Adaptive numerical resolution of the Vlasov equation[*]

Martin Campos Pinto[†] *and Michel Mehrenberger*[‡]

Abstract. A fully adaptive scheme (based on hierarchical continuous finite element decomposition) is derived from a semi-Lagrangian method for solving a periodic Vlasov-Poisson system. The first numerical results establish the validity of such a scheme.

1. Introduction

Most Vlasov solvers in use today are based on the Particle In Cell method which consists in solving the Vlasov equation with a gridless particle method coupled with a grid based field solver (see e.g. [4]). For some problems in plasma physics or beam physics, particle methods are too noisy and it is of advantage to solve the Vlasov equation on a grid of phase space. This has proven very efficient on uniform meshes in the two-dimensional phase space (see e.g. [13], [9] for structured meshes and [3] for unstructured meshes). However when the dimensionality increases the number of points on a uniform grid becomes too important for the method to be efficient, and it is essential to regain optimality by keeping only the 'necessary' grid points. Such adaptive methods have recently been developped, like in [14] and [2]-[10] where the authors use moving distribution function grids or interpolatory wavelets of Deslaurier and Dubuc. We refer also to [8] for a summary of many Vlasov solvers.
In this project, our first objective was to write a simpler algorithm based on hierarchical finite element decompositions. Compared to [2]-[10], this leads to less regularity and a heavier data structure, but the underlying partitions of dyadic tensor-product cells allow simpler memory space management and parallel implementations. After describing such hierarchical decompositions, we recall in section 3 the semi-Lagrangian scheme. The actual implementation of the scheme is given in section 4, and numerical results are presented in section 5 for two classical test cases, the linear Landau damping and the semi-Gaussian beam.

[*]This work was done in collaboration with our PhD advisors, Albert Cohen† and Eric Sonnendrücker‡.

[†]Laboratoire Jacques-Louis Lions, UMR CNRS 7598, Université Pierre et Marie Curie, Paris

[‡]Institut de Recherche Mathématique Avancée, UMR CNRS 7501, Université Louis Pasteur, Strasbourg

2. Adaptive representation of a phase space density

The *adaptivity* of our scheme relies on the representation, for each time step n, of the numerical solution f^n on a non-uniform moving mesh $\mathcal{M}^n$. But before introducing such adaptive meshes, let us first describe (with a few notations) the

Uniform meshes: considering the unit square $[0,1[^2$ as the computational domain, we denote by $\mathcal{M}_j$ the set of all phase space dyadic cells of resolution $j \in \mathbb{N}$:

$$\mathcal{M}_j := \{\alpha^j_{k,l} := \; [k\,2^{-j}, (k+1)\,2^{-j}] \times [l\,2^{-j}, (l+1)\,2^{-j}] \;:\; k,\; l \in \mathbb{Z}\,\},$$

and the corresponding approximation space by

$$V_j := \{\; f \in \mathcal{C}^0 \;:\; f_{|\alpha} \in \Pi_{1,1} \;\forall \alpha \in \mathcal{M}_j\;\},$$

consisting of all continuous functions that are bilinear on each cell. In the sequel, we shall use the short notation $|\alpha| := j$ to mean that $\alpha \in \mathcal{M}_j$, and always consider that j stands between two reference resolutions

$$j_0 \le j \le J, \tag{2.1}$$

or in other words, that $\mathcal{M}_j := \emptyset$ for any $j \notin \{j_0, \dots, J\}$.

In a very classical manner, we denote by $\Gamma(\alpha)$ the vertices of a cell α, and for any node a in $\Gamma_j := \cup_{\alpha\in\mathcal{M}_j}\Gamma(\alpha)$, we define the nodal function φ^j_a as the unique element of V_j that satisfies

$$\varphi^j_a(b) = \delta_{a,b}, \qquad b \in \Gamma_j.$$

The family $\{\varphi^j_a\}_{a\in\Gamma_j}$ is then a natural basis for V_j (normalized in L^∞) and a natural interpolatory projection on V_j is given by

$$P_j : \quad f \in \mathcal{C}^0 \;\to\; P_j f = \sum_{a\in\Gamma_j} f(a)\varphi^j_a \;\in\; V_j.$$

Adaptive meshes: to gain some flexibility in discretizing the numerical solutions, we construct meshes of variable resolution, and a natural way for doing so is to define *cell trees*. For that purpose, we first denote for any dyadic phase space cell α its children and parents by

$$\mathcal{D}(\alpha) := \{\;\beta \in \mathcal{M}_{|\alpha|+1} \;:\; \beta \subset \alpha\;\} \quad\text{and}\quad \mathcal{P}(\alpha) := \{\;\beta \in \cup_{\mathrm{j}\le|\alpha|-1}\mathcal{M}_{\mathrm{j}} \;:\; \beta \supset \alpha\;\},$$

and then define an **adaptive tree** Λ as a set of cells satisfying the two following properties:

1. $\mathcal{M}_{j_0} \subset \Lambda$
2. $\forall \alpha \in \Lambda,\; \bigcup_{\beta\in\mathcal{P}(\alpha)} \mathcal{D}(\beta) \subset \Lambda$.

