Numerical methods for the reduced Vlasov equation

Nhung Pham

Thèse

INSTITUT DE RECHERCHE MATHÉMATIQUE AVANCÉE

UMR 7501

Strasbourg

présentée pour obtenir le grade de docteur de
l'Université de Strasbourg
Spécialité MATHÉMATIQUES APPLIQUÉES

Thi Trang Nhung Pham

Méthodes numériques pour l'équation de Vlasov réduite

Soutenue le 19 décembre 2016
devant la commission d'examen

Philippe Helluy, directeur de thèse
Laurent Navoret, co-encadrant
Stéphane Brull, rapporteur
Nicolas Crouseilles, rapporteur
Francis Filbet, examinateur
Raphaël Côte, examinateur

www-irma.u-strasbg.fr

UNIVERSITÉ DE STRASBOURG

ÉCOLE DOCTORALE Mathématiques,
Sciences de l'Information et de l'Ingénieur

Institut de Recherche Mathématique Avancée, UMR 7501

THÈSE présentée par :

Thi Trang Nhung PHAM

soutenue le : **19 décembre 2016**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**
Discipline / Spécialité : MATHÉMATIQUES APPLIQUÉES

Méthodes numériques pour l'équation de Vlasov réduite.

THÈSE dirigée par :
M. HELLUY Philippe Professeur, Université de Strasbourg

THÈSE co-encadrée par :
M. NAVORET Laurent Maître de Conférences, Université de Strasbourg

RAPPORTEURS :
M. BRULL Stéphane Maître de Conférences HDR, Institut de Mathématiques de Bordeaux
M. CROUSEILLES Nicolas Chargé de recherche INRIA, HDR, centre Rennes Bretagne-Atlantique

EXAMINATEURS :
M.FILBET Francis Professeur, Université de Toulouse
M.CÔTE Raphaël Professeur, Université de Strasbourg

Remerciements

Je tiens tout d'abord à remercier très chaleureusement mon directeur de thèse Philippe Helluy qui a m'offert l'opportunité de réaliser cette thèse et m'a encadrée durant ces quatres années ainsi que pendant mon Master 2 en France. Il m'a fait m'intéresser à l'analyse numérique et plus spécifiquement au sujet de cette thèse, la résolution numérique de l'équation de Vlasov pour la modélisation des plasmas. Il a su être très patient et me faire partager ses larges connaissances et son savoir des systèmes hyperboliques. Il a toujours eu de nouvelles idées lorsque j'étais bloquée et m'a motivée tout au long de ce travail. Ce travail n'aurait pas vu le jour sans les nombreuses explications et indications qu'il a pu me fournir.

Je remercie ensuite Laurent Navoret pour m'avoir encadrée, pour son enthousiasme, sa bienveillance et surtout sa patience. Il était toujours disponible lorsque j'avais des questions et n'a jamais compté ses heures. Je tiens à le remercier tout particulièrement pour sa lecture attentive de ma thèse, pour sa gentillesse et ses conseils toujours précis. Grâce à lui, je connais beaucoup de bons livres et d'articles intéressants.

Je souhaite exprimer ma gratitude à monsieur Jean-Pierre Wintenberger pour son aide dès le début de mes études en France.

Je voudrais aussi remercier Stéphane Brull et Nicolas Crouseilles qui m'ont fait l'honneur de rapporter ce manuscrit.

Je voudrais dire merci à Francis Filbet et Raphaël Côte d'avoir accepté d'être mes examinateurs.

Je remercie par ailleurs toute l'équipe Modélisation et Contrôle de l'IRMA pour la serviabilité dont fait preuve chaque membre, pour leurs conseils et leur bienveillance. Je voulais plus particulièrement exprimer ma gratitude à Pierre qui a été d'un grand secours pour résoudre tous mes problèmes informatiques et à Edwin pour son aide et ses explications lors de mon travail dans la librairie `SELALIB`. Je souhaite également remercier Michel Mehrenberger pour sa gentilesse, ses explications très détaillées sur la méthode semi-Lagrangienne ou `SELALIB`. Un grand merci au personnel administratif de l'INRIA, de l'UFR et de l'IRMA : les secrétaires, les équipes informatiques, le personnel technique et les documentalistes qui nous aident pour les tâches administratives au quotidien.

Ce travail en France a été difficile au début à cause de la barrière de la langue, mais grâce à Jonathan, Christophe, Ranine, Céline, Christelle, Diane, Amélie qui ont toujours été disponibles pour mes questions, j'arrive à être de bonne humeur pour travailler.

Pendant mes années de thèse, j'ai eu l'opportunité d'assister à des rencontres comme le congrès SMAI à Seignog, le CEMRACS à Marseille, la SEME à Nancy et à Strasbourg et la conférence NUMKIN à Munich. Je tiens donc à remercier tous ces chercheurs que j'ai pu croiser lors de ces trois ans, pour ces moments d'échange que nous avons partagés.

Merci aux autres doctorants et particulièrement Michel, Anaïs, Ambroise, Guillaume,

François, Mathieu et Simon pour leur bonne humeur et leur amitié.

Je souhaite remercier encore une fois Philippe sans qui je n'aurais pu être là aujourd'hui pour soutenir ma thèse. Et merci de tout coeur à Christophe, à Laurent et Céline pour leur aide et leur soutien. Vous êtes comme ma famille en France.

Gửi Alice, bố Dương, mẹ Hồng, em Thuỷ, những người thân yêu nhất...

Résumé

Beaucoup de méthodes numériques ont été développées pour résoudre l'équation de Vlasov car obtenir des simulations numériques précises en un temps raisonnable pour cette équation est un véritable défi. Cette équation décrit en effet l'évolution de la fonction de distribution de particules (électrons/ions) qui dépend de 3 variables d'espace, 3 variables de vitesse et du temps. L'idée principale de cette thèse est de réécrire l'équation de Vlasov sous forme d'un système hyperbolique par semi-discrétisation en vitesse. Cette semi-discrétisation est effectuée par méthode d'éléments finis. Le modèle ainsi obtenu est appelé équation de Vlasov réduite. Nous proposons différentes méthodes numériques pour résoudre efficacement ce modèle.

Dans cette thèse, nous appliquons cette démarche à différents contextes. Nous commençons par l'étude du modèle Vlasov-Poisson en 1D (chapitre 3), puis nous adaptons la méthode au modèle de Vlasov-Poisson avec la transformation de Fourier en vitesse (chapitre 4). Ensuite, nous mettons en oeuvre la méthode pour le système Vlasov-Poisson 2D dans la bibliothèque `SELALIB` (chapitre 5 et 6). Finalement, nous nous intéressons aux modèles de Vlasov-Maxwell (chapitre 7) et aux modèles Drift-Kinetic (chapitre 8) développés dans le code `schnaps`.

Première partie : Equation de Vlasov réduite

Dans cette première partie, nous décrivons le contexte de cette thèse et nous présentons la méthode de réduction aboutissant à l'équation de Vlasov réduite.

Chapitre 1

Dans ce premier chapitre, nous expliquons rapidement le contexte physique, à savoir la fusion par confinement magnétique, et les modèles considérés dans cette thèse. Nous introduisons ensuite les équations de Vlasov, Maxwell et Poisson. Nous présentons les quantités physiques et les variables en jeu ainsi que les propriétés du modèle : principe du maximum, conservation des quantités physiques du modèle, les phénomènes physiques comme la filamentation ou la dissipation d'entropie due aux collisions. Enfin, nous donnons un panorama des méthodes numériques utilisées pour la résolution du système Vlasov-Poisson, et nous justifions les choix des méthodes utilisées dans ce manuscrit.

Chapitre 2

Dans le deuxième chapitre, nous introduisons la formulation faible en vitesse pour l'équation de Vlasov avec condition aux limites imposée faiblement. Nous utilisons ensuite des éléments

finis de type Lagrange pour obtenir le modèle réduit semi-discrétisé en vitesse : c'est un système hyperbolique avec terme source correspondant au transport dans la variable de vitesse. On démontre ensuite les propriétés du modèle réduit : propriété d'hyperbolicité, stabilité en norme L^2 et conservation des quantités physiques comme la charge, l'énergie totale et la quantité de mouvement. Afin d'être complet, nous démontrons la convergence et présentons l'ordre de convergence de la méthode des éléments finis en vitesse. Nous décrivons également le calcul effectif des matrices intervenant dans le modèle réduit : elles sont calculées grâce aux formules d'intégration de Gauss, soit celle de Gauss-Legendre soit celle de Gauss-Lobatto, et nous présentons les avantages de chacune d'elles.

Deuxième partie : Simulation numérique et validation en dimension 1

Dans les chapitres 3 et 4, nous présentons des méthodes numériques pour le modèle réduit dans le cas d'une dimension : elles sont basées sur des méthodes de volumes finis en espace, avec flux visqueux, et des méthodes de Runge-Kutta en temps. Dans cette partie, nous utilisons les points de Gauss-Legendre pour le calcul des coefficients des matrices du système.

Chapitre 3

Dans ce chapitre, l'équation de Vlasov réduite est couplée à l'équation de Poisson ou celle d'Ampère. Pour l'équation de Vlasov réduite, nous utilisons la discrétisation par méthode des volumes finis en espace et un schéma Runge-Kutta (RK) en temps d'ordre $1, 2, 3$ ou 4. Nous établissons les conditions de stabilité des schémas numériques pour le transport en espace et, de manière séparée pour le terme source. Nous montrons que le schéma en temps d'ordre 3 et 4 sont toujours stable sous condition CFL tandis que les schémas d'ordre 1 et 2 nécessitent une diffusion numérique suffisante. Nous terminons ce chapitre par montrer des résultats numériques avec les cas-test classiques comme : le cas test de transport pour valider le code et l'ordre de convergence en espace, le cas test d'amortissement Landau et le cas test d'instabilité double faisceaux. Nous comparons les résultats obtenus avec ceux de la méthode PIC. Nous faisons une comparaison des temps de calcul.

Chapitre 4

Nous nous intéressons, dans ce chapitre, à l'approximation numérique pour l'équation de Vlasov obtenue après transformation de Fourier en vitesse. Nous nous restreignons au cas du système Vlasov-Poisson $1D$. L'intérêt de passer en Fourier est de mieux contrôler les oscillations présentes en vitesse. Nous introduisons d'abord l'équation de Vlasov écrite dans ces nouvelles variables, puis nous montrons que la même méthode d'approximation que précédemment permet d'écrire l'équation sous forme réduite. On utilise pour cela des conditions aux limites en vitesse adéquates. Nous appliquons donc les mêmes méthodes d'approximation qu'au chapitre précédent. Pour valider le code, nous considérons là encore les cas test d'amortissement Landau et d'instabilité double faisceaux. Nous comparons la fonctions de distribution obtenue en utilisant différents flux numériques, centré et (faiblement) visqueux, et en faisant varier les conditions aux bords en vitesse.

Troisième partie : Mise en oeuvre en dimension 2 et parallélisation

Dans cette partie, nous présentons la mise en oeuvre des schémas en dimension 2 (2D vitesse, 2D espace). Nous utilisons le schéma de volume fini présenté précédemment ainsi qu'un schéma semi-Lagrangien et nous présentons les éléments de parallélisation du code.

Chapitre 5

Nous appliquons les méthodes numériques présentées au chapitre 3 au cas $2D$ et nous discutons les difficultés associées. Nous avons mis en oeuvre la méthode dans la bibliothèque `SELALIB` (Semi Lagrangian Library) [3]. Nous décrivons la parallélisation du code effectué avec MPI (Message Passing Interface). Nous avons validé le code avec les cas-test de Landau Damping 2D. Nous présentons également les performances du code en terme de parallélisation (scalabilité) ou en nombre d'advections par seconde en l'exécutant avec un grand nombre de processeurs sur la machine Curie (Genci, TGCC).

Chapitre 6

Dans ce chapitre, les points d'interpolation de la méthode d'éléments finis en vitesse coïncident avec les points de Gauss-Lobatto. Dans ce cas, nous perdons un peu de précision dans le calcul des intégrales (pour les matrices du système réduit) mais les matrices obtenues sont diagonales (mass lumping). On peut donc facilement appliquer la méthode semi-Lagrangienne car, pour la partie de transport en espace, c'est maintenant un système d'équations d'advection à coefficients constants découplées. Après un splitting directionnel, on se ramène finalement à résoudre des équations d'advection $1D$. Nous décrivons brièvement la méthode semi-Lagrangienne et les propriétés de cette méthode. Nous étudions l'efficacité du code (nombre d'advections par seconde) et le speedup du code sur la machine Curie.

Quatrième partie : Application au système Vlasov-Maxwell et au modèle Drift-Kinetic

Dans cette partie, nous appliquons la méthodologie au système Vlasov Maxwell 2D et au modèle Drift-Kinetic. Nous utilisons les points de Gauss-Lobatto en vitesse pour diagonaliser l'advection (comme au chapitre précédent) et nous utilisons un schéma de type Galerkin-discontinu en espace.

Chapitre 7

Dans ce chapitre, nous appliquons la méthode pour résoudre numériquement le modèle de Vlasov-Maxwell en dimension 2 dans un cadre relativiste. A noter que le domaine en vitesse est maintenant un disque et que maillage en vitesse n'est plus Cartésien mais un maillage multi-patch. Nous présentons deux méthodes numériques différentes pour le système de Vlasov-Maxwell : la première méthode est basée sur le couplage entre la méthode de Galerkin discontinue pour l'équation de Maxwell et la méthode PIC pour l'équation de Vlasov. La

deuxième méthode prolonge l'idée principale de cette thèse : nous appliquons la méthode des éléments finis en vitesse puis nous utilisons un schéma de type Galerkin discontinu pour résoudre tout le système de Vlasov-Maxwell. Le schéma agit donc sur un vecteur contenant à la fois les champs électriques et magnétiques et la fonction de distribution discrétisée en vitesse et le même schéma (Galerkin-Discontinu avec flux décentré amont) est appliqué sur toute les composantes. Dans les deux cas, nous parallélisons les codes en utilisant OpenCL dans le but d'atteindre une performance élevée en exécutant les codes sur carte graphique (GPU).

Chapitre 8

Dans ce dernier chapitre, nous nous intéressons au modèle Drift-Kinetic et à un modèle simplifié, le modèle centre-guide. Nous montrons comment formellement obtenir ces modèles à partir de l'équation de Vlasov sous l'hypothèse d'un champ magnétique intense. Là encore, nous pouvons appliquer la méthode de réduction présentée dans les chapitres précédents. Afin de gagner en précision en espace, nous considérons un schéma de type Galerkin discontinu pour la partie transport en espace et élément fini pour l'équation de Poisson. Nous avons mis en oeuvre le schéma dans le code `schnaps` (Solveur pour les lois de Conservation Hyperboliques Non-linéaires Appliqué aux PlasmaS) [2]. Ce code permet de considérer des domaines courbes grâce à un maillage multi-patch. Nous avons valider le code grâce au cas test de l'instabilité du diocotron pour le modèle du centre guide : nous avons mis en évidence les erreurs dû à l'approximation de géométrie du domaine. Le code est parallélisé grâce à OpenCL : nous avons effectué des tests de scalabilité pour la partie transport et nous avons comparé les temps de calculs sur multi-CPU et sur GPU.

Publications et communications

Certains résultats de cette thèse sont déjà parus dans les publications suivantes :

- Le chapitre 3 a été développé en partie au cours du Cemracs 2012 dont une partie est le résultat du proceeding suivant :

  P. Helluy, N. Pham, and A. Crestetto. Space-only hyperbolic approximation of the Vlasov equation. ESAIM: Proceedings, 43:17–36, 2013.

- Dans le chapitre 4, on développe la méthode et on présente les résultats qui ont été présentés au congrès CANUM 2013 dont le proceeding est :

  Nhung Pham, Philippe Helluy, and Laurent Navoret. Hyperbolic approximation of the fourier transformed vlasov equation. ESAIM: Proceedings and Surveys, 45:379–389, 2014.

- Le chapitre 6 a partiellement été présenté lors du Cemracs 2014 et dans le proceeding ci-dessous :

  Sebastien Guisset, Philippe Helluy, Michel Massaro, Laurent Navoret, Nhung Pham, and Malcolm Roberts. Lagrangian/eulerian solvers and simulations for vlasov-poisson. 2015.

- Le chapitre 7 est l'article suivant :

  P. Helluy, L. Navoret, N. Pham, and A. Crestetto. Reduced Vlasov-Maxwell simulations. http://hal.archives-ouvertes.fr/hal-00957045/PDF/vlamax.pdf, March 2014.

Contents

Première partie

Equation de Vlasov réduite

Chapitre 1

Introduction

Dans cette thèse, nous nous intéressons à la simulation numérique de plasma et plus spécifiquement au plasma présent dans les dispositifs de fusion par confinement magnétique. Nous présentons dans ce chapitre le système de Vlasov-Poisson qui est une description cinétique du plasma. Nous présentons ses principales propriétés et, après un panorama des méthodes numériques employées dans la littérature, nous motivons le choix de la méthode proposée dans cette thèse.

1.1 Plasma et fusion par confinement magnétique

Un plasma est une phase de la matière constituée de particules chargées, d'ions et d'électrons. Il est généralement obtenu pour un gaz à une très haute température (vers $10^4 K$ ou plus). C'est le $4^{\text{ème}}$ état de la matière avec l'état solide, liquide et gazeux. Cet état peut se retrouver dans les tubes néons et les écrans plasma. Il se retrouve aussi dans les étoiles, dans l'ionosphère terrestre et le milieu interstellaire, il est donc répandu dans l'univers.

On peut créer de l'énergie à partir d'une perte de masse dans une réaction nucléaire selon la célèbre formule $E = mc^2$. Il existe deux types de réactions nucléaires : la réaction de fission et la réaction de fusion. La fission nucléaire consiste à générer deux noyaux d'atomes légers à partir d'un noyau plus lourd. A l'inverse, la fusion nucléaire consiste à créer un noyau lourd à partir de noyaux plus légers. La fission est utilisée dans les centrales nucléaires actuelles. La fusion nucléaire civile est encore au stade de recherche. La fusion la plus accessible est la fusion d'un noyau de deutérium et d'un noyau de tritium (tous deux isotopes de l'hydrogène) pour produire un noyau d'hélium et un neutron très énergétique (voir Fig. (1.1)). La masse des particules après la fusion est plus faible. Le défaut de masse est converti en énergie. Une très haute température est requise pour que les atomes fusionnent, à cause des forces de répulsion de Coulomb. à ces températures, les électrons se détachent des atomes pour produire un plasma.

Il y a deux façons pour confiner les réactions de fusion : le *confinement inertiel* qui consiste à projeter un faisceau laser sur une capsule de deutérium et de tritium, dans ce cas la densité est très grande dans un temps très court. Le deuxième confinement est le *confinement magnétique* qui consiste à confiner le plasma par un champ magnétique avec une densité moins élevée mais sur un temps plus long. Cela se fait dans une chambre toroïdale qui s'appelle le Tokamak (voir le modèle dans Fig. 1.1 droite). Le projet international ITER ("International Thermonuclear Experimental Reactor") entre l'Union Européenne, le Japon,

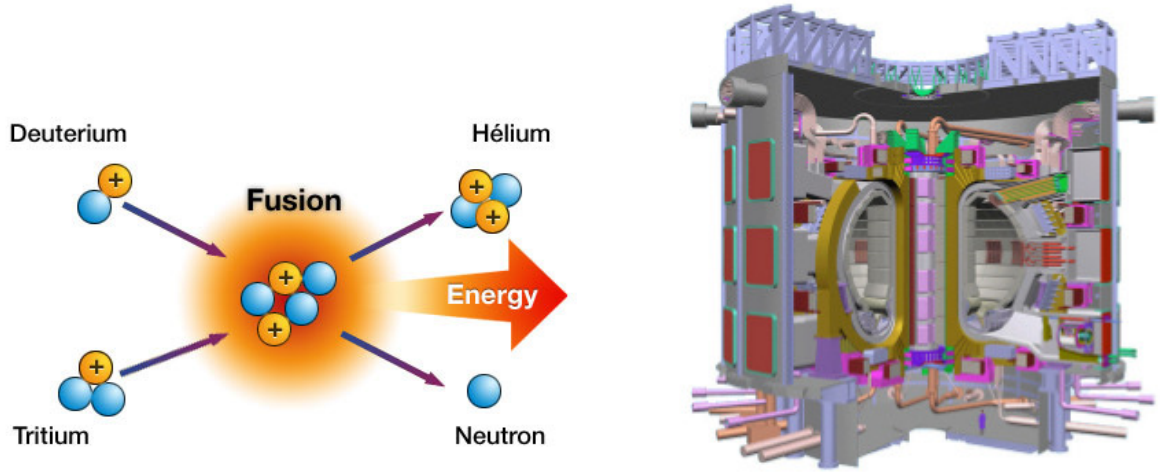


FIGURE 1.1 : Gauche : Réaction de fusion thermonucléaire d'un atome de deutérium et d'un atome de tritium. Droite : Vue d'artiste de ITER. Figures extrait à partir de [89].

la Chine, la Corée du Sud, la Russie, les Etats-Unis et l'Inde, a été signé le 21 novembre 2006 à Paris. Ce projet a pour objectif de montrer la faisabilité de la production d'électricité en utilisant le principe du confinement magnétique de la fusion nucléaire. Pour plus de détails ou d'informations, voir le site `www.iter.org`, ou (par exemple) [89].

