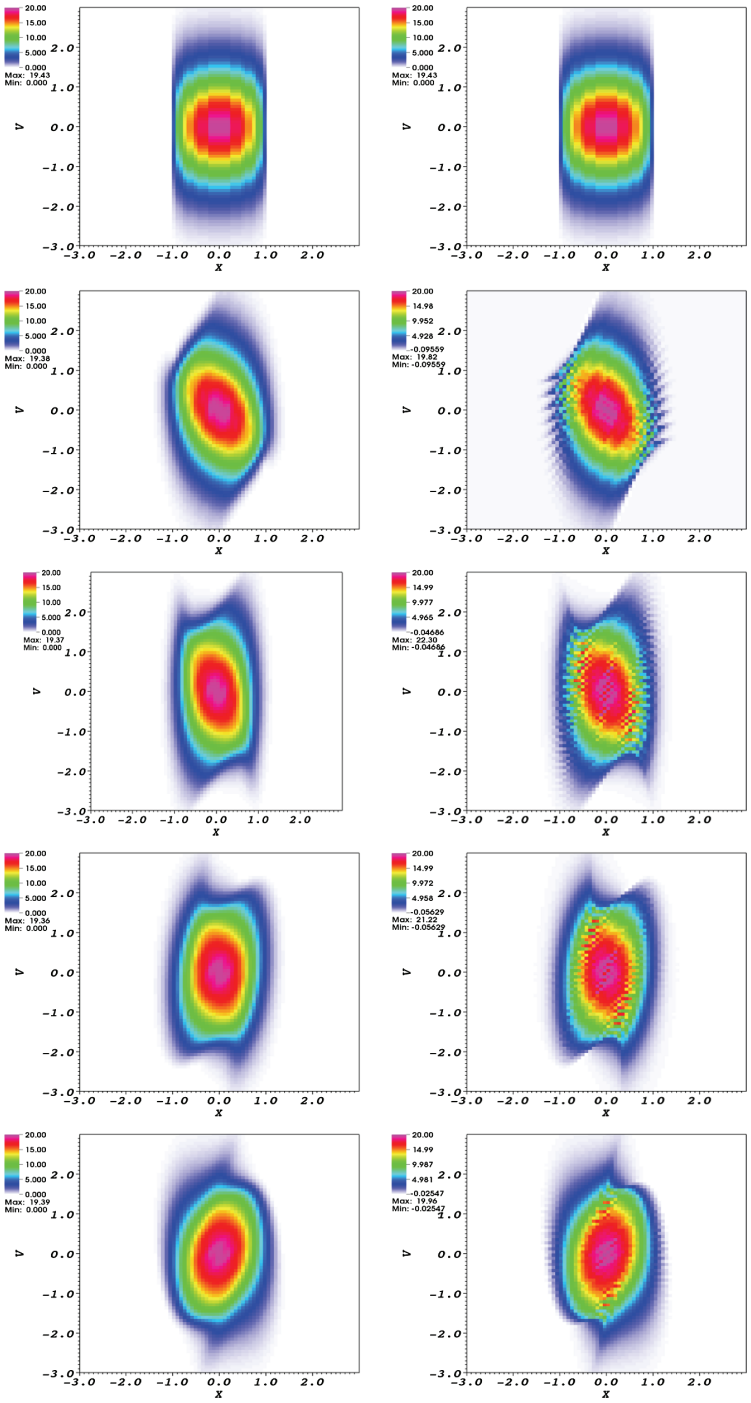
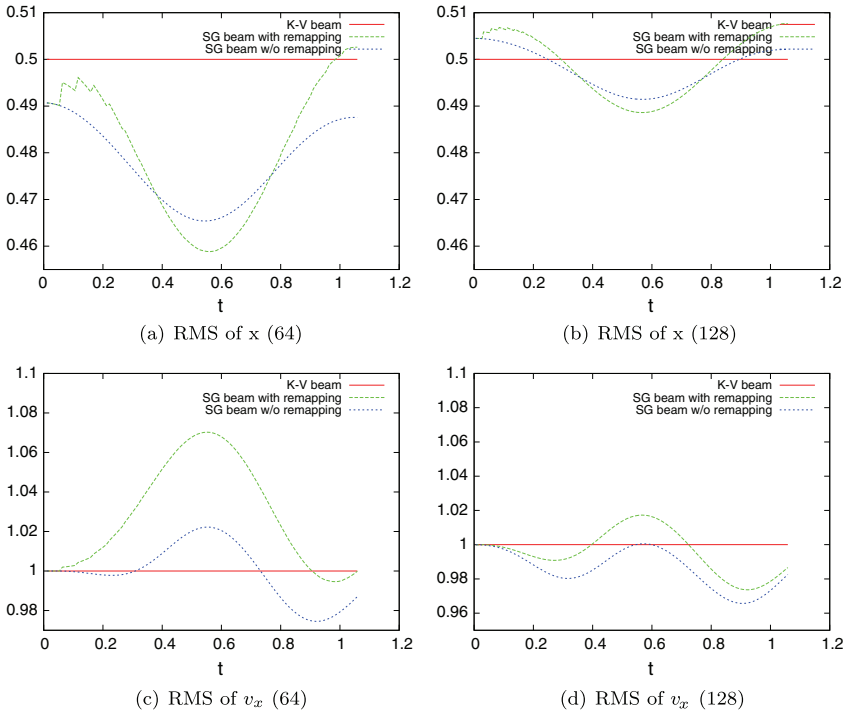