This last property implies that no cell of Λ is *partially* refined, so that the leaves (*i. e.* the unrefined cells) form a partition of the phase space. Such a set $\mathcal{M} = \mathcal{M}(\Lambda)$ will be referred to as an **adaptive mesh**. Accordingly to the uniform case, we denote by

$$V_{\mathcal{M}} := \{\, f \in \mathcal{C}^0 \;:\; f_{|\alpha} \in \Pi_{1,1}\; \forall \alpha \in \mathcal{M} \,\}$$

the set of all continuous functions that are bilinear on each cell of $\mathcal{M}$, and by

$$\Gamma(\mathcal{M}) := \cup_{\alpha \in \mathcal{M}} \Gamma(\alpha),$$

the set of all nodes of $\mathcal{M}$.

Now if we want to define a projection $f_{\mathcal{M}} \in V_{\mathcal{M}}$ from a continuous function f, it is readily seen that the interpolatory equations

$$f_{\mathcal{M}}(a) = f(a), \quad a \in \Gamma(\mathcal{M})$$

are incompatible (unless $\mathcal{M}$ is uniform). Instead, we can define $f_{\mathcal{M}}$ as the unique element of $V_{\mathcal{M}}$ that satisfies

$$f_{\mathcal{M}}(a) = f(a), \quad a \in \Gamma_c(\mathcal{M}),$$

where

$$\Gamma_c(\mathcal{M}) := \{a \in \Gamma(\mathcal{M}) \;:\; \text{if } \beta \in \mathcal{M} \text{ satisfies } a \in \beta, \text{ then } a \in \Gamma(\beta)\}$$

denotes the **conforming nodes** of $\mathcal{M}$.

Obviously, $P_{\mathcal{M}} : f \to f_{\mathcal{M}}$ is a linear mapping, and with the additional requirement that $\mathcal{M}$ is *graded* (that is, two neighboring cells α and β satisfy $\big||\alpha| - |\beta|\big| \le 1$), $P_{\mathcal{M}}$ is locally stable in the sense that for any $\alpha \in \mathcal{M}$, we have

$$\|P_{\mathcal{M}} f\|_{L^\infty(\alpha)} \le \|f\|_{L^\infty\left(\mathcal{P}^1(\alpha)\right)},$$

where $\mathcal{P}^1(\alpha) := \mathcal{P}(\alpha) \cap \mathcal{M}_{|\alpha|-1}$ is the mother cell of α.

Wavelets. This data structure can also be described in the spirit of multiresolution analysis (see, for instance, [5] or [11]), since we see that the spaces V_j are nested. Thus for each j, we may define the *detail space* W_j as the complement of V_j with respect to V_{j+1} whose basis is given by $\{\varphi_a^{j+1}\}_{a \in \nabla_j}$, where $\nabla_j := \Gamma_{j+1} \backslash \Gamma_j$. It is indeed easily verified that

$$P_{j+1} f = P_j f + \sum_{a \in \nabla_j} d_a(f) \varphi_a^{j+1},$$

where the details $d_a(f)$ are defined by

$$d_a(f) := [P_{j+1} f - P_j f](a) = \sum_{b \in \Gamma_j} [f(a) - f(b)] \varphi_b^j(a), \tag{2.2}$$

where j is such that $a \in \nabla_j$. Following the usual wavelet notation, we write ψ_a instead of φ_a^{j+1}, and when a is a *coarse node* of $\nabla_{j_0-1} := \Gamma_{j_0}$, write for short $(d_a(f), \psi_a)$ instead of $(f(a), \varphi_a^{j_0})$. Hence the wavelet decomposition of f reads

$$P_J f = \sum_{a \in \nabla^J} d_a(f)\psi_a, \tag{2.3}$$

where $\nabla^J := \cup_{j=j_0-1}^{J-1} \nabla_j$.

Now defining the truncation operator

$$\mathcal{T}_B f \;:=\; \sum_{a \in B} d_a(f)\psi_a, \quad B \subset \nabla^J,$$

we can verify that $\mathcal{P}_{\mathcal{M}} = \mathcal{T}_{\Gamma_c(\mathcal{M})}$.

3. The semi-Lagrangian scheme

We briefly recall here the principle of the semi-Lagrangian scheme for solving the periodic one-dimensional Vlasov-Poisson system, and refer the reader to [1] for a more detailed presentation and a proof of convergence in the uniform case.

Denoting by $f(t,x,v)$ the distribution function in phase space (with $x, v \in \mathbb{R}$), and by $E(t,x)$ the self consistent electric field, the Vlasov-Poisson system reads

$$\partial_t f + v \cdot \partial_x f + E \cdot \partial_v f = 0 \tag{3.4}$$

$$\partial_x E = \rho, \tag{3.5}$$

where the charge density is given by

$$\rho(t,x) = \int_{-\infty}^{\infty} f(t,x,v)dv - 1. \tag{3.6}$$

If we consider that f and E are 1-periodic in space, this problem becomes well posed with the additional zero-mean electrostatic condition

$$\int_0^1 E(t,x)dx = 0, \quad t \geq 0, \tag{3.7}$$

and an initial data

$$f(0,x,v) = f^0(x,v), \quad x \in [0,1], \quad v \in \mathbb{R}. \tag{3.8}$$

The semi-Lagrangian method consists in taking advantage of the conservation of the density f along the characteristics, so let us first consider for a time step