1.2 Équations de Vlasov, de Maxwell et de Poisson

Il y a trois catégories de modèles pour d'écrire les plasmas : le modèle à N corps, les modèles cinétiques et les modèles fluides. Le modèle à N corps ou les modèles microscopiques est le modèle le plus précis car il considère l'ensemble des particules constituant le plasma et décrit les interactions des particules deux à deux. Mais le problème est que dans un plasma constitué d'un très grand nombre de particules, de l'ordre 10^{10} ou plus, décrire et calculer numériquement toutes les interactions serait très coûteux. Dans les modèles cinétiques, les particules sont représentées par leur fonction de distribution dans l'espace des phases (position/vitesse) : c'est une description statistique en vitesse. Finalement, les modèles fluides décrivent (seulement) la dynamique spatiale des quantités macroscopiques telles que la charge, le courant, l'énergie interne ou encore la température des particules.

Au centre d'un Tokamak (au centre de la coupe poloïdale), une description fluide est insuffisante et ions et électrons doivent être décrit par leur fonction de distribution. Celle-ci vérifie l'équation de Vlasov.

Dans toute la suite, nous considérerons uniquement la *distribution des électrons*. Plus précisément, comme les ions ont une masse très grande devant celle des électrons, nous supposons en première approximation que les ions sont immobiles. Excepté au chapitre 7, nous considèrerons donc que les électrons évoluent dans un fond d'ions immobiles.

1.2.1 Équation de Vlasov

L'équation de Vlasov est la suivante :

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = 0, \tag{1.2.1}$$

où $f(\mathbf{x}, \mathbf{v}, t)$ est la fonction de distribution des particules à position $\mathbf{x} = (x_1, x_2, x_3) \in \Omega_{\mathbf{x}} \subset \mathbb{R}^3$ avec la vitesse $\mathbf{v} = (v_1, v_2, v_3) \in \Omega_{\mathbf{v}} \subset \mathbb{R}^3$ et au temps $t \in \mathbb{R}^+$, q est la charge élémentaire des particules et m est la masse des particules, $\mathbf{E}$ (resp. $\mathbf{B}$) est le champ électrique (reps. le champ magnétique). La fonction de distribution est donc positive. Cette équation décrit le transport des particules dans l'espace des phases suivant la dynamique :

$$\frac{d\mathbf{x}}{dt}(t) = \mathbf{v}(t), \quad \frac{d\mathbf{v}}{dt}(t) = \frac{q}{m}\big(\mathbf{E}(\mathbf{x}, t) + \mathbf{v}(t) \times \mathbf{B}(\mathbf{x}, t)\big). \tag{1.2.2}$$

L'équation de Vlasov est une équation de transport linéaire. Elle est équivalente à la forme conservative suivante :

$$\frac{\partial f}{\partial t} + \cdot\nabla_{\mathbf{x}} \cdot \Big(\mathbf{v} f\Big) + \nabla_{\mathbf{v}} \cdot \Big(\frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B})\, f\Big) = 0, \tag{1.2.3}$$

en remarquant que le champ $\mathbf{F}(\mathbf{x}, \mathbf{v}) = \big(\mathbf{v}, \frac{q}{m}(\mathbf{E}(\mathbf{x}, t) + \mathbf{v} \times \mathbf{B}(\mathbf{x}, t)\big)^T$ est à divergence nulle :

$$\nabla_{(\mathbf{x},\mathbf{v})} \cdot \mathbf{F}(\mathbf{x}, \mathbf{v}) = 0. \tag{1.2.4}$$

Adimensionnement l'équation de Vlasov. Notant x_0, v_0 et t_0 les échelles caractéristiques d'espace, de vitesse et de temps dans notre domaine d'étude, nous introduisons les nouvelles variables suivantes :

$$t = t_0\, \tilde{t}, \quad \mathbf{x} = x_0\, \tilde{\mathbf{x}}, \quad \mathbf{v} = v_0\, \tilde{\mathbf{v}}. \tag{1.2.5}$$

Nous introduisons également les échelles caractéristiques du champ électrique E_0 et magnétique B_0 ainsi que les variables adimensionnées :

$$\mathbf{E} = E_0\, \tilde{\mathbf{E}}, \quad \mathbf{B} = B_0\, \tilde{\mathbf{B}}.$$

Notant $\tilde{f}(\tilde{\mathbf{x}}, \tilde{\mathbf{v}}, \tilde{t}) = f(\mathbf{x}, \mathbf{v}, t)$, l'équation (1.2.1) devient

$$\frac{1}{t_0}\frac{\partial \tilde{f}}{\partial \tilde{t}} + \frac{v_0}{x_0}\tilde{\mathbf{v}} \cdot \nabla_{\tilde{\mathbf{x}}} \tilde{f} + \frac{q}{m}\frac{1}{v_0}(E_0 \tilde{\mathbf{E}} + v_0 B_0 \tilde{\mathbf{v}} \times \tilde{\mathbf{B}}) \cdot \nabla_{\tilde{\mathbf{v}}} \tilde{f} = 0, \tag{1.2.6}$$

puis

$$\frac{\partial \tilde{f}}{\partial \tilde{t}} + \frac{v_0 t_0}{x_0}\tilde{\mathbf{v}} \cdot \nabla_{\tilde{\mathbf{x}}} \tilde{f} + \frac{q t_0 E_0}{m v_0}(\tilde{\mathbf{E}} + \frac{v_0 B_0}{E_0}\tilde{\mathbf{v}} \times \tilde{\mathbf{B}}) \cdot \nabla_{\tilde{\mathbf{v}}} \tilde{f} = 0. \tag{1.2.7}$$

En imposant les relations suivantes

$$v_0 = \frac{x_0}{t_0}, \quad E_0 = \frac{m v_0}{q t_0}, \quad B_0 = \frac{E_0}{v_0},$$

l'équation de Vlasov devient (en supprimant les tildes)

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = 0. \tag{1.2.8}$$

Les champs électrique et magnétique sont en partie générés par les particules elles-même. L'équation de Vlasov est alors couplée aux équations de Maxwell ou de Poisson. Le système qui en résulte est non-linéaire.

1.2.2 Équations de Maxwell, équation de Poisson

Equations de Maxwell. Les champs électrique $\mathbf{E}(\mathbf{x},t)$ et magnétique $\mathbf{B}(\mathbf{x},t)$ satisfont les équations de Maxwell

$$\frac{\partial \mathbf{E}}{\partial t} - \nabla \times \mathbf{B} = \mathbf{J}, \tag{1.2.9}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = 0, \tag{1.2.10}$$

et les lois de Gauss s'écrivent

$$\nabla \cdot \mathbf{E}(\mathbf{x},t) = \rho(\mathbf{x},t) - \rho_0 \tag{1.2.11}$$

$$\nabla \cdot \mathbf{B} = 0. \tag{1.2.12}$$

Ici, $\mathbf{J}$ est la densité de courant, définie par

$$\mathbf{J}(\mathbf{x},t) = \int_{\Omega_\mathbf{v}} f(\mathbf{x},\mathbf{v},t)\,\mathbf{v}\,d\mathbf{v}, \tag{1.2.13}$$

et ρ est la densité de charge des électrons

$$\rho(\mathbf{x},t) = \int_{\Omega_\mathbf{v}} f(\mathbf{x},\mathbf{v},t)\,d\mathbf{v}, \tag{1.2.14}$$

Comme mentionné plus haut, on considère le cas d'un fond d'ions neutralisant, dont la densité notée $\rho_0 \in \mathbb{R}$ est indépendante de $\mathbf{x}$ et t.

Équation de Poisson. Lorsque le champ magnétique $\mathbf{B}$ autoconsistant peut-être négligé, les équations de Maxwell se simplifient et l'on obtient l'équation de Poisson. D'après l'équation (1.2.10), le champ électrique dérive d'un potentiel : on peut écrire $\mathbf{E}(\mathbf{x},t) = -\nabla\Phi(\mathbf{x},t)$ avec Φ le potentiel électrique. Le potentiel satisfait

$$-\Delta\Phi(\mathbf{x},t) = \rho(\mathbf{x},t) - \rho_0 \tag{1.2.15}$$

d'après la relation (1.2.11). Il s'agit de l'équation de Poisson.

Système Vlasov-Poisson. On considère le système de Vlasov-Poisson adimensionné suivant

$$\frac{\partial f}{\partial t} + \mathbf{v}\cdot\nabla_\mathbf{x} f + \mathbf{E}\cdot\nabla_\mathbf{v} f = 0, \tag{1.2.16}$$

$$\mathbf{E} = -\nabla\Phi, \tag{1.2.17}$$

$$-\Delta\Phi = \rho - \rho_0. \tag{1.2.18}$$

Dans la suite, nous souhaitons résoudre numériquement ce système (ou le système Vlasov-Maxwell).

1.2.3 Modèle drift-kinetic

Comme nous l'avons vu au paragraphe 1.1, dans un Tokamak, un fort champ magnétique est appliqué pour confiner le plasma. Le modèle drift-kinetic est un modèle simplifié d'évolution des plasma dans tokamak qui tient compte de la gyration rapide des particules autour des lignes de champs. Dans cette thèse, nous considèrerons le modèle drift-kinetic sur un domaine spatial cylindrique (sans courbure) : le champ magnétique $\mathbf{B} = e_3$ est constant et orienté dans la direction du cylindre. Dans le modèle drift-kinetic, la fonction de distribution ne dépend que des 4 variables (x_1, x_2, x_3, v_3) et satisfait l'équation

$$\partial_t f + \mathbf{E}_\perp^\perp \cdot \nabla_{\mathbf{x}_\perp} f + v_{||} \partial_{x_{||}} f + E_{||} \partial_{v_{||}} f = 0, \tag{1.2.19}$$

où $\mathbf{E}_\perp = (E_{x_1}, E_{x_2})^T$ est la partie du champ électrique orthogonal à $\mathbf{B}$ et $E_{||} = E_3$ la partie parallèle au champ. On note $\mathbf{E}_\perp^\perp = (-E_{x_2}, E_{x_1})^T$ le vecteur perpendiculaire et les variables sont données par $\mathbf{x}_\perp = (x_1, x_2)^T$, $\mathbf{v}_\perp = (v_1, v_2)^T$ et $x_{||} = x_3$, $v_{||} = v_3$. Enfin, dans le cas d'un plasmas homogène le long du cylindre, le système de drift-kinetic 4D devient un système de centre guide 2D dans le plan poloïdal du cylindre

$$\partial_t \rho + \mathbf{E}_\perp^\perp \cdot \nabla_{\mathbf{x}_\perp} \rho = 0. \tag{1.2.20}$$

Pour les deux équations, le champs électrique doit être déterminé par une équation de Poisson (quasineutre). Pour plus de détails, nous renvoyons au chapitre 8.

1.3 Propriétés mathématiques du système de Vlasov-Poisson

1.4 Conservations

Nous présentons quelques propriétés de conservation vérifiés par les solutions du système Vlasov-Poisson. Pour le démonstration, nous renvoyons à [89]. Ces propriétés seront utiles pour étudier les méthodes numériques : on regarde si les propriétés suivantes sont satisfaites au niveau discret.

Proposition 1.4.1 (Principe du maximum)**.** *Soit $f_0(\mathbf{x}, \mathbf{v})$ la fonction de distribution initiale du système de Vlasov-Poisson, supposée positive ou nulle, et $f(\mathbf{x}, \mathbf{v}, t)$ la solution de* (1.2.16)*. On a alors*

$$0 \leq f(\mathbf{x}, \mathbf{v}, t) \leq \max_{(\mathbf{x},\mathbf{v}) \in \Omega_\mathbf{x} \times \Omega_\mathbf{v}} f_0(\mathbf{x}, \mathbf{v}).$$

Proposition 1.4.2. *Si la fonction de distribution est à décroissance suffisament rapide lorsque $|v| \to \infty$, on a l'équation de conservation de la charge*

$$\partial_t \rho + \nabla_\mathbf{x} \cdot \mathbf{J} = 0, \tag{1.4.1}$$

où la charge ρ est définie par (1.2.14) *et le courant $\mathbf{J}$ est définie par* (1.2.13)*.*

Démonstration 1.4.3. *En intégrant l'équation de Vlasov en vitesse sous sa forme conservative* (1.2.3)*, on obtient*

$$\begin{aligned} 0 &= \int_{\Omega_{\mathbf{v}}} (\partial_t f + \nabla_x \cdot (\mathbf{v} f) + \nabla_{\mathbf{v}} \cdot \mathbf{E} f) \\ &= \partial_t \left(\int_{\Omega_{\mathbf{v}}} f \right) + \nabla_x \cdot \Big(\int_{\Omega_{\mathbf{v}}} \mathbf{v} f \Big) + \int_{\Omega_{\mathbf{v}}} \nabla_{\mathbf{v}} \cdot (\mathbf{E} f) \\ &= \partial_t \rho + \nabla_{\mathbf{x}} \cdot \mathbf{J}, \end{aligned}$$

car le dernier terme s'annule (par le théorème de Gauss) du fait de la condition de décroissance à l'infini.

Proposition 1.4.4 (Conservation)**.** *Quand on considère le système de Vlasov-Poisson dans tout l'espace $\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}} = \mathbb{R}^3 \times \mathbb{R}^3$, la charge totale ρ_{tot} et l'énergie totale $\mathcal{E}_{tot}$, définies par*

$$\begin{aligned} \rho_{tot}(t) &= \int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} f(\mathbf{x}, \mathbf{v}, t)\, d\mathbf{x} d\mathbf{v}, \\ \mathcal{E}_{tot}(t) &= \frac{1}{2} \int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} \mathbf{v}^2 f(\mathbf{x}, \mathbf{v}, t)\, d\mathbf{x} d\mathbf{v} + \frac{1}{2} \int_{\Omega_{\mathbf{x}}} \mathbf{E}^2(\mathbf{x}, t)\, d\mathbf{x}, \end{aligned}$$

sont conservées au cours du temps. Le courant total $\mathbf{J}_{tot}(t)$ (coïncidant ici avec la quantité de mouvement totale des particules) défini par

$$\mathbf{J}_{tot}(t) = \int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} \mathbf{v} f(\mathbf{x}, \mathbf{v}, t)\, d\mathbf{x} d\mathbf{v}, \tag{1.4.2}$$

est aussi conservé.

On a aussi la conservation de l'intégrale en espace de toutes les fonctions de la fonction de distribution f. En particulier, les normes L^1 et L^2 (ou plus généralement la norme L^p avec un entier p tel que $1 \leq p \leq \infty$) de la fonction de distribution sont conservées

$$\begin{aligned} \|f\|_1 (t) &= \int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} |f(\mathbf{x}, \mathbf{v}, t)|\, d\mathbf{x} d\mathbf{v}, \\ \|f\|_2 (t) &= \left(\int_{\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}} f^2(\mathbf{x}, \mathbf{v}, t)\, d\mathbf{x} d\mathbf{v} \right)^{\frac{1}{2}}. \end{aligned}$$

On remarque que la fonction de distribution est positive donc la norme L^1 est aussi la charge totale. Certaines méthodes numériques produisent des valeurs négatives. Dans ce cas, il n'y a bien sûr plus équivalence entre charge et norme L^1.

Remarque 1.4.5. *Tous les propriétés pour le système de Vlasov-Poisson sont également vraies pour le système de Vlasov-Maxwell (avec quelques adaptations notamment pour l'énergie totale).*

1.4.1 Filamentation

Pour l'équation de Vlasov-Poisson, on a bien la propriété de conservation de la masse et de la norme L^2 de la fonction de distribution. Ces quantités sont conservées globalement

et permettent à des échelles fines de se développer dans l'espace des phases. Par exemple, considérons l'équation de transport cinétique (équation de Vlasov sans force électrique)

$$\partial_t f + \mathbf{v}\partial_{\mathbf{x}} f = 0 \tag{1.4.3}$$

Grâce à la méthode des caractéristiques, la solution est donnée par

$$f(\mathbf{x}, \mathbf{v}, t) = f_0(\mathbf{x} - \mathbf{v}t, \mathbf{v}). \tag{1.4.4}$$

Donc, si on considère la condition initiale $f_0(\mathbf{x}, \mathbf{v}) = (1 + \cos(k\mathbf{x}))\, e^{-\mathbf{v}^2/2}$, on obtient

$$f(\mathbf{x}, \mathbf{v}, t) = (1 + \cos(k(\mathbf{x} - \mathbf{v}t)))e^{-\mathbf{v}^2/2}. \tag{1.4.5}$$

Les oscillation en vitesse augmentent donc quand $kt \to \infty$ donc quand $t \to \infty$. Sur la figure 1.2, on a tracé la solution dans l'espace des phases.

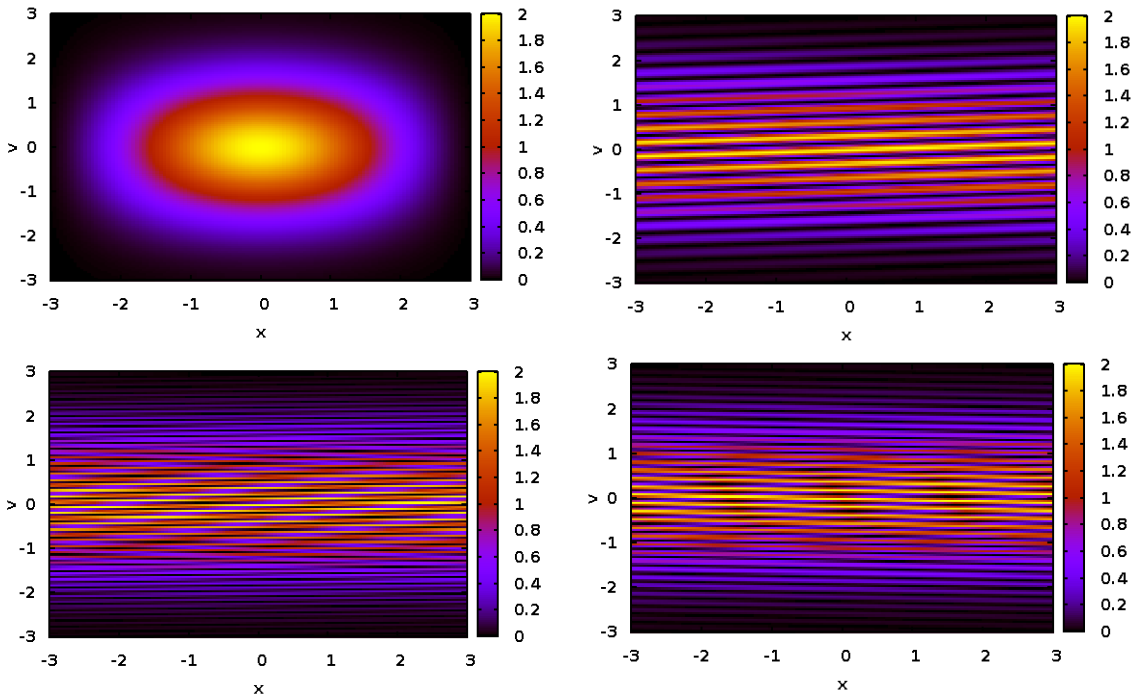


FIGURE 1.2 : La solution au temps $t = 0$ (haut gauche), $t = 10$ (haut droit), $t = 50$ (bas gauche), $t = 100$ (bas droit) avec $k = 1$, $nx = nv = 128$, $x_{\max} = 2\pi$, $v_{\max} = 5$.

