Journal of Computational Physics 365 (2018) 362–375

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Semi-Lagrangian particle methods for high-dimensional Vlasov–Poisson systems

Georges-Henri Cottet

Univ. Grenoble Alpes and CNRS, Laboratoire Jean Kuntzmann, Grenoble, France

ARTICLE INFO

Article history:
Received 8 September 2017
Received in revised form 15 March 2018
Accepted 29 March 2018
Available online 4 April 2018

Keywords:
Particle methods
Semi-Lagrangian methods
Vlasov–Poisson equations

ABSTRACT

This paper deals with the implementation of high order semi-Lagrangian particle methods to handle high dimensional Vlasov–Poisson systems. It is based on recent developments in the numerical analysis of particle methods and the paper focuses on specific algorithmic features to handle large dimensions. The methods are tested with uniform particle distributions in particular against a recent multi-resolution wavelet based method on a 4D plasma instability case and a 6D gravitational case. Conservation properties, accuracy and computational costs are monitored. The excellent accuracy/cost trade-off shown by the method opens new perspective for accurate simulations of high dimensional kinetic equations by particle methods.

1. Introduction

The Vlasov–Poisson system describes the dynamics of particles moving in a self-consistent electric field. The unknown is the distribution function $f = f(\Xi, \Psi, t)$ which gives the probability for particles to occupy a given position Ξ with velocity Ψ at time t. In the most complex case, one thus has to deal with 6-dimensional phase-spaces. High resolution simulations may therefore require a tremendous computational effort.

Particle methods [12] have long been the method of choice to approximate this system, as they allow to restrain the computations in the support of f. Numerical particles mimic the dynamics of physical particles in the phase space. In the most popular implementation of particle methods, Particle-In-Cell methods assign values of f on a three-dimensional grid to obtain charge density which are used to compute the electric field on this grid. The electric field is in turn interpolated on particle locations to advance the particles in the phase-space.

One drawback of particle methods is the inherent noise that affects their accuracy. This noise results from the chaotic behavior of particles and makes it in general necessary to average particle quantities on a large number of particles, which has a strong impact on their computational cost.

On the other hand, the event of large supercomputers and parallel algorithms has made possible the use of Eulerian methods to discretize the Vlasov–Poisson equation with a better accuracy. Two important developments in this field have in particular been made in the recent years: semi-Lagrangian and multi-resolution methods.

Semi-Lagrangian methods [19] are grid-based methods well adapted to the fact that the Vlasov equation are advection-driven. At each grid point, particle trajectories are traced back in the phase space and values on the solution are updated by interpolating the values of the distribution function at the foot of the trajectory. Recent developments include the derivation

E-mail address: georges-henri.cottet@univ-grenoble-alpes.fr.

https://doi.org/10.1016/j.jcp.2018.03.042
0021-9991/

of high order [5] and conservative [6] methods. One drawback of the method still lies on its computational complexity, and its use, to our knowledge, has so far been limited to low dimensional or small time simulations.

Multi-resolution methods are a rather attractive approach to address the dimension issue of Vlasov–Poisson systems due to the ability of these methods to concentrate their effort on limited areas of the computational domain. In [11] AMR techniques were devised and applied to two-dimensional problems. Very recently, in [7] a wavelet-based approach was developed for the Vlasov–Poisson equations and applied with success to 4 and 6 dimensional systems. The latter method gives rigorous and flexible criteria to define the multi-resolution hierarchy grids. All these methods are however based on Eulerian discretizations and they do not have the attractive robustness that Lagrangian or semi-Lagrangian methods offer for the underlying transport equations. Note that multi-resolution semi-Lagrangian methods have also been devised in the context of discontinuous Galerkin methods for the Vlasov Poisson equations in [3] but their application has been so far restricted to 2D problems.

Going back to particle methods, recent development in the fields of transport equations and computational fluid dynamics have been made to overcome the accuracy limitations of these methods. Particle remeshing on a regular grid, in particular, has been found to be an efficient way to avoid numerical noise in flows submitted to high shear [13], while essentially preserving the localization properties of particle methods. Remeshing particles at each time-step yields a class of forward, conservative, semi-Lagrangian methods which can be analyzed as such [4]. The accuracy of these methods can be analyzed in terms of the moment and regularity properties of the remeshing kernel. The localization properties of semi-Lagrangian particle methods can be reinforced by using Adaptive Mesh Refinement [1] or wavelet-based multi-resolution [2, 17]. To our knowledge, remeshed particle methods have not been applied to the Vlasov–Poisson equations, with the notable exception of [15]. In this reference the method is applied with success to the two-dimensional Vlasov–Poisson system (one space and one velocity dimensions). The influence of the remeshing kernel in the overall accuracy of the method for the Landau damping is discussed. The purpose of the present work is to investigate the capabilities of semi-Lagrangian particle methods, both in terms of accuracy and computational complexity, to handle 4D and 6D Vlasov systems. The challenge is to determine splitting and remeshing strategies which are tractable in high dimensions. To investigate these strategies we will restrict ourselves to single core implementations of the method using an underlying uniform grid to remesh particles and we will use the same benchmarks as in [7].

The outline of the paper is as follows. In section 2 we recall our previous work on semi-Lagrangian particle methods. In section 3 we define our splitting and remeshing strategy in the case of multi-dimensional Vlasov–Poisson system. In section 4 we discuss the application of the method on our benchmarks. Section 5 is devoted to concluding remarks and indication of future works.

2. Semi-Lagrangian particle methods for transport equations

As we will see in the next sections, an efficient implementation of semi-Lagrangian particle methods is based on a directional splitting where particles are successively pushed and remeshed along the directions of the phase space. We can thus focus on the one-dimensional transport equation to describe the method and discuss its convergence properties.

Let us consider the following 1D model linear advection problem for the unknown function f:

$$f_t + (a\,f)_x = 0, x \in \mathbf{R}, t > 0, \tag{1}$$

where a is a given smooth velocity field. A particle method where particles are remeshed at each time step can be recast as

$$f_i^{n+1} = \sum f_j^n \,\Gamma\left(\frac{x_j^{n+1} - x_i}{\Delta x}\right), i \in \mathbf{Z}, n \geq 0. \tag{2}$$

In the above equation Δx is the grid size on which particles are remeshed (assuming a regular grid), x_j are the grid points and Γ is the remeshing interpolating kernel. x_j^{n+1} is the result of the advection at time t_{n+1} of the particle located at x_j at time t_n.

To satisfy the conservation of successive moments of the distribution f, starting with the conservation of mass, the remeshing kernel Γ must satisfy moment properties that can be written as

$$\sum_{k\in\mathbf{Z}} (x-k)^\alpha \Gamma(x-k) = \begin{cases} 1 & \text{if } \alpha = 0 \\ 0 & \text{if } 1 \leq \alpha \leq p \end{cases}, x \in \mathbf{R}, \tag{3}$$

for a given value of $p \geq 1$. An additional requirement is that Γ is globally in $W^{r+1,\infty}$ (which means that all his derivatives up to order $r+1$ are bounded), is infinitely differentiable in each integer interval (in practice Γ is a polynomial in these intervals), and satisfies the interpolation property: $\Gamma(i-j) = \delta_{ij}$. In the simple case of an Euler explicit scheme to advect particles, $x_j^{n+1} = x_j + a(x_j, t_n)\Delta t$ and when the time step satisfies the condition

$$\Delta t < \left[\sup_{x\in\mathbb{R}} |a'(x)|\right]^{-1}, \tag{4}$$

where a' denotes the spatial derivative of a, one can prove [4] that the consistency error of the semi-Lagrangian method is bounded by $O(\Delta t + \Delta x^\beta)$ where $\beta = \min(p, r)$. Using higher order Runge–Kutta schemes increase the time accuracy, as expected. Moreover, at least for kernels of order up to 4, under appropriate decay properties for the kernel Γ one can prove the stability of the method under the sole assumption (4).

A particular case, which will apply in the specific case of Vlasov–Poisson equations, where 'super convergence' can be observed, is when, after an advection stage, each cell of size Δx contains exactly 1 particle (in other words when particle distortion along the line is limited). In that case the regularity of the kernel 'across cells' is no longer necessary and the order of convergence β above is p instead of $\min(p, r)$.