Δt some approximation $\mathcal{B}(x,v)$ of the backward characteristics $(X(0),V(0))$ of (3.4)-(3.6) defined by

$$\left\{\begin{array}{l}\dot{X}(s)=V(s)\\ \dot{V}(s)=E(s,X(s))\end{array}\right. \quad \text{for } s\in[0,\,\Delta t],\ \text{ and } \left\{\begin{array}{l}X(\Delta t)=x\\ V(\Delta t)=v\end{array}\right. . \tag{3.9}$$

In the uniform case, for $n=0,1\dots$, we approximate on each time step the solution of (3.4)-(3.6) with initial value f^n by

$$f^{n+1}:=P_J\mathcal{A}f^n, \tag{3.10}$$

where the nonlinear **advection operator** $\mathcal{A}$ is defined by

$$\mathcal{A}\ :\ f^n\to f^n\circ\mathcal{B}. \tag{3.11}$$

The adaptive version of this scheme only differs by the use of an adaptive (and moving) mesh $\mathcal{M}^n$ on which f^n is represented (in the sense that $f^n\in V_{\mathcal{M}^n}$). We thus basically have to add a *mesh moving* step in the scheme, that is a prediction of the new mesh $\mathcal{M}^{n+1}$ from the data $(f^n,\mathcal{M}^n)$. The **adaptive semi-Lagrangian scheme** reads then

$$(f^n,\mathcal{M}^n)\to\mathcal{M}^{n+1}, \tag{3.12}$$

$$f^{n+1}:=P_{\mathcal{M}^{n+1}}\mathcal{A}f^n \tag{3.13}$$

for $n=0,1,\dots$

Remark (practical mesh prediction): the problem of constructing an adaptive mesh $\mathcal{M}_\varepsilon(f)$ well fitted to a *given* f is well understood (provided that f has some smoothness, see [7] or [5] for a general survey on nonlinear approximation) and near optimal algorithms can be obtained by using adaptive splitting, or wavelet-based *thresholding operators*

$$\mathcal{T}_\varepsilon\ :\ f\to P_{\mathcal{M}_\varepsilon(f)}f$$

for which $\|f-\mathcal{T}_\varepsilon f\|$ is lower than a prescribed tolerance $\varepsilon>0$. On the other hand, it is a much more difficult task to predict a mesh $\mathcal{M}^{n+1}$ that is well adapted to the *unknown* solution f^{n+1}. Hence, like most adaptive methods, we first construct a 'pessimistic' guess $\tilde{\mathcal{M}}^{n+1}$ from f^n (by a heuristic procedure), compute then a temporary solution $\tilde{f}^{n+1}\in V_{\tilde{\mathcal{M}}^{n+1}}$ following (3.13), and eventually discard the small coefficients by a compression step, computing $f^{n+1}:=P_{\mathcal{M}^{n+1}}\tilde{f}^{n+1}$ on $\mathcal{M}^{n+1}:=\mathcal{M}_\varepsilon(\tilde{f}^{n+1})$. In the context of finite volume schemes for solving conservation laws, such a refinement strategy is precisely described in [6], together with a proof of convergence.

4. Biquadratic second order in time adaptive scheme

It is well known that linear reconstruction is too diffusive in the numerical resolution of the Vlasov equation. We now detail our practical implementation of the scheme, based on biquadratic finite elements instead of bilinear ones.

Adaptive biquadratic representation: we denote by $\Gamma^2(\alpha)$ the 9 equidistant nodes of a biquadratic cell α, and by $\Gamma^2(\mathcal{M})$ the nodes of all the cells of a given adaptive mesh $\mathcal{M}$. So, our numerical solution at time n is the data

$$\mathcal{F}_n := (\mathcal{M}^n, \quad (f^n(a))_{a\in\Gamma^2(\mathcal{M}^n)}), \tag{4.14}$$

where $\mathcal{M}^n$ is an adaptive mesh. The evaluation $f^n(c)$ of the solution at a point $c \in [0,1[^2$ is obtained by searching the unique cell α of the adaptive mesh $\mathcal{M}$ where the point lives, using the values $f^n(\alpha) := (f^n(a))_{a\in\Gamma^2(\alpha)}$ and computing the local biquadratic interpolation on that cell, say $I(c, \alpha, f^n(\alpha))$.

Characteristics: let us denote by $(X, V)(s; x, v, t)$ the characteristics of the Vlasov equation, i.e. the solutions of the following system of ordinary differential equations:

$$\begin{cases} \dot{X}(s) = V(s) \\ \dot{V}(s) = E(s, X(s)) \end{cases}$$

with initial conditions $X(t) = x$, $V(t) = v$. The advection is classically performed by time-splitting (see e.g [12]). A general two time step method has been introduced in [13] to solve the characteristics, by a direct $2D$ computation (see also [14] for an efficient new method). We have used the following completely local algorithm. Knowing the final position $a = (X^{n+1}, V^{n+1})$ at time $(n+1)\Delta t$, we compute a second order approximation (X^n, V^n) of the backward advected position $(X, V)(n\Delta t; a, (n+1)\Delta t)$.
So, with the notations $t^n = n\Delta t$, $t^{n+\frac{1}{2}} = (n + \frac{1}{2})\Delta t$ and $X^{n+\frac{1}{2}} = \frac{X^n+X^{n+1}}{2}$, we can write to second order accuracy:

$$\frac{X^{n+1} - X^n}{\Delta t} = \frac{V^n + V^{n+1}}{2} + O(\Delta t^2).$$

On the other hand, again to second order accuracy:

$$\begin{aligned} \frac{V^{n+1} - V^n}{\Delta t} &= E(t^{n+\frac{1}{2}}, X^{n+\frac{1}{2}}) + O(\Delta t^2) \\ &= \frac{1}{2}[E(t^n, X^{n+\frac{1}{2}}) + E(t^{n+1}, X^{n+\frac{1}{2}})] + O(\Delta t^2) \end{aligned}$$

and

$$\frac{1}{2}[E(t^{n+1}, X^{n+\frac{1}{2}}) + E(t^{n-1}, X^{n+\frac{1}{2}})] = E(t^n, X^{n+\frac{1}{2}}) + O(\Delta t^2).$$

So, after some more computations, we obtain a second order accurate formula,

$$\frac{V^{n+1} - V^n}{\Delta t} = 2E(t^n, \frac{X^n + X^{n+1}}{2}) - \frac{E(t^{n-1}, X^n) + E(t^n, X^{n+1})}{2} + O(\Delta t^2). \tag{4.15}$$

Poisson electric field : from the data $\mathcal{F}_n$ (defined in (4.14)), we can derive the density of charge $\rho^n = \int f^n(x,v)\mathrm{d}v$ and then obtain the electric field E^n, from the Poisson equation :

$$-\partial_x E^n = \rho^n - 1$$

(in the axisymmetric case, we will take $-\frac{1}{r}\partial_r(rE_r) = \rho$).

Backward/forward advection: we denote by $B_n^{wd}(a)$ the backward advected position of a node a. We obtain it here by (4.15) with $a = (X^{n+1}, V^{n+1})$ and $B_n^{wd}(a) = (X^n, V^n)$. This system can be solved iteratively for the unknown V^n. On the other hand $F_n^{wd}(a)$ denotes the forward advected position and is also given by (4.15), writing $a = (X^n, V^n)$ and $F_n^{wd}(a) = (X^{n+1}, V^{n+1})$.

The algorithm is now guided by two quantities : the **finest resolution level** J and a **thresholding tolerance number** ε .

Time marching algorithm.

• Initialization
The initial function is given by f_0 on the unit square.
Let $\mathcal{M}^0$ be at first empty.
Starting from $j = J-1$ to j_0, for each cell $\alpha \in \mathcal{M}_j$,
▷ compute : $d(b) := f_0(b) - I(b, \alpha, f_0(\alpha))$ for $b \in \Gamma^2(\mathcal{D}(\alpha))\backslash\Gamma^2(\alpha)$
▷ add α in $\mathcal{M}^0$ if the following compression test is false :

$$\sum_{b\in\Gamma^2(\mathcal{D}(\alpha))\backslash\Gamma^2(\alpha)} \omega_{|\alpha|}|d(b)| \leq \varepsilon, \tag{4.16}$$

where $|\alpha|$ is the level of the cell α and ω_j stands for 1, 2^{-dj}, $2^{-dj/2}$ depending on the norm on which we want to compress (respectively L_∞, L_1 or L_2), with d the dimension of the code (here $d = 2$). In our simulations, we will always take the L_2 norm.
▷ add the necessary cells in order to have an adaptive mesh (i.e : so that the tree structure is respected).

For the sequel of the algorithm, we loop on n.

• Prediction of $\tilde{\mathcal{M}}^{n+1}$
Let $\tilde{\mathcal{M}}^{n+1}$ be at first empty.
For each center c of cell α of the adaptive mesh $\mathcal{M}^n$,
  $\triangleright$ compute the forward advected point $F_n^{wd}(c)$
  $\triangleright$ add the unique cell of level $|\alpha|$ which fits at that place in $\tilde{\mathcal{M}}^{n+1}$.
Finally,
  $\triangleright$ add the necessary cells so that $\tilde{\mathcal{M}}^{n+1}$ is an adaptive mesh,
  $\triangleright$ refine the cells (which are not of level J) of one level, that is, replace each cell by its daughters.

• Semi-Lagrangian advection
For each node $a \in \Gamma^2(\tilde{\mathcal{M}}^{n+1})$,
  $\triangleright$ compute the backward advected point $B_n^{wd}(a)$,
  $\triangleright$ set $\tilde{f}^{n+1}(a)$ to $f^n(B_n^{wd}(a))$
Now, we have a first $\tilde{\mathcal{F}}_{n+1}$.

• Data compression: $\tilde{\mathcal{F}}_{n+1} \to \mathcal{F}_{n+1}$
Starting from $j = J-1$ to j_0+1, for each cell $\alpha \in \tilde{\mathcal{M}}^{n+1}$ of level j,
  $\triangleright$ compute: $d(b) := \tilde{f}^{n+1}(b) - I(b, \alpha, \tilde{f}^{n+1}(\alpha))$ for $b \in \Gamma^2(\mathcal{D}(\alpha)) \backslash \Gamma^2(\alpha)$
  $\triangleright$ remove α in $\tilde{\mathcal{M}}^{n+1}$ if the compression test (4.16) is true.
The remaining data is $\mathcal{F}_{n+1}$.