1.4.2 Dissipation d'entropie

Dans l'équation de Vlasov 1.2.1, on a négligé les collisions entre les particules. Dans le cas où il y a des collisions, on considère l'équation de Vlasov-Boltzmann :

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = \mathcal{Q}(f). \tag{1.4.6}$$

L'opérateur de collisions de Boltzmann $\mathcal{Q}(f)$ est un opérateur quadratique agissant seulement sur la variable de vitesse $\mathbf{v}$, ce qui correspond à l'hypothèse de collisions localisées en espace. Il s'écrit :

$$\mathcal{Q}(f)(\mathbf{v}) = \frac{1}{m}\int_{\Omega_{\mathbf{v}}}\int_{S^2}\mathcal{B}(|\mathbf{v}-\mathbf{v}_1|,\theta)\Big[f(\mathbf{v}')f(\mathbf{v}_1')-f(\mathbf{v})f(\mathbf{v}_1)\Big]d\mathbf{v}_1 dn. \tag{1.4.7}$$

où $\mathbf{v}', \mathbf{v}_1'$ sont les vitesses de particules après collisions et $\mathbf{v}, \mathbf{v}_1$ sont les vitesses des particules avant collision. Elles vérifient :

$$\begin{cases}\mathbf{v}' = \dfrac{\mathbf{v}+\mathbf{v}_1}{2} + \dfrac{|\mathbf{v}-\mathbf{v}_1|}{2}\theta,\\ \mathbf{v}_1' = \dfrac{\mathbf{v}+\mathbf{v}_1}{2} - \dfrac{|\mathbf{v}-\mathbf{v}_1|}{2}\theta.\end{cases} \tag{1.4.8}$$

où θ désigne l'angle entre $\mathbf{v}-\mathbf{v}_1$ et $\mathbf{v}'-\mathbf{v}_1'$, n est le vecteur normal dans la sphère S^2 et $\mathcal{B}$ est le noyau de collision. Pour les collisions de type sphères dures, le noyau de collision est donné par :

$$\mathcal{B}(|\mathbf{v}-\mathbf{v}_1|,\theta) = |\mathbf{v}-\mathbf{v}_1|.$$

On introduit l'entropie :

$$H(t) = \int_{\Omega_{\mathbf{x}}}\int_{\mathbb{R}^3} f(\ln f - 1)\, d\mathbf{v}d\mathbf{x}.$$

C'est une fonctionnelle convexe. Nous avons tout d'abord la propriété de dissipation d'entropie (voir [89]) :

Proposition 1.4.6. ***Inégalité de Boltzmann*** *Toute fonction de distribution $f(\mathbf{x},\mathbf{v},t) > 0$ satisfait :*

$$\int_{\mathbb{R}^3}\mathcal{Q}(f)\,\ln f\, d\mathbf{v} \leq 0$$

De plus :

$$\int_{\mathbb{R}^3}\mathcal{Q}(f)\,\ln f\, d\mathbf{v} = 0 \quad \Leftrightarrow \quad \exists\,\rho\geqslant 0, \mathbf{u}\in\mathbb{R}^3, T\geqslant 0,\ f(\mathbf{v}) = \frac{\rho}{(2\pi T)^{\frac{1}{2}}}\exp\left(-\frac{(\mathbf{v}-\mathbf{u})^2}{2T}\right).$$

Proposition 1.4.7. ***"Théorème H"*** *Soit $f(\mathbf{x},\mathbf{v},t) > 0$ une solution de l'équation de Vlasov-Boltzmann. L'entropie satisfait :*

$$\frac{dH}{dt}(t) \leq 0.$$

Le théorème H établit donc que l'entropie est une fonction de Lyapunov (voir [60]). Le cas d'égalité dans le théorème (1.4.7) permet de montrer (formellement) que la fonction de distribution relaxe vers les distributions Maxwelliennes en vitesse :

$$M_{\rho,u,T} = \frac{\rho}{(2\pi T)^{\frac{1}{2}}}\exp\left(-\frac{(\mathbf{v}-u)^2}{2T}\right). \tag{1.4.9}$$

On peut également considérer un opérateur simplifié qui assure également la conservation de la masse, de la quantité de mouvement, de l'énergie cinétique : c'est l'opérateur de collision BGK (pour Bhatnagar, Gross et Krook) (voir [89, 13]) :

$$Q_{BGK}(f) = \nu \left(M_{\rho_f, u_f, T_f} - f \right).$$

où ρ_f, u_f et T_f sont la densité de charge, la vitesse moyenne et la température associée à f. En physique des plasma, on considère également l'opérateur non-linéaire de Fokker-Planck-Landau (voir [89, 64]).

1.5 Méthodes numériques employées

1.5.1 Méthodes numériques pour le système Vlasov-Poisson

De nombreuses méthodes numériques ont été développées pour résoudre le système de Vlasov-Maxwell ou le système de Vlasov-Poisson. Il y a principalement 3 familles de méthodes numériques classiques pour ces équations : les méthodes PIC (Particle In Cell), les méthodes Eulériennes (comme la méthode Galerkin Discontinu ou la méthode de volumes finis ou différents finis) et les méthodes semi-Lagrangiennes.

Méthodes PIC Les approches PIC (voir [14, 98]) sont des méthodes très utilisées en physique des plasmas, car elles permettent d'effectuer des simulations dans des configurations complexes avec une relativement faible quantité de mémoire et de ressources CPU. Toutefois, la méthode PIC est basée sur un choix aléatoire initial des particules et donc présentent des bruits numériques. Cette méthode a aussi des difficultés à représenter correctement la solution dans les zones où il y a peu de particules. De plus, il est difficile d'assurer la conservation de l'énergie avec cette méthode. Nous mentionnons une méthode numérique développée récemment [81] qui est intermédiaire entre la méthode PIC et la méthode semi-Lagrangienne discutée ci-dessous.

Méthodes Semi-Lagrangienne Cette méthode repose sur le fait que la fonction de distribution est constante le long des caractéristiques. C'est une méthode très intéressante d'un point de vue pratique car il n'y a pas de condition de stabilité : on peut donc prendre un pas de temps aussi grand que souhaité. C'est une méthode qui peut être également d'ordre élevée donc elle fait l'objet de développement important actuellement : elle est par exemple utilisée dans le code Gysela développé au C.E.A. [65]. Les références de base de cette méthode sont [18, 91].

Différentes versions de la méthode semi-Lagrangienne ont été développées. La méthode semi-Lagrangienne classique est basée sur l'évaluation ponctuelle de la fonction de distribution le long des caractéristiques et de différentes types d'interpolation [28]. Des méthodes de type volume fini évaluant la fonction de distribution sur les cellules (du maillage) ont également été développées [41, 101]. Elle permettent d'obtenir plus directement les propriétés de conservation de la masse et du principe du maximum. Une dernière version de la méthode semi-Lagrangienne repose sur une formulation de type Galerkin discontinu [93]. La plupart des méthodes semi-Lagrangienne sont définies en une dimension et sont étendues au problème bidimensionnel (advection en x et v) par la méthode de splitting directionnel : elles sont donc définies sur un maillage cartésien de l'espace des phases.

Méthodes Eulérienne La méthode semi-Lagrangienne est précise et sans condition de stabilité, mais les propriétés de conservation peuvent être délicates à assurer et les conditions de maillage Cartésien contraignant pour traiter les domaines courbés. Par conséquent, les méthodes Eulériennes pour résoudre des équations cinétiques sont de plus en plus populaires. Bien que soumises à des contraintes de stabilité (de type CFL), elles permettent un contrôle des propriétés de conservation et une précision importante.

Parmis ces méthodes, la méthode Galerkin discontinue a fait l'objet d'une attention plus particulière. C'est une méthode de type variationnelle (comme la méthode élément fini) où l'on approche les fonctions par des fonctions polynomiales par morceaux et où les discontinuités entre chaque élément du maillage nécessite la définition d'un flux (comme la méthode volume fini). La méthode est présentée plus en détail dans les chapitres 7 et 8. Elle a été appliqué au système Vlasov-Poisson dans les travaux suivants [30, 11, 31],[20, 19] ou encore [73].

1.5.2 Choix de la méthode

Défis. Les méthodes numériques pour le système de Vlasov-Poisson ou Vlasov-Maxwell sont confrontées aux difficultés suivantes :

1. **Coût de calcul et parallélisation :** en dimension 3, la fonction de distribution des particules dépend de 6 variables : 3 en position $\mathbf{x} = (x_1, x_2, x_3)$, 3 en vitesse $\mathbf{v} = (v_1, v_2, v_3)$. Si l'on souhaite N points de discrétisation dans chaque direction, il faut donc donc déterminer la dynamique temporelle de N^6 inconnues. Le calcul est donc très coûteux et des méthodes de parallélisation doivent être développées pour réduire le temps de calcul. Par ailleurs, les modèles gyrocinétiques, qui tiennent compte de la physique du problème dans les Tokamaks, ont été introduit pour réduire la dimension du système (en passant à 5 variables).

2. **Précision et ordre élevé :** du fait du transport cinétique et de la filamentation, des échelles d'oscillation très petites peuvent apparaître en vitesse (voir paragraphe 1.4.1). De plus, comme on le verra au chapitre 8, la dynamique spatiale dans les plans poloïdaux du Tokamak est similaire à une dynamique d'Euler incompressible et donc peut générer de la turbulence spatiale. Il faut donc utiliser des maillages relativement fins et cela nécessite de développer des méthodes d'ordre élevées pour obtenir une précision correcte.

3. **Géométrie :** Les méthodes numériques doivent pouvoir traiter le cas de la géométrie toroïdale d'un Tokamak. Des maillages curvilignes [72, 92] ou Cartésiens [65, 43] (avec condition aux bords adéquates) sont utilisés dans la littérature. Pour diminuer les sources d'erreurs numériques, le maillage devrait être dans l'idéal aligné avec les lignes de champ magnétique et donc courbé. A noter également que le maillage en vitesse peut également être courbé dans le cas relativiste où l'espace des vitesse est la sphère (cas considéré au chapitre 7).

Nos choix de méthodes numériques. Dans cette thèse, nous proposons d'écrire l'équation de Vlasov sous forme d'un système hyperbolique

$$\mathcal{M}\partial_t \mathbf{w} + \sum_{k=1}^{3} A^k \partial_{x_k} \mathbf{w} + B(\mathbf{E})\mathbf{w} = 0, \tag{1.5.1}$$

après application d'une méthode d'élément fini dans la variable de vitesse : $\mathbf{w}(x,t) \in \mathbb{R}^{N_v}$ représente le vecteurs des degrés de libertés en vitesse et $\mathcal{M}, A^k, B(\mathbf{E}) \in M_{N_v}(\mathbb{R})$ sont des matrices qui seront définies dans la proposition 2.4.1. Nous détaillons la méthode dans le chapitre suivant et les propriétés du système obtenu. Ceci permet d'appliquer des méthodes Eulériennes pour résoudre le système hyperbolique. Nous utiliserons des méthodes

- de type volume fini (chap. 3, 4, 5),
- de type Galerkin discontinu (chap. 7 et 8),
- de type semi-Lagrangienne (chap. 6).

Cette méthodologie de type Galerkin discontinu a l'avantage de permettre d'être à la fois *conservative* sur les quantités macroscopiques (charge, courant), dissipative en norme L^2 et de pouvoir naturellement traiter des géométries complexes. Bien que soumise à une contrainte de stabilité de type CFL, la méthode gagnent en efficacité en étant d'ordre élevé en vitesse et en espace et en étant parallélisable. Pour cela, nous utiliserons des maillages structurés (déformations de maillage cartésien) et multipatch pour décrire les domaines de calculs. Nous utiliserons des programmation de type MPI ou OpenCL pour les parallélisation multi-CPU ou GPU (carte graphique). Enfin, nous verrons que l'avantage de se formalisme est de pouvoir traiter avec une seule méthode et donc un seul code plusieurs types de modèle (système Vlasov-Poisson, sytème Vlasov-Maxwell, système drift-kinetic).

Chapitre 2

Reduced Vlasov model

In this chapter, we explain how to obtain the Vlasov reduced model. It is based on a semi-discretization of the Vlasov equation by the finite elements method (FEM) with respect to the velocity variable. To this aim, we introduce the weak formulation with respect to the velocity variable and then the FEM (see [39, 45]). In this way, we construct a family of reduced models, depending on the velocity discretization parameter N_v. For low values of N_v, we recover fluid behavior, while high values of N_v allow precise approximations of the kinetic model. For a fixed velocity parameter N_v, the unknown depends on space and time instead of the full phase-space variables. The approximate model is thus a linear hyperbolic system, with non-linear source terms. It is possible to establish conservation and entropy properties. These two features lead to a simpler way to reuse existing solvers for hyperbolic equations. It is possible to incorporate in this way plasma kinetic models into a general, highly optimized solver. The coupling to other fluid plasma models is also simplified. We also present properties of the reduced Vlasov equation such as : the conservation, the convergence.

2.1 Collision and reduction model

As regards the distribution function in velocity, filamentation and entropy dissipation act on opposite direction and counterbalance : while entropy dissipation regularizes the velocity distribution, filamentation results in increasing the large frequencies in velocity. However, as it is mentionned in [49], there exists some evidence that plasma in tokamak chamber have a quite smooth distribution with respect to the kinetic velocity variable. As we will see in chapter 8, the kinetic velocity variable corresponds to the velocity component parallel to the magnetic field line. Although the distribution in velocity is smooth, the distribution has not reached Maxwellian equilibria and the kinetic description is still required.

We are thus interested in approximating the distribution function in velocity with a **small** number of parameters. This methodology enters the class of reduction method. Note that this method will not be relevant when the filamentation is too strong : in that case, the required number of parameters will increase drastically (see Landau damping test cases).

We here present some possible choice of parametrization :

- **Fourier basis.** A discretization of the continuous Fourier transform in velocity is used in [62, 37]. Filtering techniques or absorbing boundary conditions are designed to make

a correct the large frequencies (when they leave the computational domain). See also chapter 4.

- **Orthogonal polynomial basis.** Hermite polynomials are mostly considered as they corresponds to the Maxwellian distribution weight. Writing the equations in this basis results in a hierarchy of equations. Numerically, the hierarchy is stopped by imposing the coefficients to eventually be zero. We refer to [88, 67] and also to [33] for analytical results using this decomposition.

- **Moment method.** The moment methods consist in describing the distribution function as a function of some first moments in velocity (for instance the k-th first ones). This requires to close the set of equations by expressing the next moment (the $k+1$-th one) as function of the other. But to be well-posed, this closure should satisfy entropy dissipation properties. This program was successfully carried out in [36] for the radiative transfer equation (where the velocity space is the sphere) and extended to plasma models in [74].

Note that the two first parametrization are linear, while the third one is non-linear (the moments parametrize a submanifold). In our work, we will consider a **piece-wise polynomial basis** (or finite element basis). As we explain below, this choice can be regarded as a reduction method since we control (and potentially decrease) the L^2 entropy of the distribution function.

Entropy variable. We consider a kinetic entropy $S(f)$: this is a smooth strictly convex function of f. By multiplying equation (1.4.6) with $S'(f)$, we then obtain

$$\partial_t S(f) + \mathbf{v} \cdot \nabla_{\mathbf{x}} S(f) + \mathbf{E} \cdot \nabla_{\mathbf{v}} S(f) = \mathcal{Q}(f) S'(f). \tag{2.1.1}$$

where $\mathcal{Q}(f)$ is the Boltzmann operator. We introduce the entropy variable g, which is related to f by the relations :

$$g = S'(f), \quad f = S^{*'}(g),$$

where S^* is the Legendre transform of S

$$S^*(g) = \max_f (gf - S(f)).$$

The kinetic entropy can be the physical entropy $S(f) = f(\ln f - 1)$ and then $f = \exp(g)$, or the quadratic function $S(f) = f^2/2$ and $g = f$. Other entropies can be considered (see for instance [46]). Note that the Boltzmann entropy $\mathcal{Q}$ introduced in (2.1.1) is the integral in velocity of the physical kinetic entropy $S(f) = f(\ln f - 1)$.

Approximation. Now, we consider arbitrary basis functions $\{\varphi_i\}_{1 \leq i \leq N_v}$ in velocity and we expand the entropy variable g in the basis $\{\varphi_i\}_i$. Suppose that

$$g(\mathbf{x}, t, \mathbf{v}) = \sum_{k=1}^{N_{\mathbf{v}}} g_k(\mathbf{x}, t) \varphi_k(\mathbf{v}). \tag{2.1.2}$$

We denote by Π the orthogonal projection of g on a subspace $\mathcal{V}_0$ of the space $\mathcal{V} = \text{span}\{\varphi_i,\ 1 \leq i \leq P\}$. We can then replace the Boltzmann operator by the linear operator kernel

$$Q(f) = \lambda\,(\Pi g - g)\,. \tag{2.1.3}$$

where $\lambda > 0$ is a given constant. If f has a compact support in $\mathbf{v}$ (or a fast decay), this operator has the following properties

- [Moment conservative] $\int_{\Omega_\mathbf{v}} Q(f)\varphi = 0, \quad \forall \varphi \in \mathcal{V}_0$. In particular, if $\mathbf{v} \to \mathbf{v}^m$ is in the space $\mathcal{V}_0$ then the $m-$moment of f is conserved in time

$$\int_{\Omega_\mathbf{x}} \int_{\Omega_\mathbf{v}} f\mathbf{v}^m = Cst.$$

- [Entropy dissipative] The total entropy $\Sigma = \int_{\Omega_\mathbf{v}} S(f)\,d\mathbf{v}$ is decreasing

$$\partial_t \Sigma + \partial_\mathbf{x} G(\Sigma) \leq 0, \quad \text{with } G(\Sigma) = \int_{\Omega_\mathbf{v}} \mathbf{v} S(f).$$

The collision kernel (C.3) can be used of course for introducing a physical phenomenon. But we can also use it as a numerical tool for damping numerical oscillations.

Therefore, the only (mathematical) requirement on the velocity basis is to contain low order polynomials. Note that it is the velocity basis for the kinetic entropy variable.

In this thesis. In the following, we consider the L^2 entropy and thus the entropy variable coincides with the distribution function f. In order to have conservation of the moments (density, momentum), we choose a continuous finite-element basis on a bounded domain (it thus contains low order polynomials). In this thesis, we do not consider the collision operator $Q(f)$. It will be the subject of further work. However, it has to be noted that numerical diffusion will anyway produce L^2 entropy dissipation.

2.2 Boundary condition, classical formulation and weak formulation

In this section, we consider the Vlasov-Boltzmann equation in three-dimensional space

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_\mathbf{x} f + \mathbf{E} \cdot \nabla_\mathbf{v} f = 0, \tag{2.2.1}$$

where the electric field $\mathbf{E}(x,t)$ is a given data.

Boundary condition

In the following, the spatial domain is given by $\Omega_\mathbf{x} =]0, L[^3$, with $L > 0$, and we consider periodic boundary conditions at $\mathbf{x}_i = 0$ and $\mathbf{x}_i = L$, $i = 1, 2, 3$ for the distribution function and the electric potential. Other spatial boundary conditions could be considered.

For the physical problem, the velocity domain is unbounded, $\Omega_{\mathbf{v}} = \mathbb{R}^3$, and the distribution function vanishes at $v_i = \pm\infty$, for $i = 1, 2, 3$. However, for numerical reasons, we bound the velocity space. We thus consider the bounded domain $\Omega_{\mathbf{v}} =]-V_{\max}, V_{\max}[^3$, where $V_{\max} > 0$ is the maximal velocity. The Vlasov equation (2.2.1) is a transport equation, we thus also apply upwind boundary conditions on $\partial\Omega_{\mathbf{v}}$ depending if the electric field $\mathbf{E} = -\nabla_{\mathbf{x}}\Phi$ is pointing inside or outside $\Omega_{\mathbf{v}}$. Let v be a velocity on $\partial\Omega_{\mathbf{v}}$ and $n_{\mathbf{v}}$ be the outward normal unit vector on $\partial\Omega_{\mathbf{v}}$ at v, then the boundary condition reads

$$f(\mathbf{x}, \mathbf{v}, t) = 0, \quad \text{if } \mathbf{E}(\mathbf{x}, t) \cdot n_{\mathbf{v}} \leq 0, \tag{2.2.2}$$

or equivalently

$$(\mathbf{E}(\mathbf{x}, t) \cdot n_{\mathbf{v}})^- f(\mathbf{x}, \mathbf{v}, t) = 0, \tag{2.2.3}$$

where, for any $a \in \mathbb{R}$, $a^- = \min(0, a)$. Note that there is a priori no charge conservation in the bounded sub-domain $\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}$. Consequently, the total charge in the domain $\Omega_{\mathbf{x}} \times \Omega_{\mathbf{v}}$, ρ_0 (defined in the section 1.2.2), is now time varying.

Classical formulation

Consider equation (2.2.1) with respect to the velocity variable. Consider the space homogeneous and semi-discrete in time problem. The classical formulation of the problem reads : Find the function $f \in C^1(\Omega_{\mathbf{v}}) \cap C(\overline{\Omega}_{\mathbf{v}})$ that satisfies equation (2.2.1) with the boundary condition defined by (2.2.3).

With numerical purpose in mind, we replace the classical formulation (2.2.1)-(2.2.3) by a weak formulation (also called variational formulation).

Weak formulation

The weak formulation with weakly imposed boundary conditions reads : find $f(\mathbf{x}, ., t) \in V = \{f \in L^2(\Omega_{\mathbf{v}}), \mathbf{E} \cdot \nabla_{\mathbf{v}} f \in L^2(\Omega_{\mathbf{v}})\}$

$$\partial_t \int_{\Omega_{\mathbf{v}}} f\varphi + \nabla_{\mathbf{x}} \cdot \int_{\Omega_{\mathbf{v}}} \mathbf{v} f\varphi + E \cdot \int_{\Omega_{\mathbf{v}}} \nabla_{\mathbf{v}} f\varphi - \beta \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E} \cdot n_{\mathbf{v}})^- f\varphi = 0, \tag{2.2.4}$$

with any test function $\varphi \in V$ and where β is a positive constant.