Kernels corresponding to specific values of p and r as described above are denoted by $\Lambda_{p,r}$. The following formulas give the expression of the kernels $\Lambda_{4,2}$ and $\Lambda_{8,4}$ which will be used in the sequel:

$$\Lambda_{4,2}(x) = \begin{cases} 1 - \frac{5}{4}|x|^2 - \frac{35}{12}|x|^3 + \frac{21}{4}|x|^4 - \frac{25}{12}|x|^5 & 0 \leqslant |x| < 1 \\ -4 + \frac{75}{4}|x| - \frac{245}{8}|x|^2 + \frac{545}{24}|x|^3 - \frac{63}{8}|x|^4 + \frac{25}{24}|x|^5 & 1 \leqslant |x| < 2 \\ 18 - \frac{153}{4}|x| + \frac{255}{8}|x|^2 - \frac{313}{24}|x|^3 + \frac{21}{8}|x|^4 - \frac{5}{24}|x|^5 & 2 \leqslant |x| < 3 \\ 0 & 3 \leqslant |x|, \end{cases} \tag{5}$$

$$\Lambda_{8,4}(x) = \begin{cases} 1 - \frac{205}{144}x^2 + \frac{91}{192}x^4 - \frac{6181}{320}x^5 + \frac{6337}{96}x^6 - \frac{2745}{32}x^7 + \frac{28909}{576}x^8 - \frac{3569}{320}x^9 & 0 \leqslant |x| < 1 \\ -154 + \frac{12757}{12}x - \frac{230123}{72}x^2 + \frac{264481}{48}x^3 - \frac{576499}{96}x^4 + \frac{686147}{160}x^5 & \\ \qquad - \frac{96277}{48}x^6 + \frac{14221}{24}x^7 - \frac{28909}{288}x^8 + \frac{3569}{480}x^9 & 1 \leqslant |x| < 2 \\ \frac{68776}{7} - \frac{1038011}{28}x + \frac{31157515}{504}x^2 - \frac{956669}{16}x^3 + \frac{3548009}{96}x^4 - \frac{2422263}{160}x^5 & \\ \qquad + \frac{197255}{48}x^6 - \frac{19959}{28}x^7 + \frac{144545}{2016}x^8 - \frac{3569}{1120}x^9 & 2 \leqslant |x| < 3 \\ -56375 + \frac{8314091}{56}x - \frac{49901303}{288}x^2 + \frac{3763529}{32}x^3 - \frac{19648027}{384}x^4 + \frac{9469163}{640}x^5 & \\ \qquad - \frac{545977}{192}x^6 + \frac{156927}{448}x^7 - \frac{28909}{1152}x^8 + \frac{3569}{4480}x^9 & 3 \leqslant |x| < 4 \\ \frac{439375}{7} - \frac{64188125}{504}x + \frac{231125375}{2016}x^2 - \frac{17306975}{288}x^3 + \frac{7761805}{384}x^4 - \frac{2895587}{640}x^5 & \\ \qquad + \frac{129391}{192}x^6 - \frac{259715}{4032}x^7 + \frac{28909}{8064}x^8 - \frac{3569}{40320}x^9 & 4 \leqslant |x| < 5 \\ 0 & 5 \leqslant |x|. \end{cases} \tag{6}$$

The kernel $\Lambda_{4,2}$ (resp. $\Lambda_{8,4}$) involves a stencil made of 6 grid points (resp. 10 grid points). The benefit of using directional splitting is not only to reduce the analysis to the 1D case but also to minimize the cost when high order kernels, with large stencils, are used. For instance, if a first order splitting is used in 3 dimensions, the cost of the method for N particles with the $\Lambda_{4,2}$ (resp. $\Lambda_{8,4}$) kernels will scale as $O(18N)$ (resp. $O(30N)$) instead of $O(216N)$ (resp. $O(1000N)$) if a tensor product formula was used.

3. Algorithm for the Vlasov–Poisson equations

As already mentioned, remeshed particle methods have already been applied to 1D/1D Vlasov Poisson system in [15]. In this reference, the gain offered by fourth order methods has been demonstrated on the analysis of the one-dimensional Landau damping. In this section and the following we present an implementation of semi-Lagrangian particle methods for the 6D Vlasov–Poisson system, where we in particular emphasize the role of directional splitting and link-list algorithms to reduce the computational complexity and we further investigate the influence of high order kernels to improve the accuracy of the methods. We denote by x, y, z and u, v, w the space and velocity axis, respectively.

Particle remeshing which is essential for accuracy control also results in the need to introduce grid arrays. On the one hand, using six dimensional arrays is not affordable, except for very coarse resolutions. On the other hand using directional splitting, as suggested in the previous section, would in principle only require one-dimensional arrays to carry particle quantities but each line would have to be labelled with a five-dimensional array, which is also intractable.

A reasonable trade-off between splitting and array dimensions is to alternate motion/remeshing of particles in three dimensional spaces. The natural choice is to move/remesh particles in x, y, z and u, v, w spaces alternately.

In the sequel, at the end of each remeshing step and for each (x, y, z) or (u, v, w) grid values, we will call (x, y, z)-space (resp. (u, v, w)-space) a 3D array of particles with given x, y, z locations (resp. (u, v, w) velocities) on the grid. To identify particles in such paces, a link-list algorithm is used. Link list algorithms have long been used in grid-free particle methods, either to compute velocities in tree codes or to compute diffusion through Particle Strength Exchange algorithms. In the present case, the computation of densities can be made inside the link list algorithm used to label (x, y, z)-spaces by accumulating values of f at a given (x, y, z)-location while sweeping over all particles. The results of each of these link-list algorithms are (for the x, y, z linklist to fix ideas):

- two three-dimensional arrays: a pointer which goes from the grid values to the index of the particle, and the number of particles in each (x, y, z)-space,
- two arrays with dimension the total number of particles to specify indices of particles of the planes in the original list of particles.

Once particles are assigned in given three-dimensional spaces, they can be advected along the corresponding directions.

Assuming a first order time-splitting, the algorithm thus goes along the following steps, for each time-iteration:

1. create link-list for (u, v, w)-spaces
2. in each (u, v, w)-space push particles in the x, y, z directions using the allocated u, v, w values
3. remesh particles and create fresh particles whenever the value of f exceeds a given threshold
4. create link-list for (x, y, z)-spaces and compute the density values
5. collect all densities and compute the electric field through a 3D Poisson solver
6. in each (x, y, z)-space push particles in the u, v, w directions using the electric field at the corresponding x, y, z location
7. remesh particles and create fresh particles whenever the value of f exceeds a given threshold.

In the sequel we use a second order time-splitting method, where the step 1 to 3 above are made over half a time-step and repeated at the end of the iteration.

As mentioned in section 2, using high order remeshing kernels can lead to high computational cost if a 3D tensor-product remeshing formula is used. For this reason, steps 2–3 and 6–7 above are in turn split into one-dimensional advection-remeshing steps. To do so, one could again use link-list algorithms inside each three dimensional space to assign particles to lines. An alternative, more direct, solution is to successively advect the three-dimensional spaces by freezing 2 out of the 3 indices. Note that when using this additional splitting in, say, a given (x, y, z)-space, the link-lists are used only in the first direction. After advection-remeshing along the first direction, arrays of particles are created in the given (x, y, z)-space and used, instead of the link-lists, in the subsequent stages of the splitting in that specific space. As a result, advection/remeshing in the 2 subsequent directions of the (x, y, z)-space are faster (typically by a factor 2 in our implementation).

A few remarks on the cost and accuracy of this algorithm are now in order.

From the computational point of view, at the end of the remeshing steps when new particle values are assigned from the grid values, in order to limit the cost of this step to the number of particles and not to 6-dimensional full grids, it is important to avoid sweeping over the whole three-dimensional spaces. To do so, nearest grid points assigned to particles after particle motion are identified in stages 2 and 6 above, and when one has to assign grid values to fresh particles, one only considers these grid points and neighboring grid points (the number of which depends on the size of the support of the remeshing kernel).

From the accuracy point of view, one can make the following important observations:

1) In each (u, v, w)-space the velocity in the (x, y, z) directions is constant. Therefore pushing particles with a simple first-order Euler scheme gives exact solution of the particle advection (and similarly in the (x, y, z)-spaces). In other words, the time accuracy of the overall scheme is only dictated by the $\boldsymbol{\Xi}/\boldsymbol{\Psi}$ splitting algorithm.
2) For the same reason, one-dimensional splitting within each 3D space does not introduce further splitting error.
3) In section 2 above we have indicated that the regularity of the kernels is a limiting factor in the overall accuracy of the method, except when the advection of particles maintains exactly one particle in each grid cell. We are precisely in this particular case, since, for each splitting sub-step, on each line particles are advected by a constant velocity. In other words advection-remeshing with the kernel $\Lambda_{2,1}$ is second order, fourth order with $\Lambda_{4,2}$, and so on.
4) For the same reason, the Lagrangian CFL condition (4) does not give in the present implementation any limitation for the time-step.
5) Finally, we recall that remeshing kernels which provide second order approximations, or beyond, are not positive. In the steps 3 and 7 above, one can apply the threshold either on the absolute value of f, in which case negative values of f can appear, or to the value of f itself, which ensures the positivity of f but may compromise the conservativity of the algorithm. We will comment on this specific aspect in the sequel.