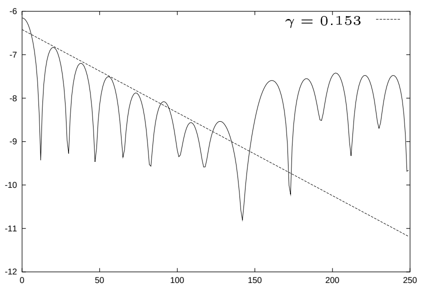
5. Test Cases

Linear Landau damping. In order to test the precision of the numerical scheme, the linear Landau damping is very classical. The initial condition is given by

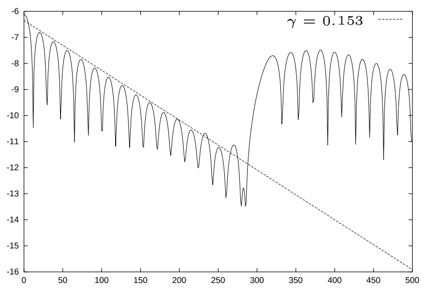
$$f(0,x,v) = \frac{1}{\sqrt{2\pi}} e^{-v^2/2}(1+\alpha\cos(kx))$$

with $\alpha = 0.01$, the period equals $L = 4\pi$ and $k = 0.5$. For the time discretization, we choose $\Delta t = 1/8$. We restrain our computional domain in velocity to an interval $[-v_{max}, v_{max}]$, with a number v_{max} big enough. The electric field decays exponentially with a theoretical decay rate of $\gamma = 0.1533$ in the L_2 norm (see e.g. [12]). In fact, the numerical solution cannot decay all the time and the solution should restitute its energy at a "recurrence time" (see e.g [12]). However, by taking more points in the velocity direction, we can push this phenomenon away and thus have a better description of the electric field for longer times.

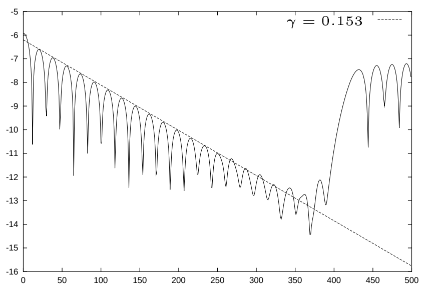
In the adaptive case (see figure 2), we take $J = 6$ and $v_{max} = 7.15$ since these parameters give good results in the uniform case (see figure 1): we obtain about



(a) $\Delta v = 0.281$

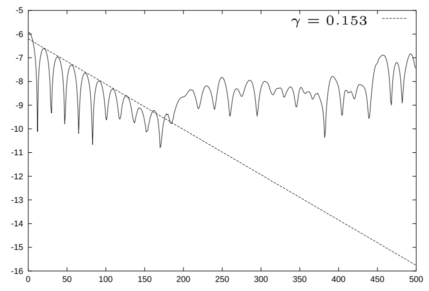


(b) $\Delta v = 0.148$

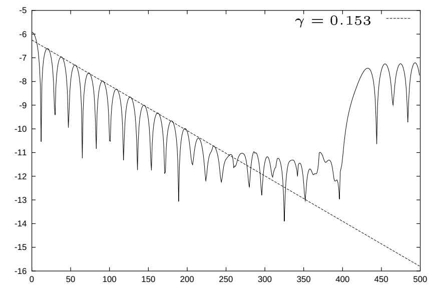


(c) $\Delta v = 0.112$

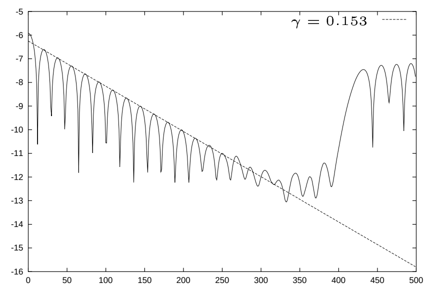
Figure 1: evolution of $\log(\int E(x,t)^2 dx)$ in terms of the number of iterations ($\Delta t = 0.125$) in the uniform case with resolution level $J = 4, 5$ and 6 (that is, respectively 256, 1024 and 4096 cells) for the linear Landau damping.



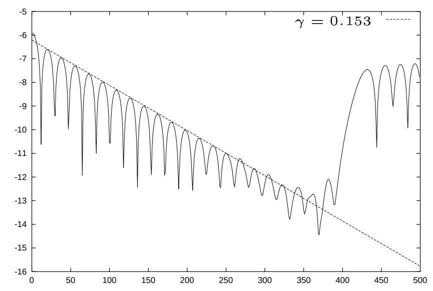
(a) $\simeq 2000 - 1400$ cells



(b) $\simeq 2400 - 2200$ cells



(c) $\simeq 2700 - 2650$ cells



(d) $\simeq 3100 - 2900$ cells

Figure 2: evolution of $\log(\int E(x,t)^2 dx)$ in terms of the number of iterations ($\Delta t = 0.125$) with $\Delta v = 0.112$ in adaptive case (with finest resolution level $J = 6$ and thresholding tolerance $\varepsilon = 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}$) for the linear Landau damping.