Remark 2.2.1. *This weak form of the transport equation introduces an upwinding only on the boundary $\partial\Omega_{\mathbf{v}}$. In our work, we take $\beta = 1$ (in chapter 5 and 6) or $\beta = 1/2$ (in chapter 3). When $\beta = 1$, we obtain the weak formulation of the discontinuous Galerkin method (see [24]). The advantages and drawbacks of this approach are discussed in [58].*

Proposition 2.2.2. *For any classical solution, the weak formulation (2.2.4) is equivalent to the initial problem (2.2.1) supplemented with the boundary condition (2.2.3).*

Démonstration. Indeed, if f is a solution of (2.2.1) with the conditions (2.2.3), it is obvious that f is also a solution of (2.2.4). Reciprocally, if we suppose that f is a solution of (2.2.4), because (2.2.4) is true for arbitrary test function φ. Thus for every function φ such that $\varphi = 0$ at the boundary $\partial\Omega_{\mathbf{v}}$ we obtain

$$\partial_t \int_{\Omega_{\mathbf{v}}} f\varphi + \nabla_x \cdot \int_{\Omega_{\mathbf{v}}} \mathbf{v} f\varphi + \mathbf{E} \cdot \int_{\Omega_{\mathbf{v}}} \nabla_v f\varphi = 0 \tag{2.2.5}$$

and thus (2.2.1). Finally, if we take test functions φ that do not vanish at $\partial\Omega_{\mathbf{v}}$, we then obtain the boundary condition (2.2.3). □

Remark 2.2.3. *The weak formulation for the steady advection equation is well-posed. (see for instance the proof in chapter 2 of [34]).*

2.3 Lagrange finite element basis in velocity

We now describe briefly the finite element method (FEM). It consists in replacing the Hilbert space V of the weak formulation(2.2.4) by a finite dimensional sub-space of finite dimension V_h. The construction of this sub-space is based on a discretization of the domain $\Omega_{\mathbf{v}}$ with a mesh. We then look for an approximate solution that belongs to subspaces of order P on each mesh element. We now recall some definitions. For more detail, we refer to [39, 40].

We suppose that the dimension of space we work in is $k \in \mathbb{N}^*$, namely $\Omega_{\mathbf{v}} \subset \mathbb{R}^k$. In this thesis, we consider the two cases : $k = 1, 2$ (see the following chapter). We present some basic concept on the FEM and we detail the the construction of the Lagrange FEM which is used in this thesis.

2.3.1 Reference element

After choosing a mesh, the second step consists in defining an approximation space in each mesh element. In this thesis, we consider the tensorial polynomial finite element space $\mathbb{Q}_{\mathrm{d}}$, with $\mathbf{d} \in \mathbb{N}$, defined by

Definition 2.3.1. *Let* $\mathrm{d} \in \mathbb{N}$. *The polynomial space* $\mathbb{Q}_{\mathbf{d}}$ *with real coefficients and of degree at most* d *in each variable, is defined by*

$$\mathbb{Q}_{\mathrm{d}} = \left\{ \sum_{0 \le i_1, \dots, i_k \le d} \alpha_{i_1 \dots i_k} v_1^{i_1} \dots v_k^{i_k} \quad \textit{with} \quad v = (v_1, v_2, \dots, v_k), \quad \alpha_{i_1 \dots i_k} \in \mathbb{R} \right\}$$

Note that the dimension of $\mathbb{Q}_{\mathrm{d}}$ is $(\mathrm{d}+1)^k$. The $\mathbb{Q}_{\mathrm{d}}$ finite element method is based on the discrete space

$$V_h = \left\{ v \in C(\overline{\Omega}_{\mathbf{v}}) \quad \text{s.t} \quad v_{|\mathcal{D}_i} \in \mathbb{Q}_{\mathrm{d}} \right\}.$$

Proposition 2.3.2. *The space* V_h *is a sub-space of* $H^1(\Omega_{\mathbf{v}})$.

The next step is to construct a basis of V_h. In this thesis, we study the case of Lagrange finite element. For others cases, e.g., Hermite finite element, see [39, 40]. Since the velocity domain may have curved boundaries (see the relativistic case, chap. 7), we consider a reference finite element on which it is easy to construct a basis of the finite element. Once we have the reference finite element, we then construct the geometric transformations (also called mapping) that maps reference element onto each mesh element.

Reference finite element. Let us consider the cuboid reference element

$$\hat{Q} = [-1, 1]^k.$$

As we want to consider polynomials with degree at most d in each variable, we consider $(\mathrm{d}+1)^k$ reference nodes ($(\mathrm{d}+1)$ in each direction)

$$\begin{aligned} &\forall\ 1 \leq K \leq (\mathrm{d}+1)^k, \\ &\hat{N}_K = (\xi_{I_1}, \ldots, \xi_{I_k}), \quad \text{with} \quad 1 \leq I_m \leq \mathrm{d}+1 \quad \text{for each} \quad m \in \{1, ..., k\}. \end{aligned} \tag{2.3.1}$$

Note that, for a given $m \in \{1, \ldots, k\}$, (ξ_{I_m}) is a $(\mathrm{d}+1)$-subdivision of the reference interval $[-1, 1]$. The relation between the index K and $(I_1, \ldots, I_k)$ is given by the function

$$K = \sum_{i=1}^{k} (\mathrm{d}+1)^{i-1}(I_i - 1) + 1, \quad 1 \leq K \leq (\mathrm{d}+1)^k. \tag{2.3.2}$$

At each node $\hat{N}_K$ is associated the K- th k-dimensional Lagrange polynomial defined by

$$\hat{L}_K(\hat{v}) = \prod_{m=1...k} \ell_{l_m}(\hat{v}_m), \tag{2.3.3}$$

where $(\ell_{\mathbf{i}})$ denotes the one-dimensional Lagrange polynomial given by

$$\ell_{\mathbf{i}}(\xi) = \prod_{j \neq \mathbf{i}} \frac{\xi - \xi_t}{\xi_{\mathbf{i}} - \xi_j}. \tag{2.3.4}$$

The Lagrange polynomials form a basis of the space $\mathbb{Q}_{\mathrm{d}}$ (def. 2.3.1) : Any element of V_h can be characterized by its values at the nodes. Therefore, we can easily defined a basis of $\mathcal{L}(V_h, \mathbb{R})$ by choosing the $(\mathrm{d}+1)^k$ linear forms $\tau_i : C(\hat{Q}) \to \mathbb{R}$, with $i = 1 \ldots (\mathrm{d}+1)^k$ such that

$$\tau_i(u) = u(\hat{N}_i), \quad \text{for all} \quad u \in V_h(\hat{Q}).$$

In the Lagrange finite elements, they correspond to the degree of freedom of the problem.

2.3.2 Velocity mesh

Mesh

Firstly, we choose a mesh to discretize the domain $\Omega_{\mathbf{v}}$. We suppose that the domain $\Omega_{\mathbf{v}}$ is divided into $M \in \mathbb{N}$ disjoint non-degenerate subdomains $\mathcal{D}_i$ $(1 \leq i \leq M)$ such that

$$\overline{\Omega}_{\mathbf{v}} = \cup_{i=1}^{M} \overline{\mathcal{D}}_i.$$

We denote $\mathcal{D}$ the set of the subdomains of $\Omega_{\mathbf{v}}$

$$\mathcal{D} = (\mathcal{D}_i)_{i=1...M},$$

then, each $\mathcal{D}_i$ is called a mesh element. We then define the diameter of each mesh element $\mathcal{D}_i$ by

$$\mathrm{diam}(\mathcal{D}_i) = \max_{x,y \in \mathcal{D}_i} \|x - y\|.$$

then, the meshsize of $\mathcal{D}$ is given by

$$h = \max_{i=1...M} \mathrm{diam}(\mathcal{D}_i).$$

We also use the notation $\mathcal{D}_h$ to indicate a mesh with meshsize equal h.

For each mesh element $\mathcal{D}_i$, we also define its roundness r which is the diameter of the biggest ball contained in $\mathcal{D}_i$

$$r(\mathcal{D}_i) = \max_{B_r \subset \mathcal{D}_i} (2r).$$

Definition 2.3.3. *A sequence of meshes $(\mathcal{D}_h)_{h>0}$ of $\Omega_{\mathbf{v}}$ is a sequence of regular meshes if*

- *The sequence $\{h\}$ tend to 0*
- *There exists a constant σ such that for all $h > 0$ and for all $\mathcal{D}_i \in \mathcal{D}$ we have*

$$\frac{diam(\mathcal{D}_i)}{r(\mathcal{D}_i)} \leq \sigma$$

Example In this thesis, we will consider mapped cartesian meshes. They always satisfy the regularity assumption. In particular case, when $k = 1$, a mesh $\mathcal{D}_h$ of $\Omega_{\mathbf{v}} =]a, b[$, $a > b$ is defined by $\mathcal{D}_i = [x_i, x_{i+1}]$ where

$$a = x_0 < x_1 < \cdots < x_{M+1} = b$$

We call x_i $(i = 0 \cdots M + 1)$ the vertices of the mesh and the intervals $\mathcal{D}_i$ are called the cells or elements of the mesh. The meshsize h in this case refers to the refinement level and then defined by

$$h = \max_{0 \leq i \leq M} (x_{i+1} - x_i).$$

Such mesh $\mathcal{D}_h$ is said to be uniform if and only if the distant between two continous cell are equal, i.e $h = (x_{i+1} - x_i)$ for all $0 \leq i \leq M$.

In practice

From the reference element, we now construct the finite element in the whole domain in velocity. This domain is supposed to be also a cuboid domain $\Omega_{\mathbf{v}} =]-V_{\max}, V_{\max}[^k$ or a mapped cuboid domain, in order to treat curved domains for the relativistic model (see chap.7). We then suppose that we can define a Cartesian mesh of $\Omega_{\mathbf{v}}$: we have

$$\Omega_{\mathbf{v}} = \cup_{r=1 \cdots M^k} Q_r,$$

where the M^k subdomains are the "tensorization" of M interval in each direction. Each subdomain Q_r, for $r \in \{1, \ldots, M^k\}$, is supposed to be the mapping of the reference element : we denote τ_r the transformation that maps the subdomain $\hat{Q}$ onto Q_r. Consequently, each node of the reference element has its images in each subdomain. This defines $N_v = (\mathrm{d}M+1)^k$ nodes in the whole domain $\Omega_{\mathbf{v}}$: they are denoted $(N_i)_{i=1...N_v}$. From the reverse viewpoint, the node N_i is the image of (at least) one node of the reference element : $N_i = \tau_r(\hat{N}_k)$. As it is usual in the finite element implementations, we define a connectivity array that makes the relation between the local index k (in the reference element) and the global index i : $\mathrm{connec}(k, r) = i$ indicates that node N_i is the k-th local node of the element Q_r.

Transformation geometry. In the sequel, we will assume that the transformations τ_r are polynomial of degree lower than d. This requires that Ω_v can be also described (or approximated) by polynomial mappings. Under this assumption, we have the following expression

$$\tau_r(\hat{v}) = \sum_{k=1}^{(\mathrm{d}+1)^k} \hat{L}_k(\hat{v})\, N_{k,r}, \tag{2.3.5}$$

where $N_{k,r} = N_i$, with $i = \mathrm{connec}(K, r)$. Note that the cuboid domain $\Omega_{\mathbf{v}} =]-V_{\max}, V_{\max}[^k$ trivially satisfies the previous requirement and, in that case, τ_r is an affine transformation for any degree d larger than 1.

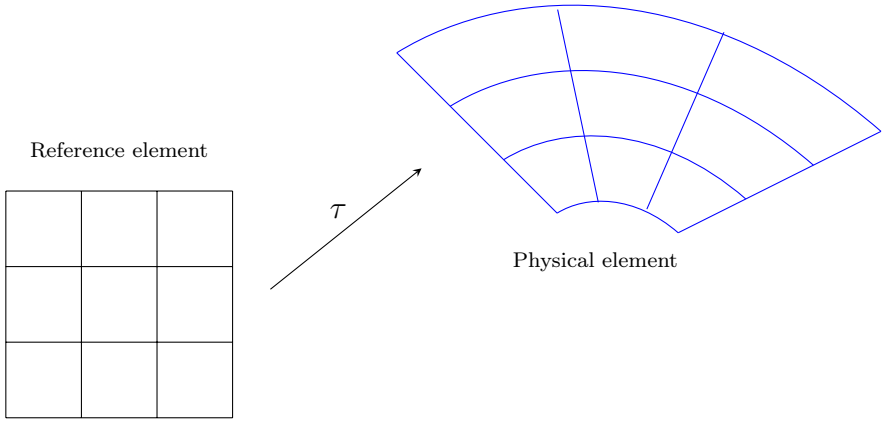


FIGURE 2.1 : Transformation τ maps a reference element onto a physical mesh element.

Remark 2.3.4 (Connectivity array)**.** *Each element Q_r, for $r = 1 \dots M^k$, is associated to the k-uplet $(m_1, \dots, m_k)$, where m_i denotes the index of the subdomain in the i-th direction. The one-to-one correspondance is given by the formula*

$$r = \sum_{i=1}^{k} M^{i-1}(m_i - 1) + 1, \tag{2.3.6}$$

and the connectivity array has the following expression

$$connec(K, r) = \sum_{i=1}^{k} M^{i-1}(I_i + (m_i - 1)\mathrm{d} - 1) + 1$$

where the (m_i) and the (I_i) are respectively defined from r and K by (2.3.6) *and* (2.3.2)*.*

2.3.3 Finite element basis

Definition 2.3.5 (Interpolation space)**.** *We consider the approximation space*

$$V_h = \left\{ w \in L^2(\Omega_{\mathbf{v}}) \cap C(\overline{\Omega}_{\mathbf{v}}), \quad \forall\ 1 \leqslant r \leqslant M^k,\ w_{|Q_r} \in \mathbb{Q}_{\mathrm{d}} \right\} \tag{2.3.7}$$

of dimension N_v.

Definition 2.3.6. ***Interpolation basis*** *The interpolation basis (φ_j) associated to V_h is constructed in such a way that each basis function φ_j is associated to a node N_j of the mesh and satisfies*

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

where δ_{ij} denotes the Kronecker symbol.

Let us describe how to evaluate the basis function φ_j at any $\mathbf{v} \in \Omega_{\mathbf{v}}$. Necessarily, $\mathbf{v}$ belongs at least to one finite element Q_r. Either node N_j does not belong to Q_r and then $\varphi_j(\mathbf{v}) = 0$, or node N_j belongs to finite element Q_r, i.e. $\exists K,\ N_j = N_{K,r}$, and then

$$\varphi_j(\mathbf{v}) = \hat{L}_K(\tau_r^{-1}(\mathbf{v})).$$

2.4 Vlasov-Poisson reduction model

2.4.1 Reduced Vlasov equation

Thanks to the finite element basis $(\varphi_j)_{j=1,\dots,N_{\mathbf{v}}}$ defined in definition 2.3.6, we are now able to consider an approximated distribution function of the form :

$$f(\mathbf{x},\mathbf{v},t) \approx \sum_{j=1}^{N_v} w_j(\mathbf{x},t)\,\varphi_j(\mathbf{v}). \tag{2.4.1}$$

We will perform a semi-discretization of equation (2.2.1) with respect to the velocity variable in order to obtain an hyperbolic system given the time evolution of the coefficient w_j.

Using the expression (2.4.1), the weak formulation

$$\partial_t \int_{\Omega_{\mathbf{v}}} f\varphi + \nabla_{\mathbf{x}} \cdot \int_{\Omega_{\mathbf{v}}} \mathbf{v} f\varphi + \mathbf{E} \cdot \int_{\Omega_{\mathbf{v}}} \nabla_{\mathbf{v}} f\varphi - \beta \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f\varphi = 0, \quad \forall \varphi \in V_h, \tag{2.4.2}$$

is then equivalent to

$$\sum_{j=1}^{N_v} \Big(\partial_t w_j \int_{\Omega_{\mathbf{v}}} \varphi_i\varphi_j + \nabla_{\mathbf{x}} w_j \cdot \int_{\Omega_{\mathbf{v}}} \mathbf{v}\varphi_i\varphi_j + w_j \int_{\Omega_{\mathbf{v}}} \varphi_i\,(\mathbf{E}\cdot\nabla_{\mathbf{v}})\varphi_j \\ -\beta\, w_j \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- \varphi_j\varphi_i \Big) \;=\; 0, \quad \forall\, 1 \leqslant i \leqslant N_v.$$

This system can be also written as follows

Proposition 2.4.1. ***Reduced Vlasov model***
The matrix formulation of these equations, referred as the reduced Vlasov equation, writes

$$\mathcal{M}\partial_t \mathbf{w} + A^k \partial_{x_k} \mathbf{w} + B(\mathbf{E})\mathbf{w} = 0, \quad k = 1\dots 3. \tag{2.4.3}$$

where $\mathbf{w}$ is the vector of N_v components $\mathbf{w} = (w_1, w_2, ..., w_{N_v})^T$. The mass matrix $\mathcal{M}$ (also called rigidity matrix) and the matrices $A^k, k = 1\dots 3$, $B(\mathbf{E})$ are matrices of dimension $N_v \times N_v$, whose elements are given by

$$\mathcal{M}_{ij} = \int_{\Omega_{\mathbf{v}}} \varphi_i\varphi_j, \quad A^k_{ij} = \int_{\Omega_{\mathbf{v}}} v_k\,\varphi_i\varphi_j, \quad k = 1\dots 3, \tag{2.4.4}$$

$$B(\mathbf{E})_{ij} = \int_{\Omega_{\mathbf{v}}} \varphi_i(\mathbf{E}\cdot\nabla_{\mathbf{v}})\varphi_j - \beta \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- \varphi_j\varphi_i. \tag{2.4.5}$$

A solution $\mathbf{w}$ of the reduced Vlasov model is called a discrete solution of the weak formulation of the Vlasov equation by FEM.

Remark 2.4.2. *Since the matrix A^k does not depend on $\mathbf{w}$, the reduced Vlasov equation 2.4.3 is a linear system. The source term comes from the transport in the velocity variable : it involves the electric field, obtained from the Poisson equation, and couples all the degree of freedom in velocity. The Poisson equation involves the density*

$$\rho(\mathbf{x},t) = \sum_{j=1}^{N_v} \alpha_j \, w_j(\mathbf{x},t), \quad \text{with } \alpha_j = \int_{\Omega_\mathbf{v}} \varphi_j.$$

Remark 2.4.3. *Due to the interpolation property of the basis $(\varphi_j)_{j=1,\dots,N_\mathbf{v}}$, the coefficient w_i corresponds to the ponctual value of the distribution function f at node N_i*

$$f(\mathbf{x},N_i,t) = \sum_{j=1}^{N_\mathbf{v}} w_j(\mathbf{x},t)\,\varphi_j(N_i) = w_i(\mathbf{x},t). \tag{2.4.6}$$

Therefore, given an initial distribution function $f_0(\mathbf{x},\mathbf{v})$, the initial condition for the reduced Vlasov equation is defined in the following way

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

2.4.2 Properties of the reduced Vlasov equation

Proposition 2.4.4. *The mass matrix $\mathcal{M}$, defined in* (2.4.4)*, is symmetric positive-definite.*

Démonstration. Indeed, we have

$$\mathcal{M}_{ij} = \int_{\Omega_\mathbf{v}} \varphi_i\varphi_j = \mathcal{M}_{ji}$$

so $\mathcal{M}$ is symmetric. In addition, let $X = (x_1, x_2, \dots, x_{N_v})$ an arbitrary vector in $\mathbb{R}^{N_v}$, then

$$X^T\mathcal{M}X = \sum_{k,j} x_k x_j \mathcal{M}_{kj} = \int_{\Omega_\mathbf{v}} \left(\sum_k x_k\varphi_k\right)^2.$$

Thanks to the interpolation property, $\sum_k x_k\varphi_k(v)$ is a vanishing function if and only if $X = 0$. We thus have $X^T\mathcal{M}X > 0,\ \forall X \in \mathbb{R}^{N_\mathbf{v}} \setminus \{0\}$ and the matrix $\mathcal{M}$ is symmetric positive-definite. □

Proposition 2.4.5 (Hyperbolicity)**.** *The reduced Vlasov system (2.4.3) is hyperbolic.*

Démonstration. The matrix $\mathcal{M}$ is symmetric positive-definite. Subsequently, it is invertible and the inverse of this matrix is symmetric. We can write the system (2.4.3) as follows

$$\partial_t\mathbf{w} + \mathcal{M}^{-1}A^k\partial_{x_k}\mathbf{w} + \mathcal{M}^{-1}B(\mathbf{E})\mathbf{w} = 0, \quad k = 1\dots3. \tag{2.4.7}$$

Furthermore, there exists a symmetric positive-definite R such that $\mathcal{M} = R^2$ and so $\mathcal{M}^{-1} = R^{-1}R^{-1}$. We have then

$$\begin{aligned} \forall\, k = 1 \dots 3, \qquad \mathcal{M}^{-1}A^k &= R^{-1}R^{-1}A^kR^{-1}R \\ &= R^{-1}S^kR, \end{aligned} \tag{2.4.8}$$

where $S^k = R^{-1}A^kR^{-1}$. We also have

$$\forall\, k = 1 \dots 3, \qquad (S^k)^T = \left(R^{-1}A^kR^{-1}\right)^T = R^{-1}A^kR^{-1} = S^k$$

so S^k is symmetric. It means that the matrices S^k are diagonalizable with real eigenvalues [85]. Noting that $\mathcal{M}^{-1}A^k$ and S^k have the same eigenvalues (from (2.4.8)), it implies that system (2.4.3) is hyperbolic (with a source term). □

Remark 2.4.6. *The hyperbolic property ensures that the linear system (2.4.3), without the source term, is well-posed (see [97]). Furthermore, classical numerical methods for hyperbolic systems can be used to solve the system.*

Proposition 2.4.7 (L^2 stability)**.** *For $1/2 \leqslant \beta \leqslant 1$, the reduced Vlasov equation is L^2 stable :*

$$\frac{1}{2}\frac{d}{dt}\left(\int_{\Omega_x\times\Omega_{\mathbf{v}}} f^2\right) \leqslant -\frac{1}{2}\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f^2, \tag{2.4.9}$$

where $f = \sum_{i=1}^{N_v} \mathbf{w}_i\varphi_i$.