The stability property just mentioned in 4) is in principle desirable, since it ensures unconditional stability of the method. However it has the drawback of not giving a clear criterion to choose the time-step, in contrast with semi-Lagrangian particle methods for flow simulations where the time-step is in general defined as a function of the maximum amount of shear in the flow. One way to determine the time-step value would be to control the accuracy of the $\boldsymbol{\Xi}/\boldsymbol{\Psi}$ splitting, which, as we have already noticed, controls the overall time-accuracy of the method. One can easily check that the accuracy of this splitting is given by the spatial derivatives of the velocity field in the phase space. In the particular case of the Vlasov–Poisson equations, these derivatives reduce to 1, on the one hand, and the spatial derivatives of the electric field on the other hand. Derivatives of the electric field are in turn bounded in terms of the density. As a result, one quantity to monitor and that can be used to adapt the time-step value is the maximum value of the density. In practice we have chosen fixed time-step values which were a fraction of the maximum density. As we will see, for the grid-size used in our simulations these values eventually correspond to large CFL numbers.

Let us finally comment on the memory foot-print and computational complexity of the method. The memory load is directly given by the number of particles. More precisely, for a six-dimensional phase space the algorithm outlined above requires in our implementation 16 arrays of size the number of particles:

- 7 main arrays, for the 3 axis and velocity directions and for the distribution function,
- 7 additional arrays which are used to temporary store particle locations and velocities during the push and remesh algorithms,
- 2 arrays to store particles addresses computed in the link list algorithms and used to push and remesh particles in the 3D spaces.

On top of these particles arrays, the algorithm requires several 3D arrays, but with a memory size which is a small fraction of that of the particle arrays. In most of our simulations the number of particles was of the order of 10^8 for a number of grid points in each 3D space of the order of 10^6. In the next section we will show the computational time involved at each stage of the algorithm.

4. Four and six dimensional benchmarks

In this section we focus on two cases borrowed from [7] and which illustrates the capabilities and limits of the method in single core implementations: a 4D plasma instability and a 6D gravitational case. All our simulations were performed on an Intel Xeon E5-2640 core running at 2.5 GHz.

4.1. Four dimensional two-beams instability

In this section we consider the Fijalkow Two Beams instability [9]. Following [7] the initial condition is given by the following formula

$$f_0(x, y, u, v) = \frac{7}{4\pi} \exp\left(-\frac{u^2 + 4v^2}{8}\right) \sin^2(\frac{u}{3}) (1 + 0.05\cos(0.3x)), \tag{7}$$

and the computational box is the rectangle

$$\Omega = [-\frac{10\pi}{3}, \frac{10\pi}{3}]^2 \times [-3\pi, 3\pi]^2.$$

For this case, we used the remeshing kernel $\Lambda_{4,2}$ given by (5), which conserves the four first moments of the distribution function (and, as a result, is not positive) and which is twice differentiable. We recall that, although in principle this remeshing kernel leads to a second order transport scheme, in the particular case of the Vlasov Poisson equation with directional splitting it yields fourth order spatial accuracy.

The cut-off value to create particles at the end of the remeshing step was taken equal to 10^{-5} and it was applied to the value of f and not its absolute values. In particular this has the effect of discarding any negative values which could result from the remeshing kernel.

In the first experiment we monitor the conservation properties of the method and we use two sets of resolutions: a coarse grid with $N_c = 64^4$ grid points and a finer grid using $N_f = 128^4$ grid points. We compare our results to the wavelet-based multi-resolution Eulerian solver in [7], with equivalent grid-sizes ranging from 32^4, at the coarsest level, to 256^4 at the finest level, and which is based on a third order finite-difference scheme.

As already mentioned, in the particular case of the Vlasov–Poisson equation, the semi-Lagrangian particle method is unconditionally stable, and the time accuracy of the algorithm is only dictated by the $(x, y, z)/(u, v, w)$ splitting. This splitting error is governed by the derivatives of velocity in the phase space, which are equal to 1 (for the three first components) and the spatial derivative of the electric field. For periodic boundary conditions, in energy norms these derivatives are in turn bounded by the density. In all our experiments the density value did not exceed 1 and we chose a constant value of 0.4 for the time-step. This time-step value correspond to a CFL value, based on the maximum particle velocity in the box, of 9 in the coarse grid case, and 18 in the finer grid case. Taking smaller time steps did not change the results shown below.

Unlike in mesh-free particle methods, in semi-Lagrangian particle methods the support of the density function can increase as a result of remeshing. To measure this spreading effect we show in Fig. 1 the particle numbers as a function of time for our run using the 128^4 grid. Surprisingly, the number of particles slightly decrease to settle to a value around $8\,10^7$. For comparison, the multi-resolution method of [7] with equivalent resolution between 32^4 and 256^4, used, beyond time $t = 10$, between $5\,10^6$ and $6\,10^7$ active grid points.

We now turn to the conservation properties of the method. Fig. 2 shows total mass, energy, entropy and L^2 norm of the distribution function f, normalized by their initial value, compared to the same quantities as obtained in [7]. Some observations can be made from these graphs, which highlight the differences between semi-Lagrangian schemes and Eulerian schemes. The conservation of mass and energy is almost perfect in the particle method, whereas in the calculations of [7] the energy tends to dissipate. The conservation of mass indicates that negative values resulting from remeshing with a

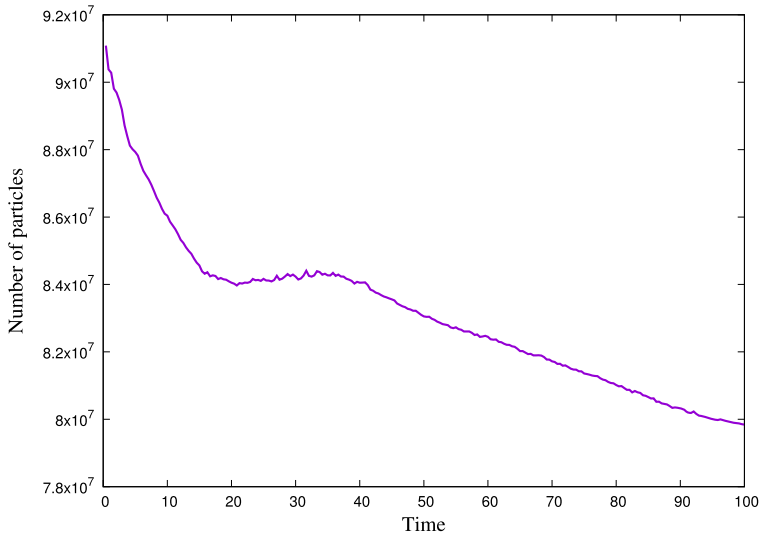

Fig. 1. Number of particles for the 4D Vlasov–Poisson two-beams instability with initial condition (7) and 128^4 effective grid resolution.

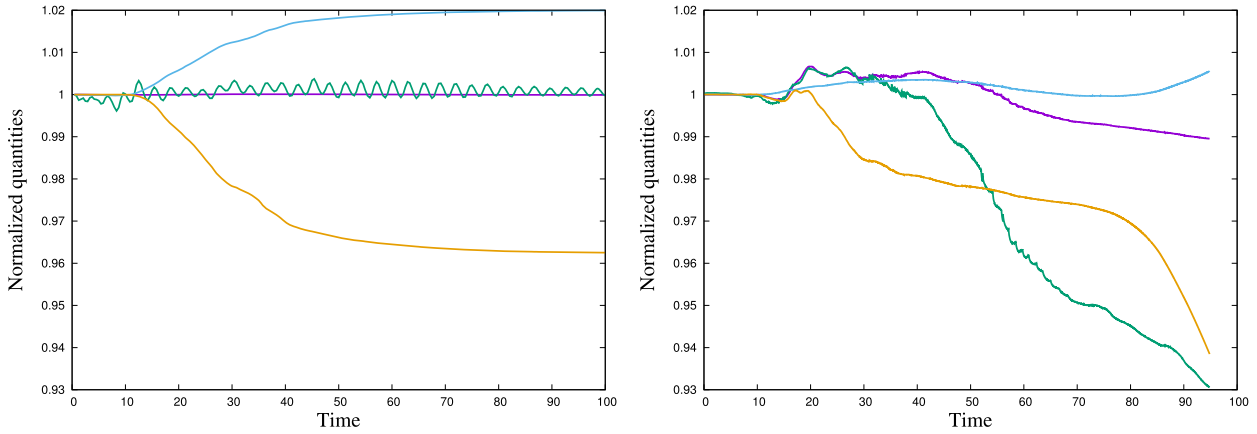

Fig. 2. Conservation properties for the same case as in Fig. 1. Left picture: present method; right picture: multi-resolution method of [7]. Magenta curve: mass; green curve: total energy; yellow curve: L^2-norm of f; blue curve: entropy. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

fourth order kernel, and which in our implementation are discarded after remeshing, would only have marginal contributions. This confirms a similar observation made in [15]. The L^2 norm of f drops at about 96% of its initial value then settles. The entropy increases by 2% then settles.