FIG. 4.5. *Time evolution of the projection of the distribution function on* (x, v_x) *at time* $t = 0, 0.3176, 0.5239, 0.7410, 0.9527$ *(from top to bottom). The projected value is* $F(x, v_x) = \int_0^L \int_{-\infty}^{\infty} f(x, y, v_x, v_y) dy dv_y$. *The grid-based distribution function is obtained by reproducing the particle-based distribution function through a second-order interpolation. The columns on the left and on the right are simulations running with and without remapping, respectively. The classical PIC method results in a noisy solution with large errors in maximum.*



(a) RMS of x (64) (b) RMS of x (128)

(c) RMS of v_x (64) (d) RMS of v_x (128)

FIG. 4.6. *Comparison of RMS quantities of semi-Gaussian beam with remapping, without remapping, and its equivalent K-V beam. We show the quantities in two different resolutions (*64 *versus* 128 *grid size in each dimension). Although the values are oscillatory, they remains close to the equivalent K-V beam and they converge to the equivalent K-V beam as the resolution increases.*

4.4. Performance study. We carry out performance studies of the algorithm on the Cray XE6 "Hopper" machine, located at the National Energy Research Scientific Computing Center (NERSC). In the studies, we apply remapping on a uniform grid in phase space. Positivity is not enforced in these scaling computations.

Figure 4.7 shows the results of the parallel speedup (strong scaling) running on 4 to 64 MPI processes for the linear Landau damping problem in section 4.1. The test starts from 4 MPI processes with each of them conducting calculation on 8×8 nonoverlapping patches in the Poisson grid. As the number of MPI processes goes up to 64, the patch size reduces to 2×2.

Table 4.1 shows a weak scaling study with respect to Poisson grid of the same problem. In the tests, we fix the patch size to be 4×4 and the remapping grid size in velocity dimensions to be 128×128. We increase the Poisson grid size as we increase the number of MPI processes proportionally. The remapping grid size in physical dimensions is increased accordingly, since we choose $\varepsilon_x/h_x = \varepsilon_y/h_y = 2$ in all simulations. Ideally, the computational time would be flat. We see an almost flat result for the weak scaling test.

Table 4.2 shows a weak scaling study with respect to phase-space remapping grid of the same problem. The patch size is fixed to be 4×4 in the tests. We increase the remapping grid size as we increase the number of MPI processes, but the number of processes is increased proportionally to the grid size in phyiscal dimensions (as well as the Poisson grid size) only. We see that the wall clock time and memory usage increase by around the same factor as the increase of the grid size in velocity

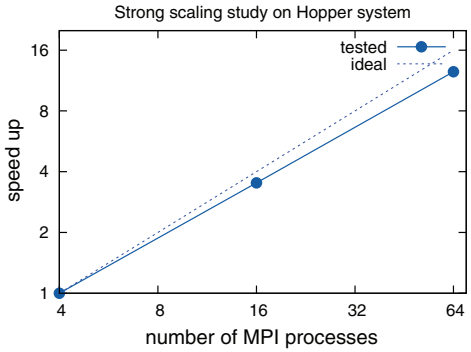


FIG. 4.7. *Parallel speedup results for the two dimensional linear Landau damping problem in section 4.1. We start with 8×8 nonoverlapped patches. We decrease the patch size as we increase the number of MPI processes proportionally.*

TABLE 4.1

Comparison of wall clock time of the weak scaling study with respect to Poisson grid. The number of MPI processes is increased proportionally to the Poisson grid. The simulation is running for one remapping circle including ten PIC integration steps with a single remapping at step 5.