Démonstration. Multiplying equation (2.4.3) by $\mathbf{w}$ and integrating in $\mathbf{x}$, we obtain

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\left(\int_{\Omega_{\mathbf{x}}} \mathbf{w}^T\mathcal{M}\mathbf{w}\right) &= -\int_{\Omega_{\mathbf{x}}} \frac{1}{2}\,\partial_{x_k}\left(\mathbf{w}^TA^k\mathbf{w}\right) - \int_{\Omega_{\mathbf{x}}} \mathbf{w}^TB(E)\mathbf{w} \\ &= -\int_{\Omega_{\mathbf{x}}} \mathbf{w}^TB(E)\mathbf{w}, \end{aligned}$$

since the matrices $\mathcal{M}$ and A^k are symmetric and the periodic boundary conditions in space. Using Green's formula, we can write matrix B as

$$\begin{aligned} B(\mathbf{E})_{ij} &= \mathbf{E}\cdot\int_{\Omega_{\mathbf{v}}} \varphi_i\nabla_{\mathbf{v}}\varphi_j - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i + \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i - \beta\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^-\varphi_j\varphi_i \\ &= R_{ij} + \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i - \beta\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^-\varphi_j\varphi_i, \end{aligned}$$

where $R = (R_{ij})$ is a matrix of dimension $N_{\mathbf{v}} \times N_{\mathbf{v}}$, which satisfies

$$\begin{aligned} R_{ij} &= \int_{\Omega_{\mathbf{v}}} \varphi_i\,(\mathbf{E}\cdot\nabla_{\mathbf{v}})\varphi_j - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i \\ &= -\int_{\Omega_{\mathbf{v}}} (\mathbf{E}\cdot\nabla_{\mathbf{v}})\varphi_i\,\varphi_j + \int_{\partial\Omega_{\mathbf{v}}} \varphi_i\varphi_j(\mathbf{E}\cdot n_{\mathbf{v}}) - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i \\ &= -\int_{\Omega_{\mathbf{v}}} (\mathbf{E}\cdot\nabla_{\mathbf{v}})\varphi_i\,\varphi_j - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\varphi_j\varphi_i \\ &= -R_{ji}, \end{aligned}$$

thanks to Green's formula. Therefore, R is an antisymmetric matrix, so, for any vector $\mathbf{w}$ of dimension $N_{\mathbf{v}}$, we have $\mathbf{w}^T R\mathbf{w} = 0$. Consequently

$$\frac{d}{dt}\left(\frac{1}{2}\int_{\Omega_{\mathbf{x}}} \mathbf{w}^T\mathcal{M}\mathbf{w}\right) = -\frac{1}{2}\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}}) w_j\varphi_j\varphi_i w_i + \beta\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- w_j\varphi_j\varphi_i w_i,$$

Noting that $w_j\varphi_j\varphi_i w_i = f^2$ and

$$\int_{\Omega_{\mathbf{x}}} \mathbf{w}^T\mathcal{M}\mathbf{w} = \int_{\Omega_{\mathbf{x}}} w_j(\int_{\Omega_{\mathbf{v}}} \varphi_i\,\varphi_j)\, w_i = \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f^2,$$

we obtain :

$$\begin{aligned}\frac{d}{dt}\left(\frac{1}{2}\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f^2\right) &= -\frac{1}{2}\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\, f^2 + \beta\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^-\, f^2,\\ &= -\frac{1}{2}\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} |\mathbf{E}\cdot n_{\mathbf{v}}|\, f^2 - (1-\beta)\int_{\Omega_x\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^-\, f^2\end{aligned}$$

using that $a = |a| + 2a^-$ for any $a\in\mathbb{R}$. For $1/2 \leqslant \beta \leqslant 1$, we have

$$\frac{d}{dt}\left(\frac{1}{2}\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f^2\right) \quad\leqslant\quad -\frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} |\mathbf{E}\cdot n_{\mathbf{v}}| f^2 - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f^2 = -\frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f^2.$$

Hence the result.

□

More generally, the boundary condition leads to the dissipation of all macroscopic quantities.

Proposition 2.4.8 (Conservation)**.** *We suppose $\beta = 1$. If the polynomials 1, v and v_iv_j, for $1\le i,j\le 3$ belong to the approximation space $V_h = span(\{\varphi_i\}_i)$ then the semi-discrete scheme satisfies :*

$$\frac{d}{dt}\rho_{tot} = -\int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f, \tag{2.4.10}$$

$$\frac{d}{dt}\mathcal{E}_{tot} = -\int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f\big(\frac{|\mathbf{v}|^2}{2} + \Phi\big), \tag{2.4.11}$$

$$\frac{d}{dt}\mathbf{J}_{tot} = -\int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f\,\mathbf{v}, \tag{2.4.12}$$

supposing periodic boundary conditions in space.

Suppose that f remains positive in time (numerically this is actually not the case), equation (2.4.10) would say that the total charge is decreasing due to the loss at the boundary (in velocity). For the total energy and current, we have a priori no dissipation property since the right-hand sides of (2.4.11) and (2.4.12) have no sign.

Démonstration. **For the total charge.** Taking $\varphi = 1$ in the weak formulation (2.4.2) we obtain

$$\int_{\Omega_{\mathbf{v}}} \partial_t f + \nabla_{\mathbf{x}}\cdot\Big(\int_{\Omega_{\mathbf{v}}} \mathbf{v} f\Big) + \int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f - \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f = 0. \tag{2.4.13}$$

Thanks to the divergence theorem (in velocity), we have

$$\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f = \int_{\Omega_{\mathbf{v}}} \nabla_{\mathbf{v}}\cdot(\mathbf{E}f) = \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})\, f. \tag{2.4.14}$$

Noting that $\mathbf{E}\cdot n_{\mathbf{v}} = (\mathbf{E}\cdot n_{\mathbf{v}})^+ + (\mathbf{E}\cdot n_{\mathbf{v}})^-$, we obtain the continuity equation :

$$\partial_t \rho + \nabla_{\mathbf{x}}\cdot\mathbf{J} + \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f = 0. \tag{2.4.15}$$

where ρ and $\mathbf{J}$ are the charge and current, defined in (1.2.14) and (1.2.13) . Then, integrating in space and using the periodic condition in space, we have

$$\int_{\Omega_{\mathbf{x}}} \nabla_{\mathbf{x}}\cdot\mathbf{J} = 0, \tag{2.4.16}$$

and then we get (2.4.10).

For the total energy. Now, we take the test function $\varphi = |\mathbf{v}|^2/2$, and integrate the weak formulation (2.4.2) with respect to $\mathbf{x}$. We obtain

$$\frac{d}{dt}\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\frac{|\mathbf{v}|^2}{2} + \int_{\Omega_{\mathbf{x}}} \nabla_{\mathbf{x}}\cdot\Big(\int_{\Omega_{\mathbf{v}}} \mathbf{v} f\,\frac{|\mathbf{v}|^2}{2}\Big) + \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\,\frac{|\mathbf{v}|^2}{2} - \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f\frac{|\mathbf{v}|^2}{2} = 0 \tag{2.4.17}$$

Thanks to the periodic condition in space, the second term vanishes. Using the Green's formula (in velocity), we have

$$\int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\,\frac{|\mathbf{v}|^2}{2} = \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} \frac{|\mathbf{v}|^2}{2}(\mathbf{E}\cdot n_{\mathbf{v}}) f - \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\mathbf{E}\cdot\nabla\frac{|\mathbf{v}|^2}{2}.$$

Using that $\mathbf{E}\cdot n_{\mathbf{v}} = (\mathbf{E}\cdot n_{\mathbf{v}})^+ + (\mathbf{E}\cdot n_{\mathbf{v}})^-$ and $\nabla|\mathbf{v}|^2 = 2\mathbf{v}$, we obtain

$$\frac{1}{2}\frac{d}{dt}\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f|\mathbf{v}|^2 + \frac{1}{2}\int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f|\mathbf{v}|^2 - \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} \mathbf{E}\cdot\mathbf{v} f = 0. \tag{2.4.18}$$

Using that $\mathbf{E} = -\nabla_x\Phi$ and Green's formula in space, we get

$$\begin{aligned} -\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} \mathbf{E}\cdot\mathbf{v} f &= \int_{\Omega_{\mathbf{x}}} \nabla_{\mathbf{x}}\Phi\cdot\mathbf{J} \\ &= -\int_{\partial\Omega_{\mathbf{x}}} \Phi\,(\mathbf{J}\cdot n_{\mathbf{x}}) - \int_{\Omega_{\mathbf{x}}} \Phi\,\nabla_{\mathbf{x}}\cdot\mathbf{J}. \end{aligned}$$

Thanks to the Dirichlet boundary condition for the potential, the first term vanishes. Using the continuity equation (2.4.15), we obtain

$$-\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} \mathbf{E}\cdot\mathbf{v} f = \int_{\Omega_{\mathbf{x}}} \Phi\,\partial_t\rho + \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}} \Phi\,(\mathbf{E}\cdot n_{\mathbf{v}})^+ f.$$

The first term equals the electric energy. Indeed, using the Poisson equation, Green's formula and the Dirichlet boundary condition, we have

$$\begin{aligned}\int_{\Omega_{\mathbf{x}}} \Phi\, \partial_t \rho &= -\int_{\Omega_{\mathbf{x}}} \Phi\, \partial_t(\Delta\Phi)\\ &= -\int_{\Omega_{\mathbf{x}}} \Phi\Delta(\partial_t\Phi)\\ &= -\int_{\partial\Omega_{\mathbf{x}}} \Phi\nabla\partial_t\Phi\cdot n_{\mathbf{x}} + \int_{\Omega_{\mathbf{x}}} \nabla\Phi\cdot\nabla\partial_t\Phi\\ &= \frac{1}{2}\frac{d}{dt}\int_{\Omega_{\mathbf{x}}} |\nabla\Phi|^2. \end{aligned} \tag{2.4.19}$$

Inserting (2.4.2)-(2.4.19) into (2.4.18), we obtain (2.4.11).

For the total current. The total momentum is the vector

$$\mathbf{J}_{\rm tot} = \Big(\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f v_k\Big)_{1\leqslant k\leqslant 3}.$$

For any $k\in\{1,2,3\}$, taking $\varphi = v_k$ in (2.4.2) by v_k and integrating with respect to $\mathbf{x}$, we obtain

$$\partial_t\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f v_k + \int_{\Omega_{\mathbf{x}}}\nabla_{\mathbf{x}}\cdot\Big(\int_{\Omega_{\mathbf{v}}}\mathbf{v} f v_k\Big) + \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}}\mathbf{E}\cdot\nabla_{\mathbf{v}} f v_k - \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}}(\mathbf{E}\cdot n_{\mathbf{v}})^- f v_k = 0 \tag{2.4.20}$$

Thanks to Green's formula, we have

$$\begin{aligned}\int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{v}}}\Big(\mathbf{E}\cdot\nabla_{\mathbf{v}} f\Big)\mathbf{v}_k &= \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}}(\mathbf{E}\cdot n_{\mathbf{v}}) f v_k - \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\,\mathbf{E}\cdot\nabla_{\mathbf{v}} v_k\\ &= \int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}}(\mathbf{E}\cdot n_{\mathbf{v}}) f v_k - \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\, E_k\end{aligned}$$

Using the relation $\mathbf{E}\cdot n_{\mathbf{v}} = (\mathbf{E}\cdot n_{\mathbf{v}})^+ + (\mathbf{E}\cdot n_{\mathbf{v}})^-$, we obtain

$$\frac{d}{dt}\mathbf{J}_{\rm tot} = -\int_{\Omega_{\mathbf{x}}\times\partial\Omega_{\mathbf{v}}}(\mathbf{E}\cdot n_{\mathbf{v}})^+ f\,\mathbf{v} + \int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\,\mathbf{E}.$$

The last term actually vanishes since, using the Poisson equation, we have

$$\begin{aligned}\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f\,\mathbf{E} &= \int_{\Omega_{\mathbf{x}}}\rho\,\mathbf{E}\\ &= \int_{\Omega_{\mathbf{x}}}\Big[(\nabla_{\mathbf{x}}\cdot\mathbf{E}) - \rho_0\Big]\mathbf{E}.\end{aligned}$$

Using spatial periodicity, we have :

$$\int_{\Omega_{\mathbf{x}}}\mathbf{E} = 0$$

and the identity also holds true

$$\begin{aligned}\int_{\Omega_{\mathbf{x}}}(\nabla_{\mathbf{x}}\cdot\mathbf{E})\mathbf{E} &= \int_{\Omega_{\mathbf{x}}}\Big[(\nabla_{\mathbf{x}}\cdot\mathbf{E})\mathbf{E} - \frac{1}{2}\nabla_{\mathbf{x}}\|\mathbf{E}\|^2\Big]\\ &= -\int_{\Omega_{\mathbf{x}}}(\nabla_{\mathbf{x}}\times\mathbf{E})\times\mathbf{E}\end{aligned}$$

which vanishes since $\mathbf{E} = -\nabla_{\mathbf{x}}\Phi$. We thus obtain (2.4.12).

□

Remark 2.4.9. *We can also get the dynamics of the full second order moment, namely it satisfies the following equation*

$$\frac{d}{dt}\left(\int_{\Omega_{\mathbf{x}}\Omega_{\mathbf{v}}} f\mathbf{v}\otimes\mathbf{v}\right) = -\frac{1}{2}\int_{\Omega_{\mathbf{x}}}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^{+}\mathbf{v}\otimes\mathbf{v} + \int_{\Omega_{\mathbf{x}}}\int_{\Omega_{\mathbf{v}}} f(\mathbf{E}\otimes\mathbf{v} + \mathbf{v}\otimes\mathbf{E}).$$

2.5 Convergence of the method (in velocity)

In this section, we study the convergence of the semi-discrete scheme in velocity by the FEM. We consider the space homogenous problem with a constant (non zero) electric field $\mathbf{E}$: it results into a linear transport equation (with constant velocity). The analysis reported below can be also found in [57]. We introduce the following bilinear form

Definition 2.5.1. *The bilinear form $\mathcal{B} : V \times V \mapsto \mathbb{R}$ is defined by*

$$\mathcal{B}(f,g) := \int_{\Omega_{\mathbf{v}}} \Big(\mathbf{E}\cdot\nabla_{\mathbf{v}} f\Big)\, g - \beta\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^{-} fg. \tag{2.5.1}$$

where $V = \{f \in L^2(\Omega_{\mathbf{v}}),\ \mathbf{E}\cdot\nabla_{\mathbf{v}} f \in L^2(\Omega_{\mathbf{v}})\}$.

Remark 2.5.2. *The space V equipped with the scalar product*

$$(u,v)_V := (u,v)_{L^2(\Omega_{\mathbf{v}})} + (\mathbf{E}\cdot\nabla_{\mathbf{v}} u + \mathbf{E}\cdot\nabla_{\mathbf{v}} v)_{L^2(\Omega_{\mathbf{v}})}, \qquad \textit{for all} \quad u, v \in V$$

is a Hilbert space. We also have $V \subset H^1(\Omega_{\mathbf{v}})$. The integration by parts hold on V and then

$$\|v\|_V := \|v\|_{L^2} + \|\mathbf{E}\cdot\nabla_{\mathbf{v}} v\|_{L^2}.$$

For more detail, see [34].

The bilinear form $\mathcal{B}$ is actually not antisymmetric. Indeed, using Green's formula, we have

$$\begin{aligned}
\frac{1}{2}\big(\mathcal{B}(f,g) - \mathcal{B}(g,f)\big) &= \frac{1}{2}\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\, g - \frac{1}{2}\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} g\, f \\
&= \int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\, g - \left(\frac{1}{2}\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\, g + \frac{1}{2}\int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} g\, f\right) \\
&= \int_{\Omega_{\mathbf{v}}} \mathbf{E}\cdot\nabla_{\mathbf{v}} f\, g - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}}) fg \\
&= \mathcal{B}(f,g) + \beta\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^{-} fg - \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}}) fg.
\end{aligned} \tag{2.5.2}$$

However, we have

Proposition 2.5.3. *With $1/2 \le \beta \le 1$, the bilinear form $\mathcal{B}$ is positive.*

Démonstration. Taking $f = g$ in equation (2.5.2), we obtain

$$\begin{aligned}\mathcal{B}(f,f) &= \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}}) f^2 - \beta \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f^2 \\ &= \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} |\mathbf{E}\cdot n_{\mathbf{v}}| f^2 + (1-\beta)\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f^2,\end{aligned}$$

using that $a = |a| + 2a^-$ for any $a \in \mathbb{R}$. Since $1/2 \leq \beta \leq 1$, this leads to

$$\mathcal{B}(f,f) \quad \geq \quad \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} |\mathbf{E}\cdot n_{\mathbf{v}}| f^2 + \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^- f^2 \quad = \quad \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E}\cdot n_{\mathbf{v}})^+ f^2,$$

which is positive. Note that it is the same proof as for the L^2 stability of the reduced Vlasov system (prop. 2.4.7) □

Remark 2.5.4. *In particular case where $\beta = 1$ then we have*

$$\mathcal{B}(f,f) = \frac{1}{2}\int_{\partial\Omega_{\mathbf{v}}} |\mathbf{E}\cdot n_{\mathbf{v}}| f^2 \tag{2.5.3}$$

The continuous problem then writes : Find $f \in C^1([0,T], V)$ such that

$$\int_{\Omega_{\mathbf{v}}} \partial_t f\, \varphi + \mathcal{B}(f,\varphi) = 0, \quad \forall \varphi \in V. \tag{2.5.4}$$

and the semi-discrete problem is : Find $f_h \in C^1([0,T], V_h)$ satisfies

$$\int_{\Omega_{\mathbf{v}}} \partial_t f_h\, \varphi + \mathcal{B}(f_h,\varphi) = 0, \quad \forall \varphi \in V_h, \tag{2.5.5}$$

where V_h is the approximation space defined in definition 2.3.5. Futhermore, since [84, 57], we have the following interpolation property :

Lemma 2.5.5 (Property of the interpolation basis)**.** *Suppose that we have a regular mesh. Let $f \in H^{\mathrm{d}+1}(\Omega_{\mathbf{v}})$, $\Pi(f)$ the L^2 orthogonal projection of f on the space $V_h \subset H^1$ and $\mathbf{D} = \Pi(f) - f$. There exists a constant γ such that*

$$\|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})} \leq \gamma\, h^{\mathrm{d}+1/2}\, \|f\|_{\mathrm{d}+1}, \tag{2.5.6}$$

$$\|\mathbf{D}\|_{L^2(\Omega_{\mathbf{v}})} \leq h^{\mathrm{d}+1}\, \|f\|_{\mathrm{d}+1}, \tag{2.5.7}$$

$$\|\nabla_{\mathbf{v}}\mathbf{D}\|_{L^2(\Omega_{\mathbf{v}})} \leq h^{\mathrm{d}}\, \|f\|_{\mathrm{d}+1}, \tag{2.5.8}$$

where $\|f\|_{d+1}$ denote the piecewise Sobolev norm (see [34]).

We then state the convergence.