A further comparison of the solutions given by the two methods is given by Fig. 3. This figure shows cuts of the distribution function in the (x, u) plane, at $y = v = 0$ at time $t = 12$, when the potential energy reaches its peak value (see Fig. 6 for the time history of the potential energy). The two results are in perfect agreement.

We now show the results obtained with a coarser background grid using 64^4 points and the same time step value $\Delta t = 0.4$. Fig. 4 shows the conservation properties for this coarse grid compared to the finer grid. One can see that even for the coarse grid the method conserves pretty well the invariants of the Vlasov–Poisson system. The good performance of the coarse grid simulation is confirmed by a comparison of the cross section of f in the (x, u) plane at time $t = 12$ (Fig. 5). A more challenging comparison between the two resolutions can be made by looking at the potential energy $1/2 \int E^2$ alone (Fig. 6). One can see that the two resolutions give the same profile during the instability growth and oscillate around the same level for later times. The results of [7] by contrast show that the inherent dissipation in the underlying finite-difference scheme does not allow to maintain the potential energy at its correct level. These comparisons allow to conclude that the semi-Lagrangian particle method retains the desirable conservation properties of grid-free particle methods and gives rather well converged results even at coarse resolutions.

The CPU time required to perform the computation up to time $t = 100$ for the 128^4 resolution, with a particle number around $8\,10^7$ particles, was about 5.5 hours, for 250 iterations. The breakdown of the computational cost between the

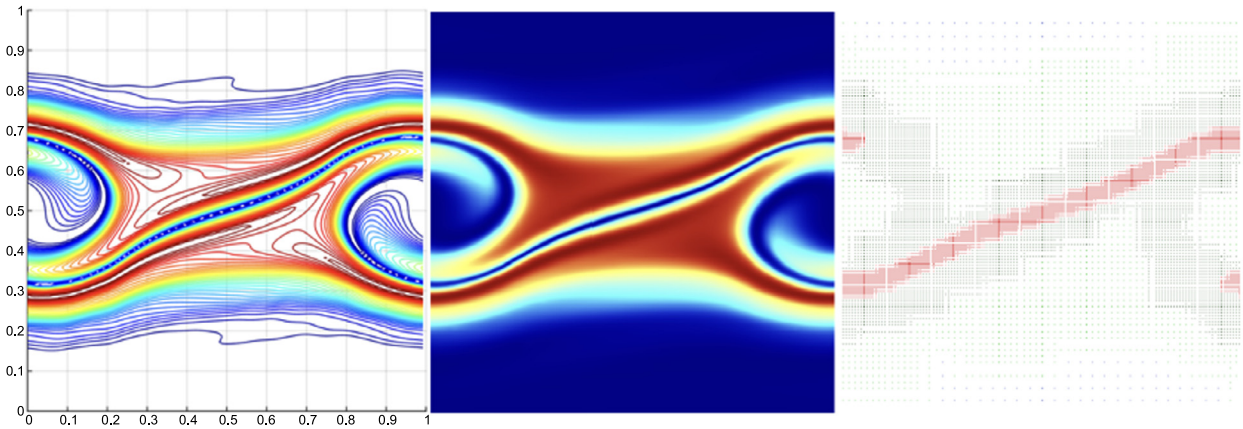

Fig. 3. Same case as in Fig. 1. Cut of the distribution function in the plane (x, u) at $y = v = 0$ and $t = 12$. Left picture: present method; middle picture: result of [7]; right picture: multi-resolution grid used in [7] (red zones correspond to an equivalent resolution of 256 grid points).

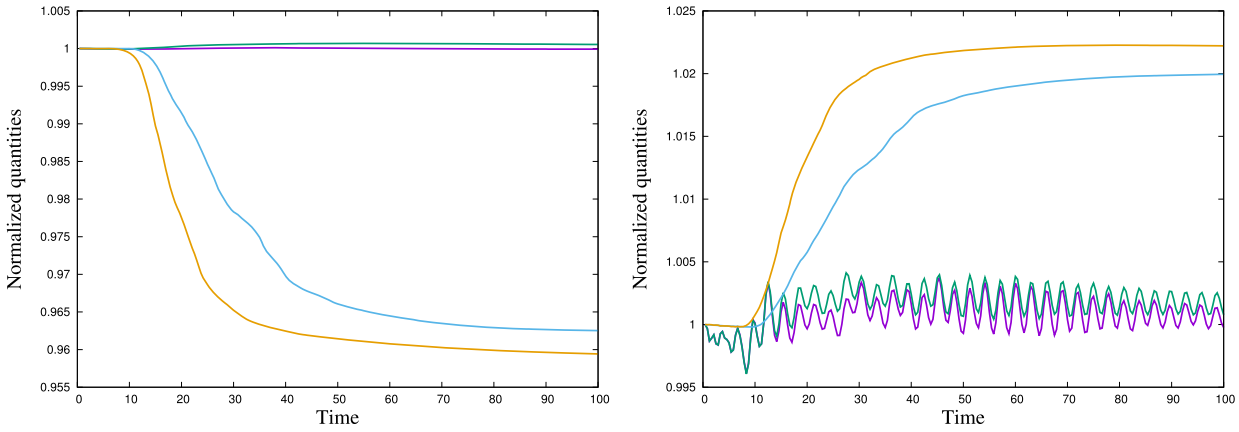

Fig. 4. Vlasov–Poisson two-beams instability with initial condition (7). Conservation properties with the present method based on coarse grid (CG, 64^4) and fine grid (FG, 128^4). Left picture: mass (magenta curve: FG, green curve: CG); L^2-norm of f (blue curve: FG, yellow curve: CG). Right picture: energy (magenta curve: FG, green curve: CG); entropy (blue curve: FG, yellow curve: CG).

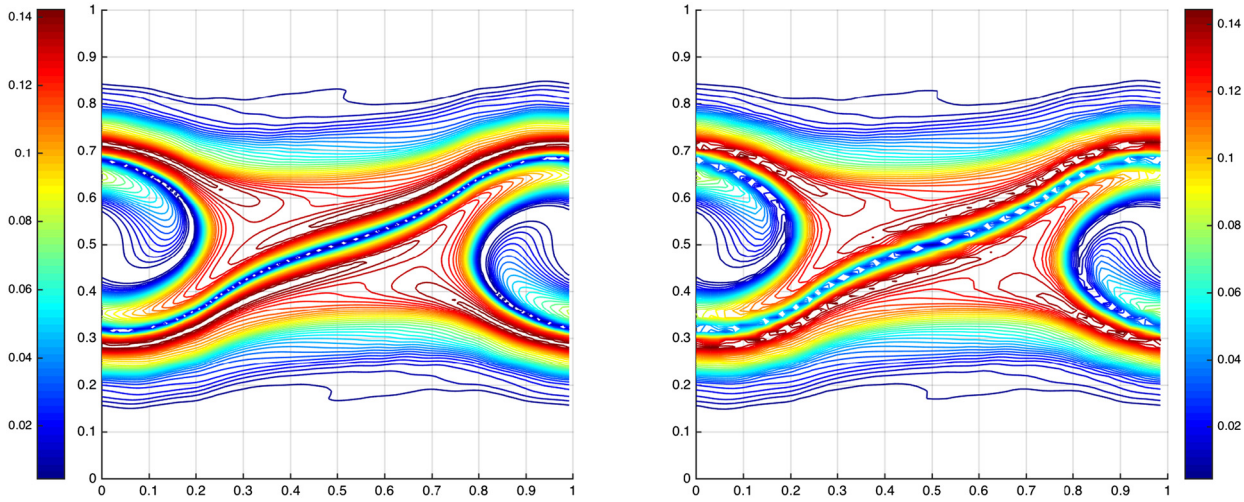

Fig. 5. Vlasov–Poisson two-beams instability with initial condition (7). Cut in the plane (x, u) at $y = v = 0$ of the distribution function f with the present method based on coarse grid (CG, 64^4) and fine grid (FG, 128^4). Left picture: FG; right picture: CG.