20 periods of oscillations. As expected, the accuracy of the damping increases, as the tolerance decreases and we reach the accuracy of the underlying uniform case with $\varepsilon = 10^{-8}$. The L_1 and L_2 norm are well conserved: the L_2 norm loses less than $4 \cdot 10^{-6}$ relative weight for $\varepsilon = 10^{-6}$ and the relative total mass error is less than 10^{-6} (however, if the tolerance is too large, the results are worser: if $\varepsilon = 10^{-5}$, the L_2 norm increases of 10^{-4} after 500 iterations).
On the other hand, the proportion of cells saved is about 1/4 (for $\varepsilon = 10^{-8}$, about 3100 cells are used at the beginning and 2900 after the 100 first iterations; for $\varepsilon = 10^{-6}$, we go from 2400 to 2200 cells): the compression is occuring where the solution is almost null.
Thus, if the use of such an adaptive grid seems not to be the most natural way to treat this test case since the solution does not develop small scales, we are now convinced that, according to this example, the adaptive scheme can go to the accuracy of the uniform solution if we choose a tolerance small enough. We have also pointed out that, with such methods, we can increase the domain of calculation with a very small additional cost.

Semi-Gaussian Beam. We now consider the semi-Gaussian beam defined by the initial condition

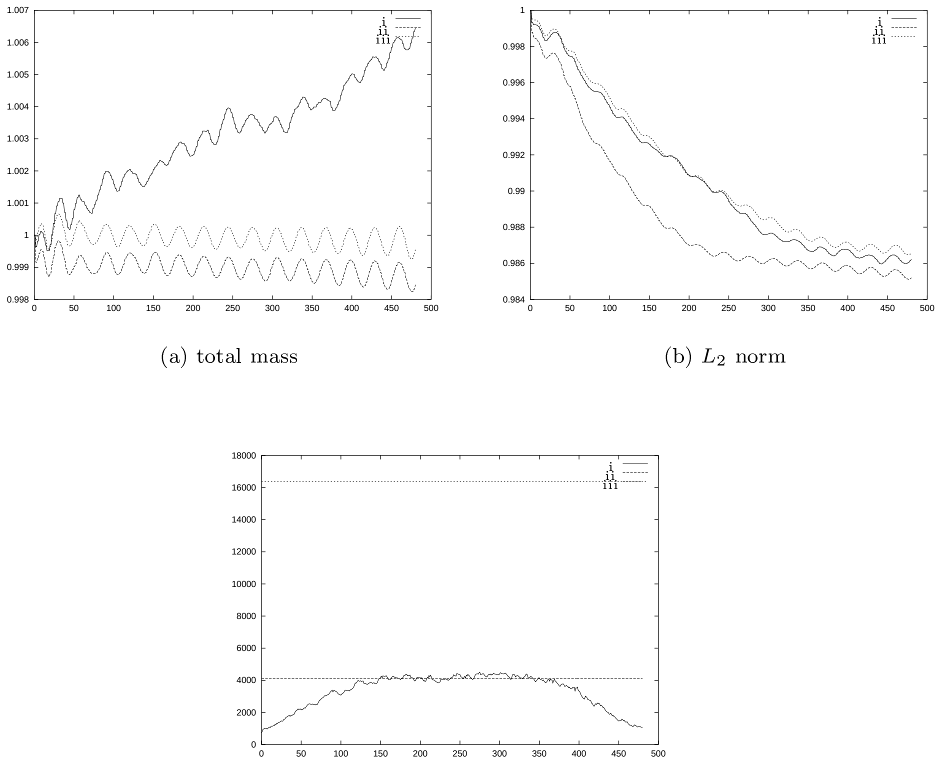
$$f_0(r,v) = \frac{1}{\pi a^2 \sqrt{2\pi}} b e^{-\frac{1}{2}(v^2/b^2)}, \quad \text{if} \quad r < a,$$

and $f_0(r,v) = 0$ elsewhere. Here $a = 4/\sqrt{15}$ and $b = 1/(2\sqrt{15})$. The time step is $\pi/16$ which corresponds to 1/32 of a period. We make 480 iterations, that is 15 periods.

The numerical solution develops small scales which disappear after a while, by diffusion. In our simulation, we have set $\varepsilon = 10^{-3}$ and $J = 7$ for the adaptive case.
We compare our adaptive solution to a coarse uniform solution (with the coarser resolution $J = 6$), and to a fine uniform one (with the same resolution $J = 7$). We see on figures 4, 5 and 6 that the grid follows the development of the small scales, while the ratio $\#(\mathcal{M}^n)/\#(\mathcal{M}_6)$ goes in 6 periods from 1/4 to 1 (see figure 3(c)). After 3 periods (figure 5), the adaptive solution seems to better catch the nonlinear effects than the coarse uniform one, it remains in fact as accurate as the fine uniform solution. On figure 6, we approach the full filamentation zone; the adaptive solution remains better than the coarse one, with always the same order of cells (see also 3(c)), but this time can not reach the accuracy of the fine uniform one. After 15 periods, the diffusion occurs and the derefining process works: we turn to the initial number of cells (see figure 7 and 3(c)).
By using biquadratic interpolation, the scheme is quite diffusive, however the adaptivity does not accelerate this phenomenon: the L_2 norm of the adaptive and fine uniform solutions are quite similar (figure 3(b)). On the other hand, the mass is not well conserved in the adaptive case (figure 3(a)): it tends to increase and this

may come from the loss or gain of weight in the compression step and the lack of conservation of the interpolation on a non-uniform grid in the semi-Lagrangian scheme.