Proposition 2.5.6 (Convergence)**.** *With $\beta = 1$, the solution f_h to the semi-discrete problem (2.5.5) converges to the solution f of the continuous problem (2.5.4) when h tends to zero. More precisely, we have*

$$\|f - f_h\|_{L^2(\Omega_{\mathbf{v}})} \leq C\, h^{\mathrm{d}}, \tag{2.5.9}$$

where C is a constant.

Let us provide the following lemma.

Lemma 2.5.7. *Let f and f_h the solutions to the continuous and discrete problems. Denoting $\mathbf{D} = \Pi(f) - f$ and $\mathbf{D}_h = \Pi(f) - f_h$, we have*

$$\mathcal{B}(\mathbf{D}, \mathbf{D}_h) \leq C_1 \, \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} \, \|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})} + C_2 \, \|\nabla_{\mathbf{v}} \mathbf{D}\|_{L^2(\Omega_{\mathbf{v}})} \, \|\mathbf{D}_h\|_{L^2(\Omega_{\mathbf{v}})}, \tag{2.5.10}$$

where C_1 and C_2 are constant.

Démonstration. Following the proof of the Cauchy-Schwarz inequality we have

$$\mathcal{B}(\mathbf{D} + t\,\mathbf{D}_h, \mathbf{D} + t\,\mathbf{D}_h) = t^2 \, \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) - t \left(\mathcal{B}(\mathbf{D}, \mathbf{D}_h) + \mathcal{B}(\mathbf{D}_h, \mathbf{D}) \right) + \mathcal{B}(\mathbf{D}, \mathbf{D}) \quad \geq 0,$$

for all $t \in \mathbb{R}$ if and only if the discriminant is negative

$$\left(\mathcal{B}(\mathbf{D}, \mathbf{D}_h) + \mathcal{B}(\mathbf{D}_h, \mathbf{D}) \right)^2 - 4 \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) \, \mathcal{B}(\mathbf{D}, \mathbf{D}) \leq 0.$$

We thus obtain

$$\mathcal{B}(\mathbf{D}, \mathbf{D}_h) + \mathcal{B}(\mathbf{D}_h, \mathbf{D}) \leq 2 \, \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} \, \mathcal{B}(\mathbf{D}, \mathbf{D})^{1/2}.$$

We then deduce the following inequality

$$\mathcal{B}(\mathbf{D}, \mathbf{D}_h) \quad \leq \quad \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} \mathcal{B}(\mathbf{D}, \mathbf{D})^{1/2} + \frac{1}{2} \left(\mathcal{B}(\mathbf{D}, \mathbf{D}_h) - \mathcal{B}(\mathbf{D}_h, \mathbf{D}) \right)$$

Taking $f = \mathbf{D}, g = \mathbf{D}_h$ and $\beta = 1$ into (2.5.2), the antisymmetric part becomes

$$\frac{1}{2} \left(\mathcal{B}(\mathbf{D}, \mathbf{D}_h) - \mathcal{B}(\mathbf{D}_h, \mathbf{D}) \right) = \int_{\Omega_{\mathbf{v}}} \mathbf{E} \cdot \nabla_{\mathbf{v}} \mathbf{D} \, \mathbf{D}_h - \frac{1}{2} \int_{\partial\Omega_{\mathbf{v}}} (E \cdot n_{\mathbf{v}}) \mathbf{D} \mathbf{D}_h.$$

We thus obtain

$$\mathcal{B}(\mathbf{D}, \mathbf{D}_h) \quad \leq \quad \mathcal{B}(\mathbf{D}, \mathbf{D})^{1/2} \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} + \int_{\Omega_{\mathbf{v}}} \mathbf{E} \cdot \nabla_{\mathbf{v}} \mathbf{D} \, \mathbf{D}_h - \frac{1}{2} \int_{\partial\Omega_{\mathbf{v}}} (E \cdot n_{\mathbf{v}}) \mathbf{D} \mathbf{D}_h. \tag{2.5.11}$$

We now bound each term of (2.5.11). For $\beta = 1$, thanks to the identity (2.5.3), we have

$$\mathcal{B}(\mathbf{D}, \mathbf{D})^{1/2} = \left| \frac{1}{2} \int_{\partial\Omega_{\mathbf{v}}} |\mathbf{E} \cdot n_{\mathbf{v}}| \, \mathbf{D}^2 \right|^{1/2} \leq \frac{E_{\max}}{\sqrt{2}} \|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})}. \tag{2.5.12}$$

Applying the Cauchy-Schwarz inequality, the third term writes

$$\left| \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E} \cdot n_{\mathbf{v}}) \mathbf{D} \mathbf{D}_h \right| \leq \left| \int_{\partial\Omega_{\mathbf{v}}} (\mathbf{E} \cdot n_{\mathbf{v}})^2 \mathbf{D}_h^2 \right|^{1/2} \|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})}$$

$$\leq C E_{\max} \, \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} \, \|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})}, \tag{2.5.13}$$

and by using the Cauchy-Schwarz inequality, the second term writes

$$\left| \int_{\Omega_{\mathbf{v}}} \mathbf{E} \cdot \nabla_{\mathbf{v}} \mathbf{D} \, \mathbf{D}_h \right| \leq E_{\max} \|\nabla_{\mathbf{v}} \mathbf{D}\|_{L^2(\Omega_{\mathbf{v}})} \left(\int_{\Omega_{\mathbf{v}}} \mathbf{D}_h^2 \right)^{1/2} \tag{2.5.14}$$

Collecting the bounds (2.5.12)-(2.5.13)-(2.5.14) into (2.5.7), we get

$$\mathcal{B}(\mathbf{D}, \mathbf{D}_h) \leq C_1 \, \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2} \, \|\mathbf{D}\|_{L^2(\partial\Omega_{\mathbf{v}})} + C_2 \, \|\nabla_{\mathbf{v}} \mathbf{D}\|_{L^2(\Omega_{\mathbf{v}})} \, \|\mathbf{D}_h\|_{L^2(\Omega_{\mathbf{v}})}, \tag{2.5.15}$$

where C_1 and C_2 are constants. □

Now, we use this lemma to prove the proposition of convergence.

Démonstration. We have $\mathbf{D}_h - \mathbf{D} = f - f_h$.

Substracting (2.5.4) and (2.5.5), we get for any $\varphi \in V_h$

$$\int_{\Omega_\mathbf{v}} \partial_t(f - f_h)\,\varphi + \mathcal{B}(f - f_h, \varphi) = 0. \tag{2.5.16}$$

Because (2.5.16) is true for all test functions $\varphi \in V_h$, taking $\varphi = \mathbf{D}_h \in V_h$ in (2.5.16) leads to

$$\int_{\Omega_\mathbf{v}} \partial_t(\mathbf{D}_h - \mathbf{D})\,\mathbf{D}_h + \mathcal{B}(\mathbf{D}_h - \mathbf{D}, \mathbf{D}_h) = 0.$$

Since $\mathbf{D} \in V_h^\perp$, $\partial_t \mathbf{D} \in V_h^\perp$ and then $\partial_t \mathbf{D}$ is orthogonal to $\mathbf{D}_h \in V_h$, we have

$$\frac{1}{2}\frac{d}{dt}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) = \mathcal{B}(\mathbf{D}, \mathbf{D}_h). \tag{2.5.17}$$

From the bound obtain in lemma 2.5.7, we deduce

$$\frac{1}{2}\frac{d}{dt}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + \mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) \leq C_1\,\mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2}\,\|\mathbf{D}\|_{L^2(\partial\Omega_\mathbf{v})} + C_2\,\|\nabla_\mathbf{v}\mathbf{D}\|_{L^2(\Omega_\mathbf{v})}\,\big(\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2\big)^{1/2}. \tag{2.5.18}$$

Applying Young's inequality (see remark 2.5.8 below, with $\alpha^2 = 1/C_1$, $a = \sqrt{C_1}\mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2}$, $b = \sqrt{C_1}h^{\mathrm{d}}\|f\|_{\mathrm{d}+1}$), we then obtain

$$C_1\,\mathcal{B}(\mathbf{D}_h, \mathbf{D}_h)^{1/2}\,\|\mathbf{D}\|_{L^2(\partial\Omega_\mathbf{v})} \leq \frac{1}{2}\mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) + C_1\|\mathbf{D}\|^2_{L^2(\partial\Omega_\mathbf{v})}. \tag{2.5.19}$$

Similarly, we have

$$C_2\,\|\nabla_\mathbf{v}\mathbf{D}\|_{L^2(\Omega_\mathbf{v})}\,\big(\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2\big)^{1/2} \leq \frac{1}{2}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + C_2\|\nabla_\mathbf{v}\mathbf{D}\|^2_{L^2(\Omega_\mathbf{v})}. \tag{2.5.20}$$

The inequation 2.5.18 becomes

$$\frac{1}{2}\frac{d}{dt}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + \frac{1}{2}\mathcal{B}(\mathbf{D}_h, \mathbf{D}_h) \leq \frac{1}{2}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + C_1\|\mathbf{D}\|^2_{L^2(\partial\Omega_\mathbf{v})} + C_2\|\nabla_\mathbf{v}\mathbf{D}\|^2_{L^2(\Omega_\mathbf{v})} \tag{2.5.21}$$

Since the positivity of $\mathcal{B}$ and since the lemma 2.5.5, we can write

$$\frac{1}{2}\frac{d}{dt}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 \leq \frac{1}{2}\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 + C\,h^{2\mathrm{d}}\,\|f\|^2_{\mathrm{d}+1}. \tag{2.5.22}$$

Applying Gronwall's inequality (see remark 2.5.9 below), we obtain

$$\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2 \leq \big(\int_{\Omega_\mathbf{v}} \mathbf{D}_h^2\big)_{|t=0}\,\exp(t) + Ch^{2\mathrm{d}}\int_0^T \exp(T-s)\|f\|^2_{\mathrm{d}+1}ds \tag{2.5.23}$$

or equivalent to

$$\|\mathbf{D}_h\|^2_{L^2(\Omega_\mathbf{v})} \leq Ch^{2\mathrm{d}}\int_0^T \|f\|^2_{\mathrm{d}+1}dt. \tag{2.5.24}$$

Finally, we have

$$\|f - f_h\|^2_{L^2(\Omega_\mathbf{v})} = \|\mathbf{D}_h - \mathbf{D}\|^2_{L^2(\Omega_\mathbf{v})} \leq \|\mathbf{D}_h\|^2_{L^2(\Omega_\mathbf{v})} + \|\mathbf{D}\|^2_{L^2(\Omega_\mathbf{v})},$$

and using (2.5.24) and the interpolation inequality (2.5.6), we get the result. □

Remark 2.5.8. *For any $a, b \in \mathbb{R}$ and $\alpha \in \mathbb{R}^*$, Young's inequality writes*

$$\frac{\alpha^2 a^2}{2} + \frac{b^2}{2\alpha^2} \geq 2\sqrt{\frac{\alpha^2 a^2}{2}\frac{b^2}{2\alpha^2}} = ab$$

Remark 2.5.9. *Suppose $f : [0,T] \to \mathbb{R}$ is differentiable. If we have*

$$\frac{d}{dt}f(t) \leq g(t)f(t) + h(t) \tag{2.5.25}$$

in which $g, h : [0,T] \to \mathbb{R}$ are continuous function. Then f satisfies the Gronwall's inequality

$$f(t) \leq f(0)\ \exp\big(\int_0^t g(s)ds\big) + \int_0^t \exp\big(\int_0^t g(s)ds - \int_0^r g(s)ds\big)h(r)dr, \tag{2.5.26}$$

for all $t \in [0,T]$. ([35, 100])

Remark 2.5.10. *Note that this convergence analysis is made when considering f_h as the solution to the discrete problem. It requires that the computation of the mass matrix $\mathcal{M}$ and the matrices $A^k, B(\mathbf{E})$ are exact. It is then important to choose the numerical integration formulas in order to have good approximations (see next section).*

2.6 Practical computation of the matrices in the reduced equation.

The matrices of the reduced Vlasov equation (2.4.1) have to be numerically computed. We present and compare Gauss Legendre and Gauss Lobatto method.

2.6.1 Gauss quadrature

The Legendre polynomials, defined on the interval $[-1,1]$, are orthogonal polynomials with respect to the scalar product

$$(f,g) := \int_{-1}^{1} f(x)g(x)\,dx, \qquad \text{for all polynomials } f, g.$$

The (normalized) Legendre polynomials are defined by

$$\forall n \in \mathbb{N}^*, \quad l_n(x) = \frac{\sqrt{n+\frac{1}{2}}}{n!2^n}\frac{d^n}{dx^n}((x^2-1)^n). \tag{2.6.1}$$

The n zeros of l_n and $n-2$ zeros of l'_{n-1} are distinct and in $]-1,1[$ [99, 5].

Proposition 2.6.1. *Let $g \in C([-1,1])$ and $n \in \mathbb{N}^*$.*

- *The Gauss-Legendre quadrature is given by :*

$$\int_{-1}^{1} g(\mathbf{v})d\mathbf{v} \simeq \sum_{i=1}^{n} \omega_i g(\xi_i), \tag{2.6.2}$$

where the quadrature points $(\xi_i)_{i=1\cdots n}$ are the zeros of l_n and the integration weights are given by

$$\omega_i = \frac{-\sqrt{2n+1}\sqrt{2n+3}}{(n+1)l_n'(\xi_i)l_{n+1}(\xi_i)}.$$

This formula is exact if g is a polynomial of degree at most $2n-1$.

- *The Gauss-Lobatto quadrature is given by :*

$$\int_{-1}^{1} g(\mathbf{v})d\mathbf{v} \simeq \frac{2}{n(n-1)}g(-1) + \sum_{j=2}^{n-1}\hat{\omega}_j g(\mu_j) + \frac{2}{n(n-1)}g(1). \tag{2.6.3}$$

where the quadrature points $(\mu_j)_{j=2\cdots n-1}$ are the zeros of l_{n-1}' and the integration weights are given by

$$\hat{\omega}_j = \frac{2}{n(n-1)(l_{n-1}(\mu_j))^2}, \quad j = 2\ldots n-1.$$

The formula is exact if g is a polynomial of degree at most $2n-3$.

Remark 2.6.2.

- *In order to compute the zeros and the weights of Gauss-Legendre and Gauss-Lobatto integrations, we make a program (Maple code) given in annex A.*

- *Given the number of quadrature points, the Gauss-Legendre integration has the advantage to be more precise. However, contrary to the Gauss-Lobatto rule, the extreme points of the interval are not quadrature points. The Gauss-Lobatto rule may be more appropriate since the quadrature points coincide with the Lagrange finite element nodes (see chapter 6).*

2.6.2 Matrix computation

The elements of matrices $\mathcal{M}$, A and B defined in (2.4.4) and (2.4.5) involve integrals on the domain $\Omega_{\mathbf{v}}$. We can write them as the sum of the integrals on each mesh element $Q_r, r = 1\ldots M^k$, that can be transform as integrals on the reference element $\hat{Q}$ thanks to the geometric transformation (2.3.5).
Indeed, the coefficients of these matrices write :

$$\mathcal{M}_{j_1,j_2} = \sum_{Q_i}\int_{\mathbf{v}\in Q_i}\varphi_{j_1}\varphi_{j_2},$$

$$A_{j_1,j_2} = \sum_{Q_i}\int_{\mathbf{v}\in Q_i}\mathbf{v}\,\varphi_{j_1}\varphi_{j_2},$$

$$B_{j_1,j_2} = \sum_{Q_i}\int_{\mathbf{v}\in Q_i}\varphi_{j_1}\,\mathbf{E}\cdot\nabla_v\varphi_{j_2} - \beta\sum_{Q_i}\int_{\partial\Omega_{\mathbf{v}}\cap\partial Q_i}(\mathbf{E}\cdot n_{\mathbf{v}})^{-}\varphi_{j_1}\varphi_{j_2}.$$

Then applying the geometric transformation, we obtain :

$$\mathcal{M}_{j_1,j_2} = \sum_{Q_i} \int_{\hat{v}\in\hat{Q}} \hat{L}_{k_1}\hat{L}_{k_2} |\det \nabla\tau_i|,$$
$$A_{j_1,j_2} = \sum_{Q_i} \int_{\hat{v}\in\hat{Q}} \tau_i\, \hat{L}_{k_1}\hat{L}_{k_2} |\det \nabla\tau_i|,$$
$$B_{j_1,j_2} = \sum_{Q_i} \int_{\hat{v}\in\hat{Q}} \hat{L}_{k_1}\, \mathbf{E}\cdot\nabla_{\hat{v}}\hat{L}_{k_2} |\det \nabla\tau_i| - \beta \sum_{Q_i} \int_{\partial\Omega_{\mathbf{v}}\cap\partial Q_i} (\mathbf{E}\cdot n_{\mathbf{v}})^{-}\varphi_{j_1}\varphi_{j_2},$$

where $\hat{L}_{k_1}$, $\hat{L}_{k_2}$ are the Lagrange polynomial associated to nodes N_{j_1} and N_{j_2} and $\tau_i : \hat{Q} \to Q_i$ is the geometric transformation of element Q_i and $\nabla\tau_i$ its Jacobian matrix. Due to the localized property of the finite elements, the above sums involves only element Q_i containing both N_{j_1} and N_{j_2} (similarly, see below the sparse property of the matrices). Finally, the above integrals are approximated using Gauss quadrature in each direction. Using the properties of Gauss quadrature, we have

Proposition 2.6.3. *Case* $\Omega_v = [-V_{\max}, V_{\max}]^k$.

1. *The computation of* $\mathcal{M}$, A *and* B *is exact when using Gauss-Legendre rule (with* $n = \mathrm{d}+1$ *integration points).*
2. *With Gauss-Lobatto rule (with* $n = \mathrm{d}+1$ *integration points), the computation of* B *is exact. When the integration points coincide with the finite element nodes (mass lumping), the matrices* $\mathcal{M}$ *and* A *are diagonal.*

Démonstration. When $\Omega_v = [-V_{\max}, V_{\max}]^k$, the transformations (τ_i) are affine. The integrand of elements of matrix $\mathcal{M}$ are polynomials of degree 2d, those of matrix A are of degree $2\mathrm{d}+1$ and those of matrix B of degree $2\mathrm{d}-1$. Hence the result. □

Remark 2.6.4. *Concerning the assembly algorithm, it is detailed in annex B. Note that we can compute and store the values and derivatives of the reference basis functions at the reference Gauss points at the beginning of the computation.*

Sparsity and storage : We first observe that matrices $\mathcal{M}$, A and B defined in (2.4.4) and (2.4.5) are sparse. More precisely, let j_1, j_2 such that N_{j_1}, N_{j_2} do not belong to the same element, we have

$$\varphi_{j_1}(\mathbf{v})\ \varphi_{j_2}(\mathbf{v}) = 0, \quad \varphi_{j_1}(\mathbf{v})\ \nabla\varphi_{j_2}(\mathbf{v}) = 0,$$

for all $\mathbf{v} \in \Omega_{\mathbf{v}}$. It means that the elements of indices (j_1, j_2) of the matrices equal zero. The matrices are band matrices. In dimension one, the bandwith equals the degree d of the local polynomial interpolation. In practice, we will use the skyline storage. Skyline storage allows to saving memory and time when computing the LU decomposition. The skyline structure is preserved during Gaussian elimination.

In particular, in order to store a sparse matrix $\mathcal{M}$ (or A or B) of size $N_v \times N_v$ in the skyline format, we use the following arrays : mdiag$(1 : N_v)$ for storing the diagonal, msup$(1 :$ nsky$)$ and mlow$(1 :$ nsky$)$ for storing the upper and lower parts. The integer nsky is unknown at the beginning. We need also an additional array mkld$(1 : N_v + 1)$ for locating the beginning

of each column (resp. row) in msup (reps. mlow). We use the convention mkld(1) = 1 and mkld(N_v + 1) = nsky + 1. In practice, for constructing mkld, we use an intermediate array prof(1 : N_v) where prof(i) contains the number of stored elements of $\mathcal{M}$ in column (row) i in msup (mlow). Thus prof(1) = 0. For building prof, we use the following algorithm

```
prof = 0
do k = 1, nel
    do ii = 1, d+1
        do jj = 1, d+1
            i = connec(ii, k)
            j = connec(jj, k)
            prof(j) = max(prof(j), j - i)
            prof(i) = max(prof(i), i - j)
        enddo
    enddo
enddo
```

Once prof is known, mkld is built with the following algorithm

```
mkld(1) = 1
do i = 1, N_v
    mkld(i+1) = mkld(i) + prof(i)
enddo
```

At the end of this algorithm, we know nsky = mkld(N_v + 1) - 1, we can thus allocate the memory for msup and mlow.