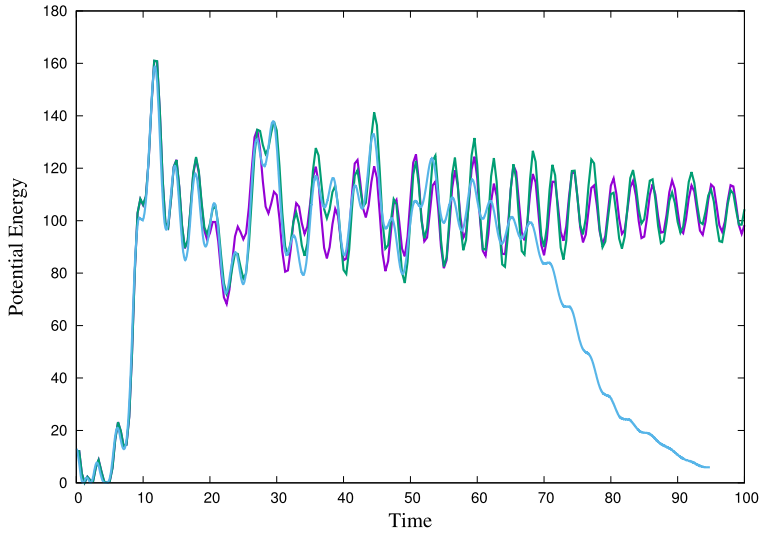

Fig. 6. Vlasov–Poisson two-beams instability with initial condition (7). Potential energy obtained with the present method with a 128^4 grid (green curve) and with a 64^4 grid (magenta curve) compared to the method in [7] (blue curve).

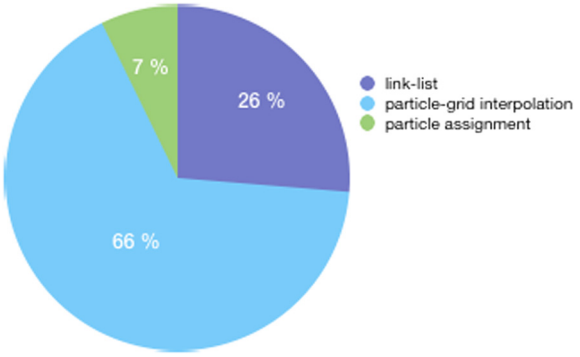

Fig. 7. Breakdown of computational cost in the semi-Lagrangian particle method for the two-beams Vlasov–Poisson instability.

link-list operations on the one hand and the particle-grid interpolations and particle assignment involved in the remeshing stages on the other hand, is given in Fig. 7. The other parts of the algorithms, including the FFT-based field evaluations, are responsible for less than 1% of the computational cost and are not shown on this graph. The memory size required for the higher resolution run was 7 GB.

4.2. Six dimensional gravitational case

We now turn to a more challenging case which involves a six-dimensional Vlasov–Poisson system. We consider the case of two density blobs, each determined by a steady-state Plummer model [10], and interacting with each other. Again we will compare our results with the multi-resolution results of [7], using equivalent resolutions ranging between 32^6 and 512^6, and also with results shown in this reference and provided by the GADGET grid-free particle software [20] (see Table 1 for the parameters of these simulations).

The distribution function of each blob is given by the following formula:

$$f_p(\Xi, \Psi) = \begin{cases} \frac{3}{7\pi^3}\left(2(1+|\Xi|^2)^{-1/2} - |\Psi|^2\right)^{7/2}, & \text{if } 2(1+|\Xi|^2)^{-1/2} - |\Psi|^2 \geq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

where $\Xi = (x, y, z)$ and $\Psi = (u, v, w)$ (see Fig. 8). This distribution function leads to a steady-state solution of the Vlasov Poisson system with unit density. Fig. 8 shows 2D and 1D cuts of f in the (x, u) plane, with all other variables set to 0.

Following [7] we choose an initial condition given by

$$f_0(x, y, z, u, v, w) = f_p(x - a_0, y, z - b_0, u - c_0, v, w) + f_p(x, y - a_0, z + b_0, u, v - c_0, w), \tag{9}$$

Table 1
CPU times for the present method, the multi-resolution method [7] and the GADGET software.

	4th order SL PM	8th order SL PM	Wavelet MRA [7]	GADGET [7]
Effective grid resolution	96	96	32 to 512	N.A.
Maximum number of active grid-points / particles	$1.8\,10^8$	$2.5\,10^8$	$5\,10^9$	$5\,10^8$
Number of time-steps	100	100	1349	N.A.
Wall clock CPU time	3.5 hours	5.8 hours	120 days	1 week
Hardware	1 Intel Xeon E5-2640 2.5 GHz	1 Intel Xeon E5-2640 2.5 GHz	32 Intel Xeon X5650 2.66 GHz	500 cores

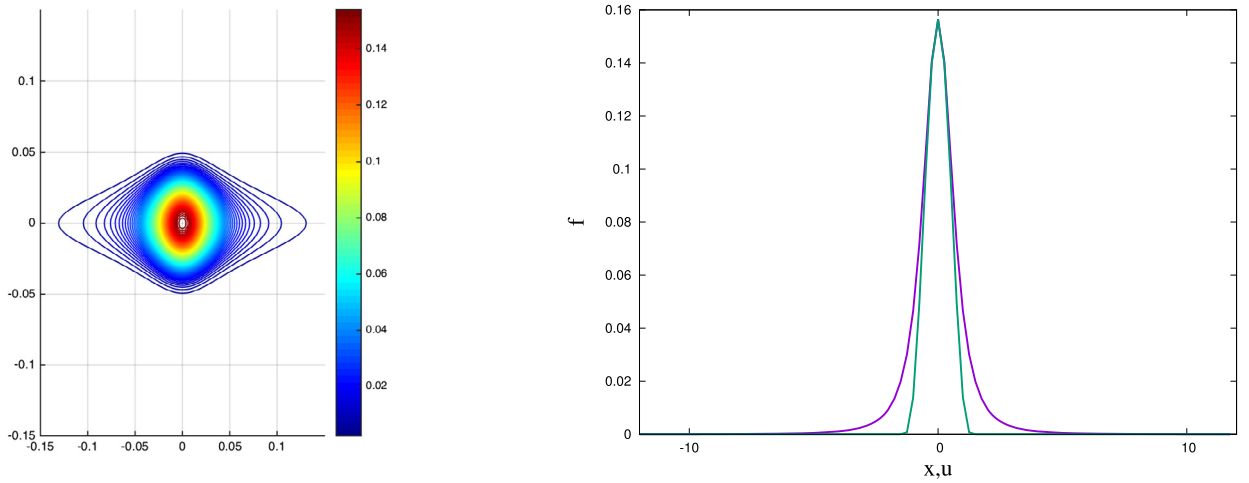

Fig. 8. Cross section in the plane (x, u) (left picture) and 1D cuts (right picture) corresponding to the distribution function given by (8). Green (resp. magenta) curve: cut in the u (resp. x) direction.

with $a_0 = -6$, $b_0 = -2$, $c_0 = 0.3$, in the box $\Omega = [-12, +12]^6$. The Poisson equation to obtain the gravity field $E = -\nabla\Phi$ from the density $\rho(\Xi) = \int f(\Xi, \Psi)\, d\Psi$ is

$$\Delta\phi = 4\pi(\rho - \bar{\rho}),$$

where $\bar{\rho} = 1/|\Omega| \int \rho\, d\Xi\, d\Psi$, with periodic boundary conditions The interaction of the two blobs produce a complex dynamics as they collide then separate then collide again, as shown in Fig. 9.

This case is more challenging than the previous one not only because of the dimension of the phase space but also because of the sharp profile of the distribution function. For high order finite-difference and semi-Lagrangian particle methods as well, this means that negative values and spurious oscillations are expected to arise.

As a matter of fact, and in strong contrast with the previous case, it turns out that discarding negative values in the remeshing stages of our algorithm as described in section 3 would severely damage the conservation of its invariants. A second observation is that, to obtain correct conservation properties, we found it necessary to decrease the threshold value to 10^{-6}, and therefore increase the number of particles.