As a conclusion, this method seems to give the expected results. However, there is still much to do: the lack of mass conservation could be improved, and higher order polynomial interpolation should be performed in order to be more accurate. At a more theoretical point of view, we are studying the convergence and the complexity of this adaptive scheme.

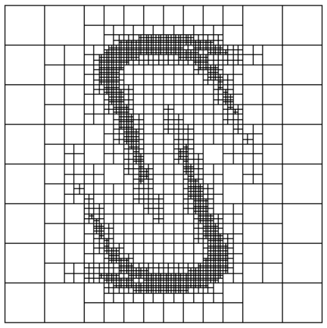


(a) total mass

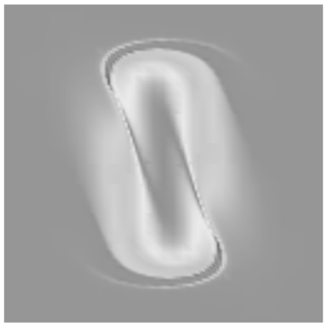
(b) L_2 norm

(c) number of cells

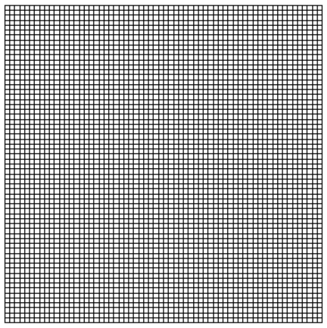
Figure 3: evolution of the relative **total mass**, L_2 **norm** of the distribution function f for the semi-Gaussian beam and the **number of cells** in terms of the number of iterations (32 iterations = 1 period) in the adaptive case with $J = 7$ and $\varepsilon = 10^{-3}$ (i), in the uniform case with $J = 6$ (ii) and with $J = 7$ (iii).



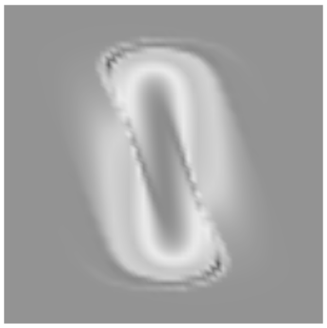
(a) $J = 7$, $\varepsilon = 10^{-3}$



(b) $J = 7$, $\varepsilon = 10^{-3}$

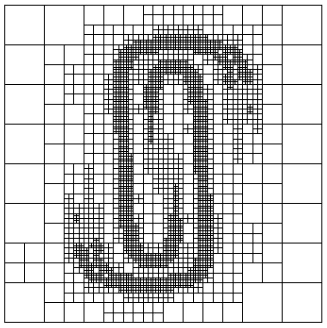


(c) $J = 6$ uniform

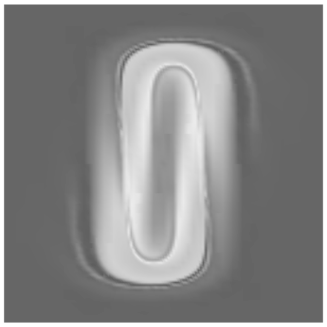


(d) $J = 6$ uniform

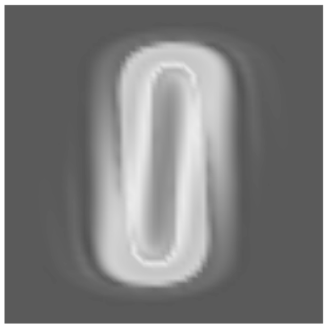
Figure 4: adaptive grid and distribution function f in (v, x) phase space after 1.5 period (48 iterations).



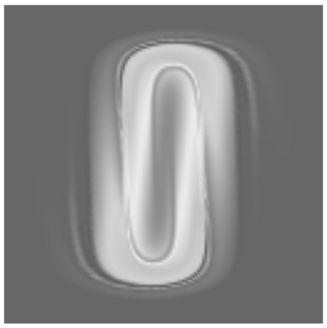
(a) $J = 7$, $\varepsilon = 10^{-3}$



(b) $J = 7$, $\varepsilon = 10^{-3}$

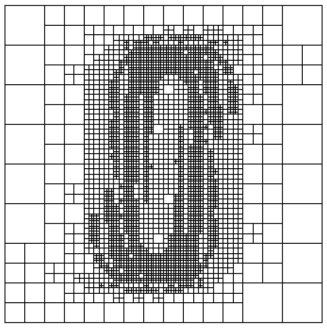


(c) $J = 6$ uniform

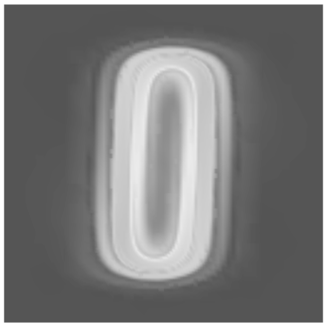


(d) $J = 7$ uniform

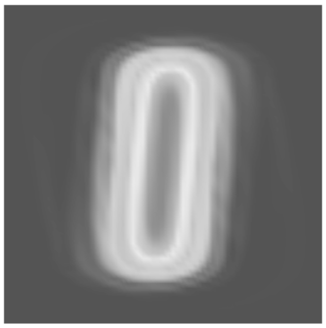
Figure 5: adaptive grid and distribution function f in (v, x) phase space after 3 periods (96 iterations).