Deuxième partie

Simulation numérique et validation en dimension 1

Chapitre 3

Finite volume scheme for the 1D Vlasov reduced equation

The first two chapters of this thesis constitute an introduction to the general physical model and how to use a FEM to construct the reduced Vlasov equation. We are now interested in finding numerical methods to solve the reduced Vlasov-Poisson or Vlasov-Maxwell equations. We also wish to study the advantages and drawbacks of each method. It is natural to try to solve the problem in one dimension first, and it the goal of this chapter. In one dimension, the Vlasov-Poisson system is equivalent with the Vlasov-Ampère system. In order to compute the electric field, we can numerically solve the Ampère equation with a finite difference method or the Poisson equation with the FFT method. For solving the reduced hyperbolic Vlasov equation, we consider the finite volume scheme (see [69]) for the space discretization and the Runge-Kutta (RK) scheme for the time discretization. We can consider the centered numerical flux, which is second order accurate but is less stable than the viscous flux, which is first order accurate and more dissipative. In this chapter, we analyze the stability domain of the schemes with the two numerical fluxes combined to the RK schemes of order $1, 2, 3$ and 4. At the end of this chapter, we present numerical results for classical plasma physics test cases : the transport test case, the Landau damping test case and the two-stream instability test case.

Notice that, in the one-dimensional case (chapter 3 and 4), we simplify the notations : $\mathbf{x}$ becomes x, $\mathbf{v}$ becomes v and $\mathbf{E}$ becomes E.

3.1 Vlasov-Poisson and Vlasov-Ampère model in one dimension and the initial condition

3.1.1 Vlasov-Poisson 1D model

In dimension one, the Vlasov-Poisson system writes

$$\partial_t f + v\,\partial_x f + E\,\partial_v f = 0, \tag{3.1.1}$$

$$\partial_x E = -\rho_0 + \rho. \tag{3.1.2}$$

As mentioned in chapter 2, we consider the phase space domain $\Omega_x \times \Omega_v =]0, L[\times\mathbb{R}$. Periodic boundary conditions in space are applied at $x = 0$ and $x = L$, which means

$$f(0, v, t) = f(L, v, t),$$
$$E(0, t) = E(L, t).$$

Consequently, integrating the Poisson equation (3.1.2), the quantity ρ_0 has to satisfy :

$$\rho_0 = \frac{1}{L}\int_{\Omega_x} \rho(t, x)\, dx = \frac{1}{L}\,\rho_{\text{tot}}(t).$$

It means that the total charge in the domain vanishes (see proposition 1.4.4) . The density ρ_0 can be interpreted as the charge density of the background ions. Because of (3.1.2) the electric field E is still defined up to a constant. Supposing that the mean electric current also vanishes, the electric field must have a zero mean value

$$\int_{\Omega_x} E = 0. \tag{3.1.3}$$

Note that this is equivalent to impose periodic Dirichlet boundary conditions for the electric potential (since $E = -\partial_x\phi$). But because the potential is defined up to a constant it also equivalent to impose homogeneous Dirichlet boundary conditions for the electric potential. System (3.1.1)-(3.1.2) is supplemented by initial conditions

$$E(x, 0) = E_0(x), \quad f(x, v, 0) = f_0(x, v).$$

We also call system (3.1.1)-(3.1.2)-(3.1.3) the 1D Vlasov-Poisson model.

Remark 3.1.1. *The simulation of the Vlasov-Poisson system requires the resolution of an elliptic equation for the electric potential. In the one-dimensional framework, the equation reduces to (3.1.2), which is numerically solved by an FFT-based algorithm. It is interesting to propose an equivalent equation for the electric field, which implies the simple numerical resolution of a local differential equation.*

3.1.2 Vlasov-Ampère 1D model

Assuming that the distribution function and the electric field are regular enough, we formally have

Proposition 3.1.2. *The 1D Vlasov-Poisson system (3.1.1)-(3.1.2) is equivalent to the 1D Vlasov-Ampère system*

$$\partial_t f + v\,\partial_x f + E\,\partial_v f = 0, \tag{3.1.4}$$
$$\partial_t E = \bar{J} - J, \tag{3.1.5}$$

under the assumption that the Poisson equation is satisfied initially

$$\big(\partial_x E - (\rho_0 - \rho)\big)_{t=0} = 0.$$

$\bar{J}$ is the average current : $\bar{J} = J_{tot}/L$, where J_{tot} is defined in (1.4.2).

Démonstration. Taking the derivative of (3.1.2) with respect to t, we obtain

$$\partial_t \left(\partial_x E\right) = \partial_t(-\rho_0 + \rho). \tag{3.1.6}$$

Using the continuity equation (1.4.1), we have

$$\partial_t \partial_x E = -\partial_x J.$$

Permuting spatial and time derivatives, we obtain

$$\partial_x \partial_t E = -\partial_x J.$$

Therefore, there exists a function of time, $C(t) \in \mathbb{R}$, such that

$$\partial_t E = -J + C. \tag{3.1.7}$$

Then, integrating with respect to x the two sides of equation (3.1.7), we obtain

$$\int_{\Omega_x} (\partial_t E) = \int_{\Omega_x} (-J + C).$$

Since the electric field has zero mean value (3.1.3), we have

$$\int_{\Omega_x} (\partial_t E) = \partial_t \left(\int_{\Omega_x} E \right) = 0.$$

Finally, we deduce that

$$C(t) = \frac{1}{L} \int_{\Omega_x} J(t,x) = \frac{1}{L} J_{\text{tot}}.$$

Therefore, we obtain equation (3.1.5).

Reciprocally, deriving the Ampère equation with respect to space and using the continuity equation we get (3.1.6). Integrating in time and assuming that the Poisson equation is satisfied initially, we obtain the Poisson equation for $t > 0$. □

Remark 3.1.3. *In higher space dimensions, we can not obtain a corresponding Ampère equation (equation* (3.1.7) *is not valid).*

For more detail about Ampère equation, see [33, 24].

3.1.3 Velocity boundary condition

As discussed in chapter 2, because of the numerical approximation, we consider a bounded velocity domain $\Omega_v =]-V_{\max}, V_{\max}[$, with $V_{\max} \in \mathbb{R}$ and the boundary conditions (2.2.3) in velocity

$$(E \cdot n_v)^- f = 0. \tag{3.1.8}$$

As already mentioned in proposition 2.4.8, the total charge density is no more a conserved quantity due to the loss at the velocity boundaries. So as to maintain the well-posedness of the Poisson equation (3.1.2), ρ_0 should still remain the mean charge

$$\rho_0(t) = \frac{1}{L} \int_{\Omega_x \times \Omega_v} f(x,v,t)\, dxdv,$$

where $\Omega_x \times \Omega_v$ is the bounded computational domain. Consequently, ρ_0 is now time varying. For the Ampère equation, the average momentum $\bar{J}$ should also be computed on the computational domain

$$\bar{J}(t) = \frac{1}{L} \int_{\Omega_x \times \Omega_v} v\, f(x,v,t)\, dxdv.$$

3.1.4 1D reduced Vlasov equation

We recall that, in chapter 2, we have semi-discretized in velocity the Vlasov equation in order to obtain a new equation in which the unknowns depend only on space and time. In dimension one, the reduced Vlasov equation given in proposition 2.4.1 writes

$$\mathcal{M}\partial_t \mathbf{w} + A\partial_x \mathbf{w} + B(E)\mathbf{w} = 0, \tag{3.1.9}$$

in which the unknown $\mathbf{w}$ is a vector of N_v components

$$\mathbf{w} = (w_1, w_2, ..., w_{N_v})^T.$$

The involved matrices are of dimension $N_v \times N_v$ with elements given by

$$\mathcal{M}_{ij} = \int_v \varphi_i \varphi_j, \quad A_{ij} = \int_v v\varphi_i \varphi_j.$$

$$B_{ij}(E) = \beta\, E^+ \varphi_j(-V)\varphi_i(-V) - \beta\, E^- \varphi_j(V)\varphi_i(V) + E\int_v \varphi_i \varphi_j',$$

where the $(\varphi_i)_i$ is the finite-element basis in velocity (see definition 2.3.6).

Remark 3.1.4. *In this chapter, we choose $\beta = 1/2$.*

As noticed in chapter 2, A and $\mathcal{M}$ are symmetric matrices and $\mathcal{M}$ is positive definite and the system (3.1.9) is hyperbolic. On the other hand, thanks to an integration by part, we find that

$$B_{ij} = -B_{ij}, \qquad \text{if } (i,j) \neq (1,1) \text{ and } (i,j) \neq (N_v, N_v).$$

In other words the matrix B is "almost" skew-symmetric. It is not skew-symmetric because of the small charge leakage at the boundary. In addition

$$\text{if } E > 0, \quad B_{11}(E) = 0, \quad B_{N_v N_v}(E) = \frac{1}{2} \tag{3.1.10}$$

and

$$\text{if } E < 0, \quad B_{11}(E) = -\frac{1}{2}, \quad B_{N_v N_v}(E) = 0. \tag{3.1.11}$$

Remark 3.1.5. *For the semi-discretization in velocity, we make the following choice*

- *we use the FEM basis with equally space nodes in $[-V_{\max}, V_{\max}]$,*
- *the matrices are computed using the Gauss-Legendre integration in order to achieve exact numerical integration (see section 2.6)*

3.2 Semi-discretization in space

We are now interested in numerical methods for solving the reduced Vlasov equation. In this section, we present the finite volume methodology for linear equation.

3.2.1 Finite volume schemes

We explain briefly how to apply finite volume scheme in our case. For more details on this method, we refer to [21, 16, 69] or [56].

We break the domain $]0, L[$ into N_x grid cells. The cell C_i is the interval $\Big]x_{i-1/2}, x_{i+1/2}\Big[$, $i = 1 \dots N_x$. The center of the cell C_i is $x_i = i\triangle x - \frac{\triangle x}{2}$. The space step is $\triangle x = \frac{L}{N_x}$.

Denoting the source term $S(\mathbf{w}) = -B(E)\mathbf{w}$, the reduced Vlasov equation (3.1.9) writes

$$\mathcal{M}\partial_t \mathbf{w} = -A\partial_x \mathbf{w} + S(\mathbf{w}). \tag{3.2.1}$$

We denote by $(\mathbf{w}_k)_k \in (\mathbb{R}^{N_v})^{N_x}$ a piecewise constant approximation (in space) of $\mathbf{w}$

$$\forall\, 1 \le k \le N_x, \forall x \in C_k, \quad \mathbf{w}_k(t) \simeq \mathbf{w}(x,t), . \tag{3.2.2}$$

A finite-volume approximation of (3.2.1) writes

$$\partial_t \mathbf{w}_i = -\frac{\mathcal{F}(\mathbf{w}_i, \mathbf{w}_{i+1}) - \mathcal{F}(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\triangle x} + S(\mathbf{w})_i. \tag{3.2.3}$$

where $(\mathbf{w}_L, \mathbf{w}_R) \in \mathbb{R}^{N_v} \times \mathbb{R}^{N_v} \mapsto \mathcal{F}(\mathbf{w}_L, \mathbf{w}_R) \in \mathbb{R}^{N_v}$ denotes the numerical flux and $S(\mathbf{w})_i = -B(E)\mathbf{w}_i$.

Remark 3.2.1.

- *For practical reasons, we also consider two virtual cells C_0 and C_{N_x+1} for applying the periodic boundary condition. At the beginning of a time step, we copy the values of the cell C_{N_x} to the cell C_0, and the values of the cell C_1 to the cell C_{N_x+1}.*

- *For the initial condition, we consider $E_0 = E(\cdot, t = 0)$, and $f_0 = f(\cdot, \cdot, t = 0)$. Because of the interpolation property $\varphi_i(N_j) = \delta_{ij}$ for all $1 \le i, j \le N_v$ we have then*

$$f(x, N_j, t) = \sum_{i=1}^{P} w_i(x,t)\varphi_i(N_j) = w_j(x,t). \tag{3.2.4}$$

3.2.2 Numerical flux

For numerical flux, we consider the centered or upwind numerical flux.

Centered flux

The centered flux is given by

$$\mathcal{F}(\mathbf{w}_L, \mathbf{w}_R) = \mathcal{M}^{-1} A \frac{\mathbf{w}_L + \mathbf{w}_R}{2}$$

We call it the centered flux since it involves the average of the right and left states.

Upwind flux

According to proposition 2.4.5, the matrix $\mathcal{M}^{-1}A$ is diagonalisable with real eigenvalues. It means that there exists an invertible matrix P and a diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{N_v})$ such that $\mathcal{M}^{-1}A = P\Lambda P^{-1}$. Denote

$$\Lambda^+ = \begin{pmatrix} \lambda_1^+ & & & \\ & \lambda_2^+ & & \\ & & \ddots & \\ & & & \lambda_{N_v}^+ \end{pmatrix}, \qquad \Lambda^- = \begin{pmatrix} \lambda_1^- & & & \\ & \lambda_2^- & & \\ & & \ddots & \\ & & & \lambda_{N_v}^- \end{pmatrix}$$

where $a^+ = \max(a, 0)$ (resp. $a^- = \min(a, 0)$) denotes the positive part and negative part of any $a \in \mathbb{R}$, The upwind flux is then given by

$$\mathcal{F}(\mathbf{w}_L, \mathbf{w}_R) = (\mathcal{M}^{-1}A)^+ \mathbf{w}_L + (\mathcal{M}^{-1}A)^- \mathbf{w}_R,$$

where $(\mathcal{M}^{-1}A)^+ = P\Lambda^+P^{-1}$ and $(\mathcal{M}^{-1}A)^- = P\Lambda^-P^{-1}$. We have $(\mathcal{M}^{-1}A)^+ + (\mathcal{M}^{-1}A)^- = \mathcal{M}^{-1}A$: $(\mathcal{M}^{-1}A)^+$ and $(\mathcal{M}^{-1}A)^-$ respectively represent the right going and left going propagation. It is then equivalent with the following formula

$$\mathcal{F}(\mathbf{w}_L, \mathbf{w}_R) = (\mathcal{M}^{-1}A)\frac{\mathbf{w}_L + \mathbf{w}_R}{2} - |\mathcal{M}^{-1}A|\frac{(\mathbf{w}_R - \mathbf{w}_L)}{2} \tag{3.2.5}$$

with $|\mathcal{M}^{-1}A| = (\mathcal{M}^{-1}A)^+ - (\mathcal{M}^{-1}A)^-$. We will use this numerical flux in chapters 7 and 8.

Viscous flux

More generally, we can define a viscous flux by

$$\mathcal{F}(\mathbf{w}_L, \mathbf{w}_R) = (\mathcal{M}^{-1}A)\frac{\mathbf{w}_L + \mathbf{w}_R}{2} - \kappa\,\frac{(\mathbf{w}_R - \mathbf{w}_L)}{2}, \qquad \text{with} \quad \kappa \geq 0. \tag{3.2.6}$$

where κ is the numerical diffusion. If $\kappa = 0$, we recover the centered flux. Taking $\kappa = \max_{i=1\ldots N_v}\{|\lambda_i|\}$, we obtain the so-called Rusanov flux (see [90, 16]). This numerical flux will be the one chosen for the Vlasov-Poisson simulations in 1D (in the present chapter and chapter 4) and in 2D (in chapter 5).

3.3 Time discretization

We consider a sequence of times t_n, $n \in \mathbb{N}$, such that $t_0 = 0$ and $t_n = n\triangle t$, with $\triangle t > 0$. We denote $\mathbf{w}_i^n$ the approximation of $\mathbf{w}$ at time t^n in the cell C_i

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

We denote $\mathfrak{F}(\mathbf{w})$ the right hand side of equation (3.2.3)

$$\mathfrak{F}(\mathbf{w})_i = -\frac{\mathcal{F}(\mathbf{w}_i, \mathbf{w}_{i+1}) - \mathcal{F}(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\triangle x} + S(\mathbf{w}_i)$$

We consider the following Runge-Kutta schemes

First-order Euler method

$$\mathbf{w}_i^{n+1} = \mathbf{w}_i^n + \Delta t\, \mathfrak{F}(\mathbf{w}^n)_i. \tag{3.3.1}$$

Runge-Kutta 2

$$\begin{aligned} \widetilde{\mathbf{w}}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{2}\, \mathfrak{F}(\mathbf{w}^n)_i, \\ \mathbf{w}_i^{n+1} &= \mathbf{w}_i^n + \Delta t\, \mathfrak{F}(\widetilde{\mathbf{w}}^{n+1})_i. \end{aligned} \tag{3.3.2}$$

Rung-Kutta 3

$$\begin{aligned} \widetilde{\mathbf{w}}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{2}\, \mathfrak{F}(\mathbf{w}^n)_i, \\ \widetilde{\widetilde{\mathbf{w}}}_i^{n+1} &= \mathbf{w}_i^n - \Delta t\, \mathfrak{F}(\mathbf{w}^n)_i + 2\Delta t\, \mathfrak{F}(\widetilde{\mathbf{w}}^{n+1})_i, \\ \mathbf{w}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{6}\, \mathfrak{F}(\mathbf{w}_i^n) + \frac{2\Delta t}{3}\, \mathfrak{F}(\widetilde{\mathbf{w}}^{n+1})_i + \frac{\Delta t}{6}\, \mathfrak{F}(\widetilde{\widetilde{\mathbf{w}}}^{n+1})_i. \end{aligned} \tag{3.3.3}$$

Rung-Kutta 4

$$\begin{aligned} \widetilde{\mathbf{w}}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{2}\, \mathfrak{F}(\mathbf{w}^n)_i, \\ \widetilde{\widetilde{\mathbf{w}}}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{2}\, \mathfrak{F}(\widetilde{\mathbf{w}}^{n+1})_i, \\ \widetilde{\widetilde{\widetilde{\mathbf{w}}}}_i^{n+1} &= \mathbf{w}_i^n + \Delta t\, \mathfrak{F}(\widetilde{\widetilde{\mathbf{w}}}^{n+1})_i, \\ \mathbf{w}_i^{n+1} &= \mathbf{w}_i^n + \frac{\Delta t}{6}\, \mathfrak{F}(\mathbf{w}^n)_i + \frac{\Delta t}{3}\, \mathfrak{F}(\tilde{\mathbf{w}}^{n+1})_i + \frac{\Delta t}{3}\, \mathfrak{F}(\widetilde{\widetilde{\mathbf{w}}}^{n+1})_i + \frac{\Delta t}{6}\, \mathfrak{F}(\widetilde{\widetilde{\widetilde{\mathbf{w}}}}^{n+1})_i. \end{aligned} \tag{3.3.4}$$

where $\widetilde{\mathbf{w}}, \widetilde{\widetilde{\mathbf{w}}}, \widetilde{\widetilde{\widetilde{\mathbf{w}}}}$ are intermediate states. Computing $\mathfrak{F}$ requires to compute the electric field E by solving the Poisson or Ampère equation (see section 3.4). For instance, if we use the RK4 scheme, we have to solve the Ampère or Poisson equation 4 times per time step. Note that, without the source term, $\mathfrak{F}$ is linear and the Runge-Kutta schemes are equivalent to Taylor schemes.

For explicit schemes, stability requires that the time step is constrained by a CFL condition. We denote by $\wp$ the CFL number. In the following, the time step $\triangle t$ equals

$$\triangle t = \wp\, \frac{\triangle x}{V_{\max}}, \quad 0 < \wp \leq 1. \tag{3.3.5}$$

Stability will be discussed in section 3.5.

Remark 3.3.1. ***Initial condition of Vlasov equation***
We have then that the j^{th} component of the initial condition is given by

$$w_j(x, 0) = f_0(x, N_j) \tag{3.3.6}$$

for each $j = 1 \ldots N_v$.

Remark 3.3.2. ***Implementation aspects***
In practice, for saving CPU time and memory, we use two subroutines for computations with matrices stored in the sparse skyline format (see section (2.6.2)). The first one computes the product of a sparse matrix with a vector. The other one is used to solve linear systems (by the LU method)

3.4 Numerical schemes for the source term

We have explained how we evolve the reduced Vlasov equation. We have also to evolve the electric field E in order to compute the source term. As seen in section 3.1, we known that the electric field can be deduced from the Poisson equation or Ampère equation. The simplest method consists in solving the Ampère equation (3.1.5). The other method is based on a resolution of the Poisson equation (3.1.2).

3.4.1 Ampère equation

We consider the Ampère equation (3.1.5). From expression (2.4.1) of the distribution function, we have

$$\begin{aligned}\partial_t E(x,t) &= -\int_{\Omega_v} f(x,v,t)\, v \\ &= -\int_{\Omega_v} \sum_{j=1}^{N_v} w_j(x,t)\varphi_j(v) v = -\sum_{j=1}^{N_v} w_j(x,t)\,\zeta_j,\end{aligned}$$

where

$$\zeta_j = \int_{\Omega_v} v\,\varphi_j(v).$$

We can compute the vector $(\zeta_j)_{j=1\cdots N_v}$ with the Gauss numerical integration as described in Section 2.6. For the time integration, we use either the first-order Euler scheme or the second-order improved Euler scheme or RK3, RK4.