Like in the previous case, we set the time-step value to $\Delta t = 0.4$ for all our simulations. Fig. 10 shows the number of particles as time goes on with the semi-Lagrangian particle method using the kernel $\Lambda_{4,2}$ and an underlying grid of 96^6 points. In that case the CFL number corresponding to our time-step and the maximum velocity value on particles is around 6. For comparison, the multilevel method of [7] with equivalent resolutions between 32^6 and 512^6 used a maximum of about $5\,10^9$ grid points in the same time interval and a time step varying between $1.2\,10^{-2}$ and $3\,10^{-2}$. The increase in the number of particles, which contrasts with what was observed in the previous section, results from the need to resolve small scales produced by the dynamics but also spurious oscillations created by particle remeshing. This simulation used about 24 GB of RAM memory.

In Fig. 11 we check the conservation of mass, entropy and L^2 norm of f compared to the multi-resolution method of [7]. One can see that, except for the total mass, the invariants produced by the particle method rapidly show some discrepancy, in particular for the L^2 norm of f. This is confirmed by the time history of the kinetic energy $E_k = 1/2 \int f(\Xi, \Psi)\, |\Psi|^2\, d\Xi\, d\Psi$. Fig. 12 shows how our method, with the 96^6 resolution and the kernel $\Lambda_{4,2}$ compares with the multi-resolution method of [7] and also with the result of the GADGET software using $5\,10^8$ particles.

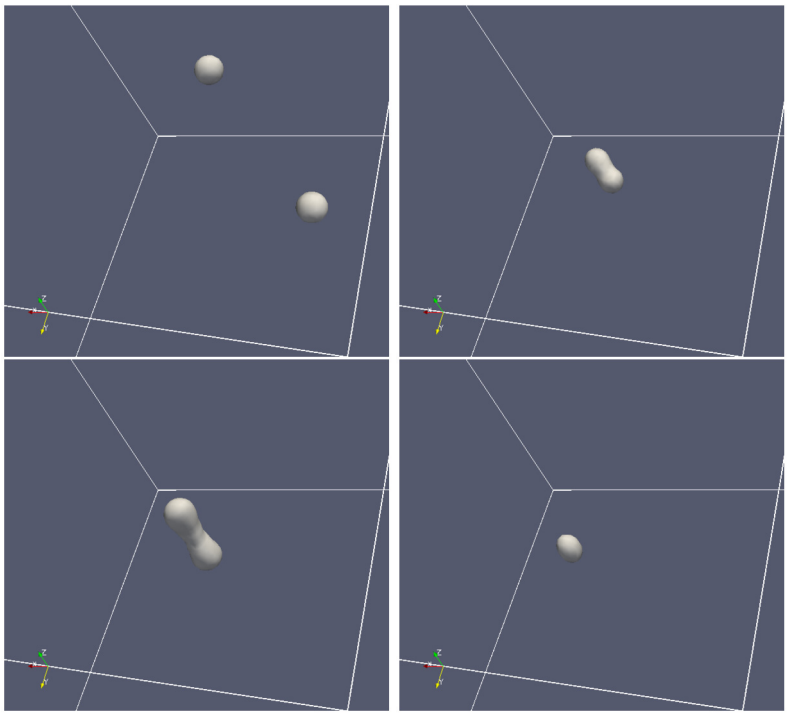

Fig. 9. Density rendering for the initial condition (9) at times (from left to right, top to bottom) 0.7, 15.4, 20 and 25.6. Isosurfaces correspond to one third of the maximum density which is respectively 0.22, 0.36, 0.11 and 0.35.

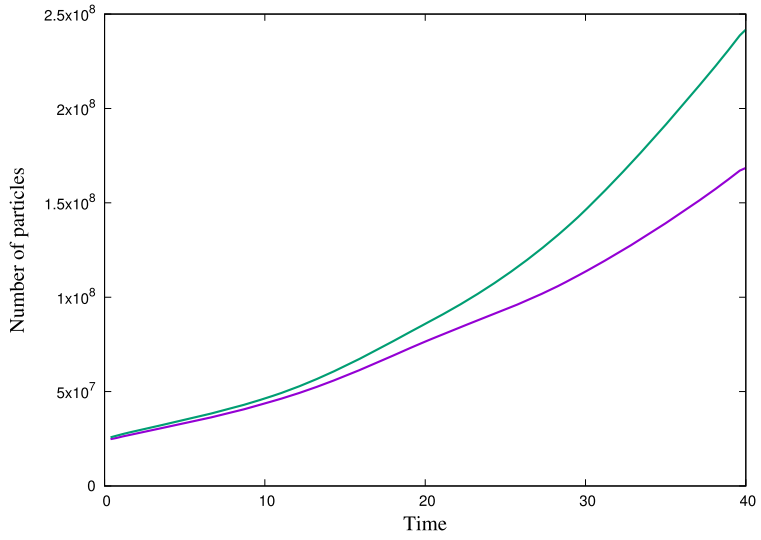

Fig. 10. Number of particles for the initial condition (9) and an underlying grid of 96^6 points with 4th and 8th order methods. Magenta curve: kernel $\Lambda_{4,2}$; green curve: kernel $\Lambda_{8,4}$.

To investigate whether higher order particle methods could improve these diagnostics, we next tested the $\Lambda_{8,4}$ kernel given by formula (6). Fig. 10 gives a comparison of the increase in number of particles which results from this remeshing formula with that obtained with the previous kernel. Fig. 13 shows the selected invariants and the kinetic energy when this 8th order method is used. With this higher order kernel, the loss in the L^2 norm of f is significantly reduced, in particular in the early stage of the simulation, and the method gives an excellent fit with GADGET for the kinetic energy. Although with a much lower maximum resolution, it avoids at the late stage of the simulation the numerical dissipation of the underlying finite-difference method in the MRA method of [7]. Note that an implementation of the method of [7] with a finest level of refinement corresponding to a 256^6 grid instead of 512^6 does not give the correct energy profile for the second collision around $t = 25$ [8].

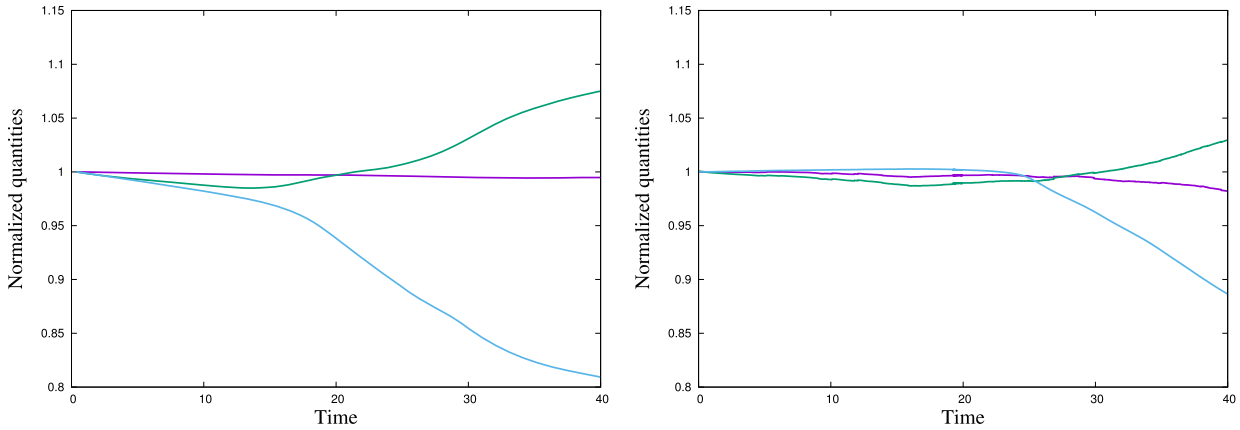

Fig. 11. Conservation properties for the gravity test (9). Let picture: present method with kernel $\Lambda_{4,2}$ and 96^6 grid. Right picture: multi-resolution method [7]. Magenta curves: total mass; green curves: entropy; blue curves: L^2-norm. Quantities are normalized by their initial value.