The number of MPI processes	16	64	256
Poisson grid size	16^2	32^2	64^2
Remapping grid size	$32^2 \times 128^2$	$64^2 \times 128^2$	$128^2 \times 128^2$
Total wall clock time(s)/factor	196.99/1.00	217.82/1.11	226.12/1.15

TABLE 4.2

Comparison of wall clock time and memory usage of the weak scaling study with respect to remapping grid. The number of MPI processes is increased proportionally to the Poisson grid only. The simulation is running for one remapping circle including ten PIC integration steps with a single remapping at step 5. It shows that the wall clock time and memory usage increase by around the same factor as the increase of the remapping grid in velocity dimensions.

The number of MPI processes	16	64	256
Poisson grid size	16^2	32^2	64^2
Remapping grid size	$32^2 \times 32^2$	$64^2 \times 64^2$	$128^2 \times 128^2$
Peak memory usage in each PE/factor	30M/1.00	121M/4.03	485M/16.17
Total wall clock time(s)/factor	13.45/1.00	58.34/4.34	226.12/16.81

dimensions. As discussed earlier, this shows the potential limitation of the physical domain decomposition in phase-space scaling. To solve this issue, two alternative solutions are discussed in section 3.

Finally, we would like to mention that the computation time in the tables is used as an illustration of scalability of the parallel algorithm. The remapped PIC solver is expected to be much faster in the future since the current implementation is not optimized. For example, the current implementation uses a linked list data structure for particle representation in each cell. Although linked-list potentially makes poor use of caches when streaming the particle list, it solves the random memory access issue in gather and scatter operations and shows some benefit in density accumulation and field interpolation in physical space. Unfortunately, linked-list has no benefit in phase-space remapping where each cell has approximately only one particle. An array data structure with periodically binning is preferred here to better utilize caches. In addition, we will hope to see more speedup by integrating a fully optimized mesh refinement implementation.

5. Conclusion and future work. In this paper, we have presented the adaptive remapped PIC method to the high dimensional Vlasov equation and demonstrated in linear Landau damping, the two stream instability, and the beam propagation problems. We have shown that the algorithm reduces the numerical noise significantly and have proved the effectiveness of the method for high dimensional Vlasov–Poisson systems. The parallel performance of the implementation has been studied through strong and weak scaling experiments. The current parallel domain decomposition has potential to scale to a large number of processors in device scaling. We have also shown the limitation of the physical domain decomposition in phase-space grid scaling. As the extension of the current research, two more general parallel decomposition algorithms for particle-in-cell methods are discussed, with the potential to scale well on both device scaling and phase-space grid size scaling.

REFERENCES

[1] C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation*, Taylor & Francis, New York, 1991.

[2] A. K. Chaniotis, D. Poulikakos, and P. Koumoutsakos, *Remeshed smoothed particle hydrodynamics for the simution of viscous and heat conducting flows*, J. Comput. Phys., 182 (2002), pp. 67–90.

[3] C. Z. Cheng and G. Knorr, *The integration of the Vlasov equation in configuration space*, J. Comput. Phys., 22 (1976), pp. 2330–2351.

[4] Y. Chen and S. E. Parker, *Coarse-graining phase space in Δf particle-in-cell simulations*, Phys. Plasmas, 14 (2007), p. 082301.

[5] I.-L. Chern and P. Colella, *A Conservative Front Tracking Method for Hyperbolic Conservation Laws*, Technical report, Lawrence Livermore National Laboratory, Livermore, CA, 1987.

[6] P. Colella, M. R. Dorr, J. A. F. Hittinger, and D. F. Martin, *High-order finite-volume methods in mapped coordinates*, J. Comput. Phys., 230 (2011), pp. 2952–2976.

[7] P. Colella and P. C. Norgaard, *Controlling self-force errors at refinement boundaries for AMR-PIC*, J. Comput. Phys., 229 (2010), pp. 947–957.

[8] G.-H. Cottet and P. D. Koumoutsakos, *Vortex Methods: Theory and Practice*, Cambridge University Press, Cambridge, UK, 2000.

[9] G.-H. Cottet and P. A. Raviart, *Particle methods for the one-dimensional Vlasov–Poisson equations*, SIAM J. Numer. Anal., 21 (1984), pp. 52–76.

[10] N. Crouseilles, M. Gutnic, G. Latu, and E. Sonnendrucker, *Comparision of two Eulerian solvers for the four-dimensional Vlasov equation: Part* II, Commun. Nonlinear Sci. Numer. Simul., 13 (2008), pp. 94–99.