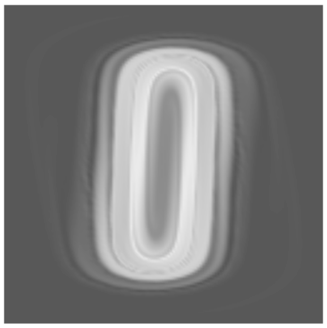
(a) $J = 7$, $\varepsilon = 10^{-3}$



(b) $J = 7$, $\varepsilon = 10^{-3}$

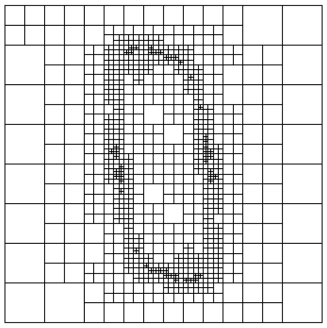


(c) $J = 6$ uniform

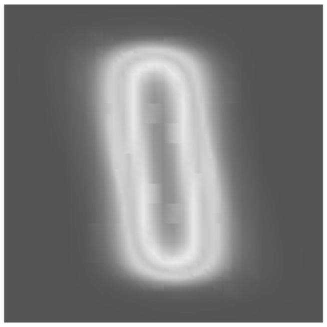


(d) $J = 7$ uniform

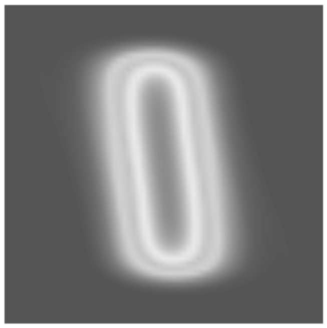
Figure 6: adaptive grid and distribution function f in (v, x) phase space after 6 periods (192 iterations).



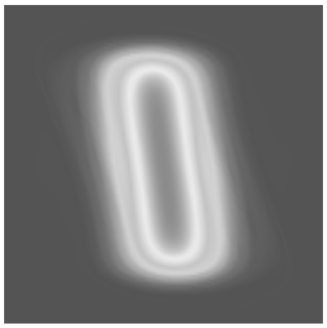
(a) $J = 7$, $\varepsilon = 10^{-3}$



(b) $J = 7$, $\varepsilon = 10^{-3}$



(c) $J = 6$ uniform



(d) $J = 7$ uniform

Figure 7: adaptive grid and distribution function f in (v, x) phase space after 15 periods (480 iterations).

Acknowledgments The authors thank the CEMRACS organizers.

References

[1] N. Besse, *Convergence of a semi-Lagrangian scheme for the one-dimensional Vlasov-Poisson system* , SIAM J. Numer. Anal., Vol 42, (2004), pp. 350-382.

[2] N. Besse, F.Filbet, M. Gutnic, I. Paun, E. Sonnendrücker, *An adaptive numerical method for the Vlasov equation based on a multiresolution analysis*, Numerical Mathematics and Advanced Applications ENUMATH 2001, Eds F. Brezzi, A. Buffa, S. Escorsaro, A. Murli, Springer, (2001), pp 437-446.

[3] N. Besse, E. Sonnendrücker *Semi-Lagrangian schemes for the Vlasov equation on an unstructured mesh of phase space* J. Comput. Phys. 191 (2) (2003) 341-376.

[4] C. K. Birdshall, A.B. Langdon, *Plasmaphysics via computer simulation* , McGraw-Hill, 1985.

[5] A. Cohen, *Numerical analysis of wavelet methods,* North-Holland, 2003.

[6] A. Cohen, S. M. Kaber, S. Müller and M. Postel, *Fully adaptive multiresolution finite volume schemes for conservation laws,* Math. Comp. 72 (2003), no. 241, 183–225.

[7] R. DeVore, *Nonlinear approximation,* Acta numerica, No. 7, Cambridge Univ. Press, Cambridge, 1998, 51-150.

[8] F. Filbet, *Numerical Methods for the Vlasov equation* ENUMATH'01 Proceedings.

[9] F. Filbet, E. Sonnendrücker, P. Bertrand, *Conservative Numerical schemes for the Vlasov equation*, J. Comput. Phys., 172-1, (2000),166-187.

[10] M. Gutnic, Ioana Paun, E. Sonnendrücker, *Vlasov simulations on an adaptive phase-space grid* to appear in Comput. Phys. Comm.

[11] S. Mallat, *A wavelet tour of signal processing.* Academic Press, 1998.

[12] T. Nakamura, T. Yabe, *Cubic interpolated propagation scheme for solving the hyper-dimensional Vlasov-Poisson equation in phase space*, Comput. Phys. Comm., 120, (1999), 122-154.

[13] E. Sonnendrücker, J. Roche, P.Bertrand, A. Ghizzo (1999) *The Semi-Lagrangian Method for the Numerical Resolution of Vlasov Equations*, J. Comput. Phys 149 (1999), 201-220.

[14] E. Sonnendrücker, F.Filbet, A. Friedman, E. Oudet, J.L. Vay *Vlasov simulation of beams on a moving phase-space grid* to appear in Comput. Phys. Comm.