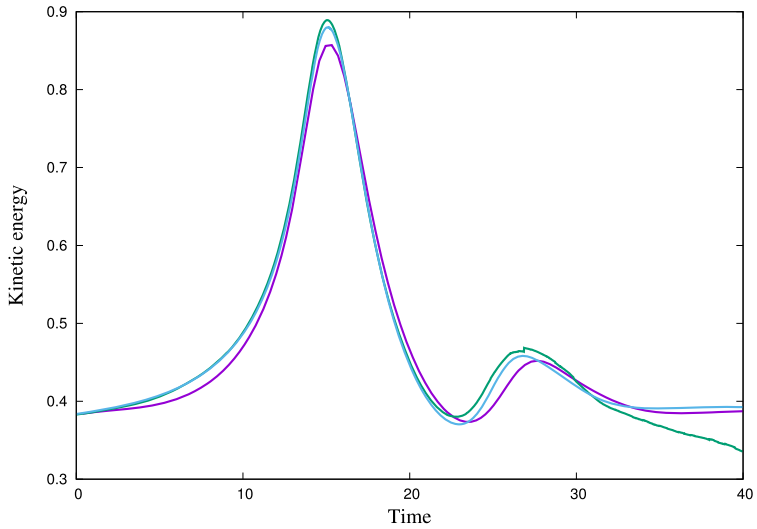

Fig. 12. Kinetic energy for the gravity test (9). Magenta curve: present method with kernel $\Lambda_{4,2}$ and 96^6 grid; green curve: multi-resolution method [7]; blue curve: GADGET simulation [7].

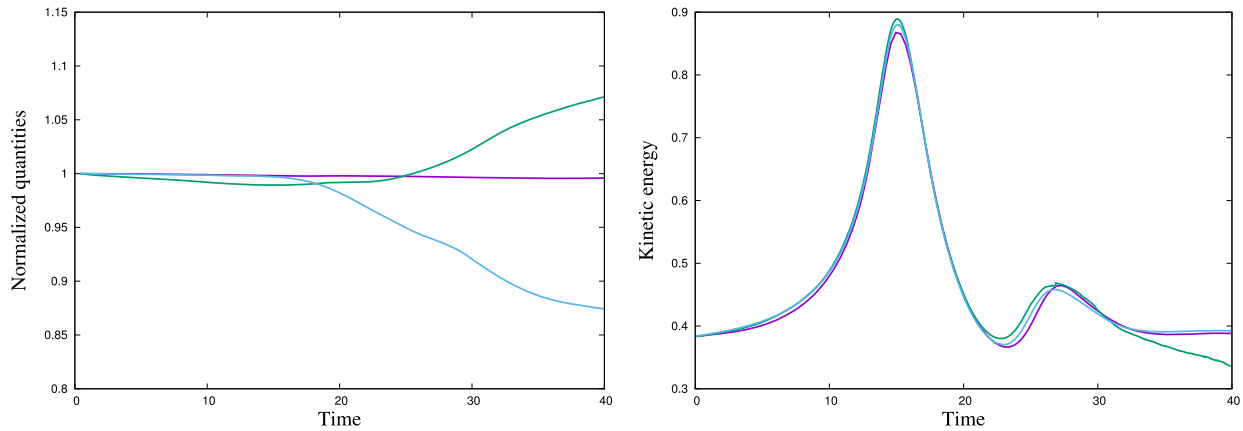

Fig. 13. Left picture: same as Fig. 11 with kernel $\Lambda_{8,4}$. Right picture: same as Fig. 12 with kernel $\Lambda_{8,4}$.

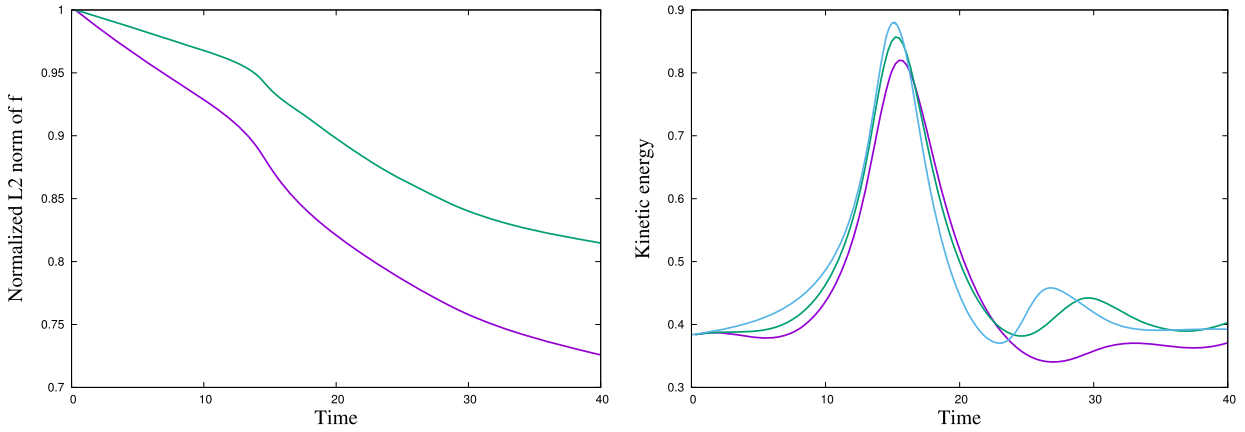

Fig. 14. Comparison of the methods using $\Lambda_{4,2}$ (magenta curves) and $\Lambda_{8,4}$ (green curves) kernels on a 64^6 grid. Left picture: L^2 norm of f; right picture: kinetic energy (blue curve is the reference GADGET result).

The improvement provided by the high order kernel is even clearer on lower resolution simulations. Fig. 14 shows the result obtained at a coarse resolution corresponding to a 64^6 grid. The high order kernel already provides reasonable results at this low resolution, albeit with a delayed second collision and at a lower level, whereas the 4th order kernel in particular totally fails to reproduce the second collision in the kinetic energy.

The satisfactory behavior of semi-Lagrangian particle methods to reproduce energy profiles and conserve invariants with affordable resolution should however not hide the fact that this resolution is not sufficient if one desires to obtain accurate local values of the distribution function. The comparison with the results obtained in multi-level method of [7] in Fig. 15 shows that the 96^6 equivalent resolution has difficulties to represent accurately the local values of the distribution function beyond time $t = 16$. Another caveat concerning the present method is that, as already mentioned, it does not preserve the positivity of the distribution function (note however that density values always remain positive). This difficulty, also present in the multi-resolution calculations in [7], is inherently linked to the use of high order (and thus non positive) interpolation kernel. It is possible to derive semi-Lagrangian methods with TVD limiters [14], but in the present case these methods proved to be over dissipative. Deriving along the same lines Weno type remeshing formulas is certainly possible but has not yet been tried. It could be fruitful in the present applications.

We now come to the computational complexity of the method. Table 1 compares the computational cost of the present method, with the two kernels used in our simulations, to those of the multi-resolution and GADGET simulations reported in [7].

One can first observe that the ratio in CPU times between semi-Lagrangian particle methods based on the $\Lambda_{4,2}$ and $\Lambda_{8,4}$ kernels matches pretty well the ratio between the size of their stencils (6 points vs 10 points). This indirectly confirms that using 3D tensor product formulas instead of splitting based formulas would significantly increase the computational cost of these methods. As already noted, the large CPU time required in the GADGET simulation results from the need to consider blobs containing many particles in the field calculation. We believe that the significant speed-up provided by the particle method compared to the multi-resolution method in [7] is not only due to a larger time step and a smaller number of particles, but also to its algorithmic simplicity. It is actually interesting to note that, assuming enough memory to run the particle method on an underlying uniform 512^6 grid, which is the maximum resolution in [7], and a perfect scaling of the CPU time, since in the particle method the time-step is independent of the spatial resolution the 4th order method would require about 3350 days on a single core, which compares well with the 120 days on 32 cores in [7]. The main advantage of the multi-resolution approach seems to be in the memory requirement (the high resolution simulation in [7] *only* requires 512 GB while we already need 24 GB). One can conclude that the localization property of semi-Lagrangian particle methods combined with their accuracy and algorithmic simplicity make them suitable for large scale computations even when used with uniform grids.

The breakdown of the computational time in the main stages of the algorithm is given in Fig. 16. It shows the same trends as in the previous 4D case, with however a reduced contribution of the link-list algorithm, due to the fact that this part of the method does not increase with the dimension of the problem, and an increased contribution of the assignment stages at the end of the remeshing steps.