[11] N. Crouseilles, G. Latu, and E. Sonnendrücker, *A parallel Vlasov solver based on local cubic spline interpolation on patches*, J. Comput. Phys., 228 (2009), pp. 1429–1446.

[12] J. Denavit, *Numerical simulation of plasma with periodic smoothing in phase space*, J. Comput. Phys, 9 (1972), pp. 75–98.

[13] A. M. Dimits and W. W. Lee, *Partially linearized algorithms in gyrokinetic particle simulation*, J. Comput. Phys., 107 (1993), pp. 309–323.

[14] S. Ethier, M. Adams, J. Carter, and L. Oliker, *Petascale parallelization of the gyrokinetic toroidal code*, in Proceedings of VECPAR'10 9th International Meeting on High Performance Computing for Computational Science, 6449, Springer, Lecture Notes in Computer Science, 2010.

[15] E. Fijalkow, *A numerical solution to the Vlasov equation*, Comput. Phys. Comm., 116 (1999), pp. 319–328.

[16] F. Filbet, E. Sonnendrücker, and P. Bertrand, *Conservative numerical schemes for the Vlasov equation*, J. Comput. Phys., 172 (2001), pp. 166–187.

[17] F. Filbet and E. Sonnendrucker, *Modeling and numerical simulation of space charged dominated beams in the paraxial approximation*, Math. Models Methods Appl. Sci., 16 (2006), pp. 763–791.

[18] J. Hilditch and P. Colella, *A projection method for low Mach number fast chemistry reacting flow*, in Proceedings of the AIAA Aerospace Sciences Meeeting, The American Institute of Aeronautics and Astronautics, Reno, NV, 1997.

[19] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, McGraw-Hill, New York, 1981.
[20] R. A. James, *The solution of Poisson's equation for isolated source distributions*, J. Comput. Phys., 25 (1977), pp. 71–93.
[21] K. Lackner, *Computation of idea MHD equilibria*, Comput. Phys. Comm., 12 (1976), pp. 33–44.
[22] W. W. Lee, T. G. Jenkins, and S. Ethier, *A generalized weight-based particle-in-cell simulation scheme*, Comput. Phys. Comm, 182 (2011), pp. 564–569.
[23] Z. Lin, T. S. Hahm, W. W. Lee, W. M. Tang, and R. B. White, *Turbulent transport reduction by zonal flows: Massively parallel simulations*, Science, 281 (1998), pp. 1835–1837.
[24] P. McCorquodale, P. Colella, G. T. Balls, and S. B. Baden, *A scalable parallel Poisson solver in three dimensions with infinite-domain boundary conditions*, In the Proceedings of the 2005 International Conference on Parallel Processing Workshops (ICPPW'05), IEEE Computer Society, 2005, pp. 163–172.
[25] P. McCorquodale, P. Colella, G. T. Balls, and S. B. Baden, *A local corrections algorithm for solving Poisson's equation in three dimensions*, Commun. Appl. Math. Comput. Sci., 2 (2007), pp. 57–81.
[26] J. J. Monaghan, *Particle methods for hydrodynamics*, Comput. Phys. Rep., 3 (1985), pp. 71–124.
[27] T. Nakamura and T. Yabe, *Cubic interpolated propagation scheme for solving the hyper-dimensional Vlasov-Poisson equation in phase space*, Comput. Phys. Commun., 120 (1999), pp. 122–154.
[28] S. Parker, W. Lee, and R. Santoro, *Gyrokinetic simulation of itg driven turbulence in 3d toroidal geometry*, Phys. Rev. Lett., 71 (1993), pp. 2042–2045.
[29] E. Sonnendrücker, F. Filbet, A. Friedman, E. Oudet, and J.-L. Vay, *Vlasov simulations of beams with a moving grid*, Comput. Phys. Comm., 164 (2004), pp. 390–395.
[30] E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo, *The semi-Lagrangian method for the numerical resolution of Vlasov equations*, J. Comput. Phys., 149 (1998), pp. 201–220.
[31] S. Vadlamani, S. E. Parker, Y. Cheng, and C. Kim, *The particle-continuum method: An algorithmic unification of particle-in-cell and continuum methods*, Comput. Phys. Comm., 164 (2004), pp. 209–213.
[32] B. Wang, G. H. Miller, and P. Colella, *A particle-in-cell method with adaptive phase-space remapping for kinetic plasmas*, SIAM J. Sci. Comput., 33 (2011), pp. 3509–3537.
[33] M. F. Adams, S. Ethier, and N. Wichmann, *Performance of particle in cell methods on highly concurrent computational architectures*, J. Phys. Conf. Ser., 78 (2007), p. 012001.