5. Conclusion and outlook

In this paper we have presented implementations of high order semi-Lagrangian particle methods that could handle high dimensional Vlasov–Poisson systems on uniform grids with attractive trade-off between CPU costs and accuracy. The method was tested against state-of-the-art multi-resolution and grid-free particle methods in a 4D plasma instability case and in a 6D gravitational case. In both cases, the possibility to use large time-steps without compromising neither stability

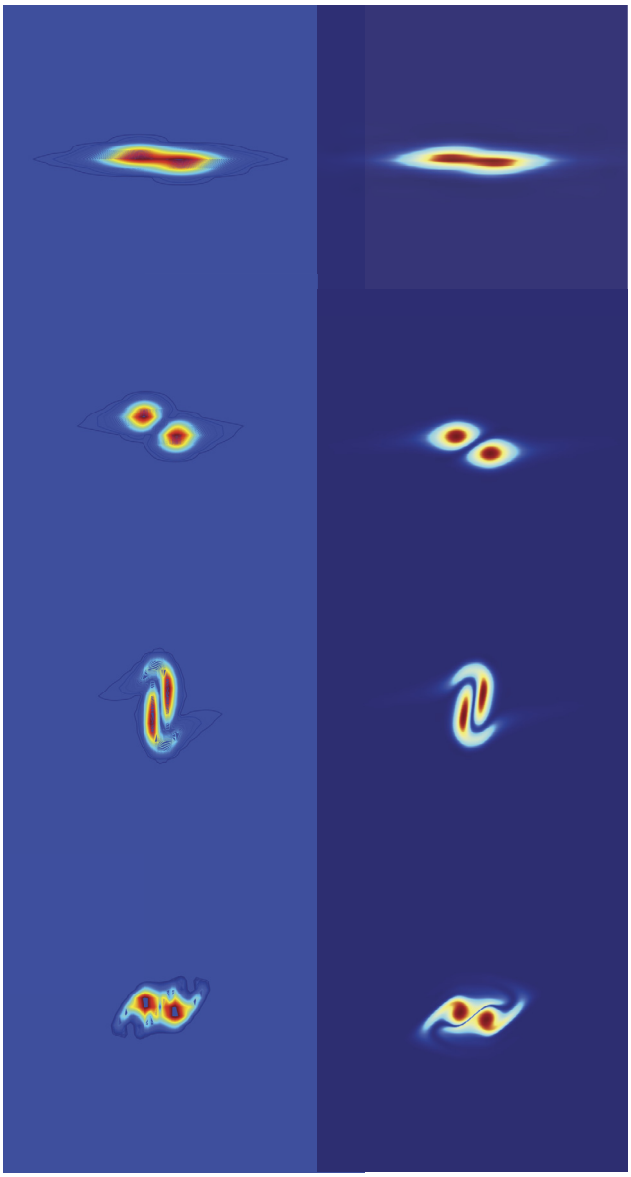

Fig. 15. Cuts in the plane (z, w) for the gravity test. Left column: present method with 96^6 resolution and $\Lambda_{8,4}$ kernel; right column: [7]. From top to bottom, times are 6, 12, 16 and 20.

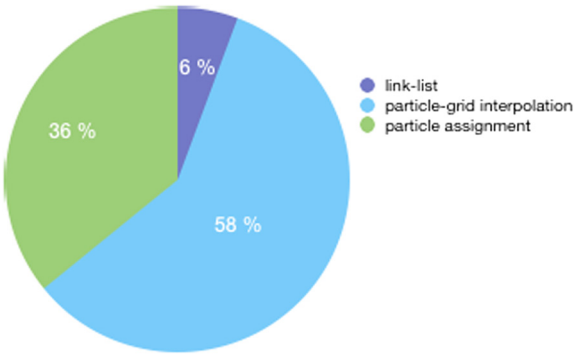

Fig. 16. Computational cost of the various stages of the semi-Lagrangian method for the 6D gravitational test.

nor accuracy was verified. In the first test, the method gives excellent results, even at coarse resolution, both in terms of global quantities and local values. The 6D case shows the benefit of using high order kernels and accurate global quantities are satisfactorily recovered at reasonable computational cost. The need of higher resolution is however apparent if accurate local values of the distribution function are sought in long time simulations.

Particle-grid methods naturally lend themselves to parallel implementations, including on GPU processors [16,4] or in hybrid GPU/CPU platforms [18]. The fact that more than 90% of the computational time reported here is devoted to particle-grid operations gives reasonable hope that the good scalability demonstrated in these prior works in fluid mechanics will carry on to the splitting strategy described here for the Vlasov–Poisson equations.

However it is important to note that, despite the localization properties demonstrated in the present work, memory requirements will clearly make parallel implementations of the present method not sufficient to reach level of resolutions comparable to the higher level of refinement in [7]. Future work will therefore be first and foremost to implement multi-resolution semi-Lagrangian particle methods, along the lines of [2] for sequential implementations, then [17] for multi-core implementations. One can expect from these further developments a valuable tool to address challenging multi-dimensional plasma or gravitational systems.

Acknowledgement

The author is grateful to Erwan Deriaz for kindly sharing his data and for enlightening discussions during the preparation of this work.

References

[1] M. Bergdorf, G.-H. Cottet, P. Koumoutsakos, Multilevel adaptive particle methods for convection–diffusion equations, SIAM J. Multiscale Model. Simul. 4 (2005) 328–357.
[2] M. Bergdorf, P. Koumoutsakos, A Lagrangian particle-wavelet method, SIAM J. Multiscale Model. Simul. 5 (2006) 980–995.
[3] N. Besse, E. Deriaz, E. Madaule, Adaptive multiresolution semi-Lagrangian discontinuous Galerkin methods for the Vlasov equations, J. Comput. Phys. 332 (2017) 376–417.
[4] G.-H. Cottet, J.-M. Etancelin, F. Perignon, C. Picard, High order semi-Lagrangian particles for transport equations: numerical analysis and implementation issues, ESAIM: Math. Model. Numer. Anal. 48 (2014) 1029–1060.
[5] N. Crouseilles, T. Respaud, E. Sonnendrucker, A forward semi-Lagrangian method for the numerical solution of the Vlasov equation, Comput. Phys. Commun. 180 (2009) 1730–1745.
[6] N. Crouseilles, M. Mehrenberger, E. Sonnendrucker, Conservative semi-Lagrangian schemes for Vlasov equations, J. Comput. Phys. 229 (2010) 1927–1953.
[7] E. Deriaz, S. Peirani, Six-dimensional adaptive simulation of the Vlasov equations using a hierarchical basis, Multiscale Model. Simul. 16 (2018) 583–614.
[8] E. Deriaz, personal communication.
[9] E. Fijalkow, Behaviour of phase-space holes in 2D simulations, J. Plasma Phys. 61 (1999) 65–76.
[10] T. Fujiwara, Integration of the collisionless Boltzmann equation for spherical stellar systems, Publ. Astron. Soc. Jpn. 35 (1983) 547–558.
[11] J.A.F. Hittinger, J.W. Banks, Block-structured adaptive mesh refinement algorithms for Vlasov simulation, J. Comput. Phys. 241 (2013) 118–140.
[12] R.W. Hockney, J.W. Eastwood, Computer Simulation Using Particles, CRC Press, 1988.
[13] P. Koumoutsakos, A. Leonard, High resolution simulations of the flow around an impulsively started cylinder, J. Fluid Mech. 296 (1995) 1–38.
[14] A. Magni, G.-H. Cottet, Accurate, non-oscillatory remeshing schemes for particle methods, J. Comput. Phys. 231 (2012) 152–172.
[15] A. Myers, P. Colella, B. Van Straalen, A 4th-order Particle-in-Cell method with phase-space remapping for the Vlasov–Poisson equation, SIAM J. Sci. Comput. 39 (2017) B467–B485.
[16] D. Rossinelli, M. Bergdorf, G.H. Cottet, P. Koumoutsakos, GPU accelerated simulations of bluff body flows using vortex particle methods, J. Comput. Phys. 229 (2010) 3316–3333.
[17] D. Rossinelli, B. Hejazialhosseini, W. van Rees, M. Gazzola, M. Bergdorf, P. Koumoutsakos, MRAG-I2D: multi-resolution adapted grids for remeshed vortex methods on multicore architectures, J. Comput. Phys. 288 (2015) 1–18.
[18] D. Rossinelli, C. Conti, P. Koumoutsakos, Mesh-particle interpolations on GPUs and multicore CPUs, Philos. Trans. R. Soc. A 369 (2011) 2164–2175.
[19] E. Sonnendrucker, J. Roche, P. Bertrand, A. Ghizzo, The semi-Lagrangian method for the numerical resolution of the Vlasov equation, J. Comput. Phys. 149 (1999) 20–220.
[20] V. Springel, The cosmological simulation code GADGET-2, Mon. Not. R. Astron. Soc. 364 (2005) 1105–1134.