Time discretization. In order to obtain precise results, we consider the following semi-explicit coupling between the Vlasov and Ampère resolution

1. Compute $\mathbf{w}^{n+1}$ from E^n with the scheme presented in section 3.3,
2. Compute E^{n+1} from $\mathbf{w}^{n+1}$ with the Euler scheme

$$\frac{E(x,t^{n+1}) - E(x,t^n)}{\Delta t} = -\sum_{j=1}^{N_v} w_j(x,t^{n+1})\,\zeta_j,$$

or a second order scheme.

3.4.2 Poisson equation

Another way to compute the electric field is by solving the Poisson equation (3.1.2). We remark that this approach is followed by nearly all the other works or articles. In our work, taking advantage of the uniform mesh, we use the FFT (Fast Fourier Transform) algorithm.

The Poisson equation (3.1.2) reads

$$\begin{aligned}\partial_x E(x,t^n) &= -\rho_0 + \int_{\Omega_v} f(x,v,t^n) \\ &= -\rho_0 + \sum_{j=1}^{N_v} w_j(x,t^n)\varrho_j, \qquad (3.4.1)\end{aligned}$$

where

$$\varrho_j = \int_{\Omega_v} \varphi_j(v).$$

The vector $(\varrho_j)_j$ is computed using the numerical integration methods described in section 2.6. To simplify the notation, let us write the Poisson equation

$$\partial_x E = \sigma(x),$$

where σ stands for the right-hand side of equation (3.4.1). We consider the cell-centered electric field approximation $(E_0, E_1, ..., E_{N_x-1})$ in the grid cells (C_i). The centered finite-difference approximation of the Poisson equation reads

$$\frac{E_{i+1} - E_{i-1}}{2\triangle x} = \sigma(x_i), \tag{3.4.2}$$

where index arithmetic is considered modulo N_x (in other words we assume that $i \in \mathbb{Z}/N_x\mathbb{Z}$). For any vector $(v_i) \in \mathbb{R}^{N_x}$, we denote by $(\widehat{v}_k) \in \mathbb{R}^{N_x}$ its Discrete Fourier Transform (DFT) . It is given by

$$\widehat{v}_k = \sum_{j=0}^{N_x-1} v_j \, e^{\frac{-2I\pi jk}{N_x}}, \quad k = 0 \cdots N_x - 1,$$

where $I = \sqrt{-1}$. The inverse DFT is given by

$$v_j = \frac{1}{N_x} \sum_{k=0}^{N_x-1} \widehat{v}_k \, e^{\frac{2I\pi jk}{N_x}}, \quad \forall k = 0 \cdots N_x - 1.$$

Taking the DFT of the two sides of (3.4.2), we obtain

$$\widehat{E}_k = \frac{\triangle x}{I \sin\left(\frac{2k\pi}{N_x}\right)} \widehat{\sigma}_k, \quad \forall k \neq 0, \tag{3.4.3}$$

For the case $k = 0$, using the zero average condition (3.1.3), we impose

$$\widehat{E}_0 = \sum_{j=0}^{N-1} E_j = 0.$$

and we recover E by taking the inverse DFT.

Implementation

We use the Fast Fourier Transform (FFT) algorithm (see [82]) for computing the direct and inverse DFT.

3.5 Stability of numerical scheme for the reduced Vlasov equation

It is of interest to know whether the numerical schemes we have chosen are stable or not. In this section, we first recall some basic concepts and notations related to the stability property. In order to study the stability of the reduced Vlasov equation (3.1.9) we split the reduced Vlasov equation into two parts : the transport part and the source term part. In the two cases, studying the stability is almost the same. We thus only consider the case of the transport equation, the case of the source term part is similar.

Note that the idea of studying the stability in this way is given for instance in the work of Levy, Doron and Tadmor, Eitan in [70]. Their work is for a more general case with the energy method. In this thesis, we apply the idea to our particular case. Note also that Jund, in his thesis (see [59]), presents a stability study in time of the ordinary differential systems of the first order

$$\frac{dU}{dt} = AU,$$

where A is a skew-symmetric matrix coming from the non-dissipative discretization of a hyperbolic system. Here, in this section, the work is a little bit different because the discretization contains dissipation.

3.5.1 Some basic concept of stability property

We here recall the definition of the stability for the finite-difference schemes. For more details and examples, we refer to [7, 33].

Definition 3.5.1. *A finite-difference scheme, defining recurrence sequences $(\mathbf{w}^n)_{n\in\mathbb{N}}$, is said to be **A-stable** with norm $\|\ \|$ if there is a constant $K > 0$, which does not depend on $\Delta t, \Delta x$ (when they tend to 0) such that*

$$\forall n \in \mathbb{N}, \quad \|\mathbf{w}^n\| \leq K \|\mathbf{w}^0\|, \tag{3.5.1}$$

for any initial vector $\mathbf{w}^0$.

Remark 3.5.2.

- *The existence of $K > 0$ such that we have (3.5.1) might be true only under some conditions on $\Delta t, \Delta x$ must satisfy some conditions in order to exist $K > 0$. Then we say that the numerical scheme is stable under these conditions.*
- *Given two equivalent norms $\|\ \|_1$ and $\|\ \|_2$. A numerical scheme can be stable with respect to norm $\|\ \|_1$ without being stable with respect to norm $\|\ \|_2$.*

Definition 3.5.3. ***Spectral radius***
Given a complex matrix B of dimension $n \times n$

1. *The spectrum of B, denoted $\sigma(B)$, is the set of eigenvalues of B.*
2. *The spectral radius of B, denoted by $\rho(B)$, is defined by*

$$\rho(B) := \sup\{|\lambda|,\ \lambda \in \sigma(B)\}$$

3.5.2 Stability of the space transport part

We here consider the space transport part of the reduced Vlasov equation

$$\mathcal{M}\,\partial_t \mathbf{w} + A\,\partial_x \mathbf{w} = 0. \tag{3.5.2}$$

The semi-discrete approximation in time reads

$$\partial_t \mathbf{w}_i = -\frac{\mathcal{F}(\mathbf{w}_i, \mathbf{w}_{i+1}) - \mathcal{F}(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\Delta x}, \tag{3.5.3}$$

where $\mathcal{F}$ is one of the numerical fluxes defined in section 3.2.2. We here consider the viscous flux and, as a particular case, the centered flux. So, using the viscous flux (3.2.6) for the transport part equation, we have

$$\partial_t \mathbf{w}_i = -(\mathcal{M}^{-1}A)\frac{\mathbf{w}_{i+1} - \mathbf{w}_{i-1}}{2\Delta x} + \frac{\kappa}{2\Delta x}(\mathbf{w}_{i+1} + \mathbf{w}_{i-1} - 2\mathbf{w}_i), \tag{3.5.4}$$

where we recall that $\mathbf{w}_i \in \mathbb{R}^{N_v}$. Denoting $\mathbf{w} \in M_{N_v \times N_x}(\mathbb{R})$ the matrix whose columns are the $(\mathbf{w}_i)$, the scheme writes

$$\partial_t \mathbf{w} = -(\mathcal{M}^{-1}A)\,\mathbf{w}S + \kappa\,\mathbf{w}C, \tag{3.5.5}$$

where S and C are the square matrices of dimension $N_x \times N_x$

$$S = \frac{1}{2\Delta x}\begin{pmatrix} 0 & -1 & 0 & & 0 & 1 \\ 1 & 0 & -1 & & 0 & 0 \\ 0 & 1 & 0 & & 0 & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & & 0 & -1 \\ -1 & 0 & 0 & & 1 & 0 \end{pmatrix},\ C = \frac{1}{2\Delta x}\begin{pmatrix} -2 & 1 & 0 & & 0 & 1 \\ 1 & -2 & 1 & & 0 & 0 \\ 0 & 1 & -2 & & 0 & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & & -2 & 1 \\ 1 & 0 & 0 & & 1 & -2 \end{pmatrix}. \tag{3.5.6}$$

As seen in section 2.4.2, the matrix $\mathcal{M}^{-1}A$ is diagonalizable with real eigenvalues

$$\mathcal{M}^{-1}A = P\Lambda P^{-1} \tag{3.5.7}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{N_v})$ ($\{\lambda_i\}_{i=1\ldots N_v}$ are the eigenvalues) and P is the change-of-basis matrix of size $N_v \times N_v$. Multiplying equation (3.5.5) by P^{-1} and using (3.5.7), we obtain

$$\partial_t(P^{-1}\mathbf{w}) = -\Lambda(P^{-1}\mathbf{w})S + \kappa\,(P^{-1}\mathbf{w})C, \tag{3.5.8}$$

or, denoting $\mathbf{u} = P^{-1}\mathbf{w} \in \mathrm{M}_{N_v \times N_x}(\mathbb{R})$,

$$\partial_t \mathbf{u} = -\Lambda\mathbf{u}S + \kappa\mathbf{u}C. \tag{3.5.9}$$

This gives N_v independent differential equations

$$\partial_t u_j^T = (\lambda_j\, S + \kappa\, C)\, u_j^T, \qquad \forall j = 1 \ldots N_v. \tag{3.5.10}$$

where u_j denotes the j-th line of $\mathbf{u}$ (we use that $S^T = -S$ and $C^T = C$).

For linear equations, the Runge-Kutta schemes are equivalent to Taylor ones. In that case, the Runge-Kutta scheme of order k thus writes

$$(u_j^{n+1})^T = R_k\Big(\Delta t\,(\lambda_j\,S + \kappa C)\Big)\,(u_j^n)^T, \tag{3.5.11}$$

where R_k is the polynom

$$R_k(x) = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^k}{k!}, \tag{3.5.12}$$

Since the matrix S is skew-symmetric and thus normal, we have

Proposition 3.5.4. *Let $\Delta t > 0$. If the following conditions are satisfied*

1. *there exists $K > 0$, independent of Δv, such that $\|P\|_2\|P^{-1}\|_2 \leq K$,*
2. *for all Δx and for all $j = 1 \dots N_v$, the eigenvalues of $(\Delta t(\lambda_j\,S + \kappa\,C))$ belong to the stability domain*
$$\mathcal{S}_k = \{\mu \in \mathbb{C}, \quad |R_k(\mu)| \leq 1\},$$

then the scheme is stable in L^2 norm.

Démonstration. If 2 is true then $\|u^n\| \leqslant \|u^0\|$ (because of (3.5.11)). Since $\mathbf{u} = P^{-1}\mathbf{w}$ and using the stability in the $\mathbf{u}$ variable, we can write

$$\begin{aligned}\|\mathbf{w}^n\|_2 &\leq \|P\|_2\|\mathbf{u}^n\|_2\\ &\leq \|P\|_2\|\mathbf{u}^0\|_2\\ &\leq \|P\|_2\|P^{-1}\|_2\|\mathbf{w}^0\|_2 \leq K\|\mathbf{w}^0\|_2\end{aligned}$$

using the first assumption. □

Proposition 3.5.5. ***Stability domain for*** $k = 1, 2, 3, 4$**.**

Let us denote $\mu = a + bI \in \mathbb{N}$, with $I^2 = -1$. We have :

$$\begin{aligned}|R_1(\mu)|^2 &= |1 + (a+bI)|^2 = (1+a)^2 + b^2,\\ |R_2(\mu)|^2 &= \big|1 + (a+bI) + \frac{(a+bI)^2}{2!}\big|^2 = \left(1 + a + \frac{a^2-b^2}{2}\right)^2 + (b+ab)^2\\ |R_3(\mu)|^2 &= \big|1 + (a+bI) + \frac{(a+bI)^2}{2!} + \frac{(a+bI)^3}{3!}\big|^2\\ &= \left(1 + a + \frac{a^2-b^2}{2} + \frac{a^3}{6} - \frac{ab^2}{2}\right)^2 + \left(b + ab + \frac{a^2b}{2} - \frac{b^3}{6}\right)^2\\ |R_4(\mu)|^2 &= \big|1 + (a+bI) + \frac{(a+bI)^2}{2!} + \frac{(a+bI)^3}{3!} + \frac{(a+bI)^4}{4!}\big|^2\\ &= \left(1 + a + \frac{a^2-b^2}{2} + \frac{a^3}{6} - \frac{ab^2}{2} + \frac{a^4+b^4}{24} - \frac{a^2b^2}{4}\right)^2\\ &\quad + \left(b + ab + \frac{a^2b}{2} - \frac{b^3}{6} + \frac{a^3b - ab^3}{6}\right)^2\end{aligned}$$

We plot the related stability domain in Fig.3.1.

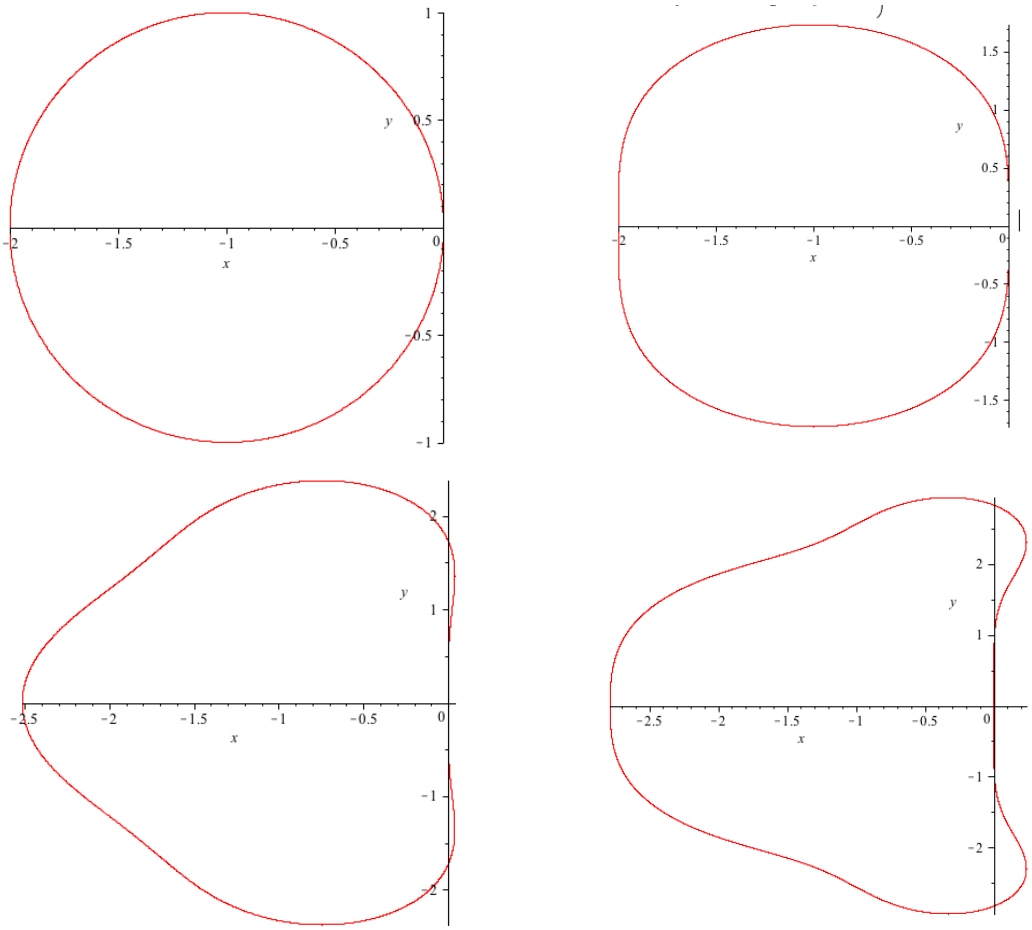


FIGURE 3.1 : Stability domain (in time) : $\mathcal{S}_1$ (top left), $\mathcal{S}_2$ (top right), $\mathcal{S}_3$ (bottom left), $\mathcal{S}_4$ (bottom right).

Lemma 3.5.6. *For any $j = 1, \dots, N_v$, the eigenvalues of the matrix $(\Delta t\,(\lambda_j\, S + \kappa\, C))$ are given by*

$$\mu_{j,i} = \frac{\Delta t\ \lambda_j}{\Delta x} \sin \frac{2\pi i}{N_x} - \frac{\Delta t\, \kappa}{\Delta x}(1 - \cos(\frac{2\pi i}{N_x})), \qquad \textit{for } i = 1 \cdots N_x. \tag{3.5.13}$$

Démonstration. The matrix $(\Delta t(\lambda_j\, S + \kappa\, C))$ is circulant : it writes

$$\Delta t(\lambda_j\, S + \kappa\, C) = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} \\ a_{n-1} & a_0 & a_1 & & a_{n-2} \\ a_{n-2} & a_{n-1} & a_0 & & a_{n-3} \\ \vdots & & & \ddots & \vdots \\ a_1 & a_2 & a_3 & \cdots & a_0 \end{pmatrix}.$$

wit $a_0 = -2\kappa\Delta t/(2\Delta x), a_1 = (\lambda_j + \kappa)\Delta t/(2\Delta x), a_{n-1} = (-\lambda_j + \kappa)\Delta t/(2\Delta x)$ and $a_j = 0$ for

$j \neq 0, 1, (n-1)$. The eigenvalues of $(\Delta\lambda_j S)$ are given by (we use the notation $I^2 = -1$) :

$$\begin{aligned}\mu_{j,i} &= \sum_{j=0}^{n-1} a_j e^{2I\pi jk/n} \\ &= \frac{\Delta t\lambda_j}{2\Delta x}\big(e^{2I\pi k/n} - e^{2I\pi(n-1)k/n}\big) + \frac{\Delta t\,\kappa}{2\Delta x}\big(e^{2I\pi k/n} + e^{2I\pi(n-1)k/n} - 2\big) \\ &= I\,\frac{\Delta t\,\lambda_j}{\Delta x}\sin\frac{2\pi i}{N_x} - \frac{\Delta t\,\kappa}{\Delta x}(1-\cos(\frac{2\pi i}{N_x})) \qquad \text{for } i = 1\cdots N_x. \end{aligned}$$ □

Stability for the centered flux.

Lemma 3.5.7. *The eigenvalues of matrix $\mathcal{M}^{-1}A$ are bounded by V_{max}*

$$\rho(M^{-1}A) \leq V_{max}$$

Démonstration. Let $\lambda \in \mathbb{R}$ an eigenvalue of $\mathcal{M}^{-1}A$ and $u \in \mathbb{R}^{N_v}$ an associated eigenvector. We have

$$\mathcal{M}^{-1}A\,u = \lambda u$$

or equivalently

$$A\,u = \lambda\mathcal{M}\,u$$

Taking the scalar product with u, we have

$$(Au \cdot u) = \lambda(\mathcal{M}u \cdot u)$$

or

$$\sum_{i,j=1}^{N_v}\int_{\Omega_v} v\varphi_i(v)\varphi_j(v)u_ju_idv = \lambda\sum_{i,j=1}^{N_v}\int_{\Omega_v}\varphi_i(v)\varphi_j(v)u_ju_idv.$$

Denoting $g(v) = \sum_{i=1}^{N_v} u_i\varphi_i(v)$, we have

$$\int_{\Omega_v} vg^2(v)dv = \lambda\int_{\Omega_v} vg^2(v)dv.$$

Since u and g do not vanish,

$$|\lambda| = \frac{|\int_{\Omega_v} vg^2(v)|}{|\int_{\Omega_v} g^2(v)|} \leq V_{\max}.$$ □

Proposition 3.5.8. *(case $\kappa = 0$) The Euler and RK2 schemes with the **centered** flux are unstable. For the RK3 and RK4 schemes, there exists a constant $\wp \in (0,1)$ such that the schemes are stable under the CFL condition $\Delta t = \wp\frac{\Delta x}{V_{max}}$.*

Démonstration. According to lemma 3.5.6 (with $\kappa = 0$), we observe that all the eigenvalues of matrix $(\Delta t\,\lambda_j S)$ are purely imaginary and we have

$$\rho(\Delta t\lambda_j S) = \max_{1\leq i\leq N_x}\{|\mu_{j,i}|\} \leq \left|\frac{\Delta t\lambda_j}{\Delta x}\right|. \tag{3.5.14}$$

Therefore, since the eigenvalues have to belong to the stability domain $\mathcal{S}_k$ (see proposition 3.5.4), they should lie on the intersection of $\mathcal{S}_k$ and the imaginary axis. For the Euler and RK2 schemes ($k = 1, 2$), we can see on Fig. 3.1 that this intersection reduces to the origin $(0, 0)$. We can conclude that the Euler and RK2 schemes are unstable with the centered flux.

For the case $k = 3$ or $k = 4$, the intersection of the stability domain $\mathcal{S}_k$ with the imaginary axis is a segment $\{0\} \times [-C, C]$ with $C > 0$. For example, $C = 2\sqrt{2}$ for the RK4 scheme. Then the stability is ensured if

$$\rho(\Delta t \lambda_j S) \leq C.$$

and then, according to (3.5.14), under the condition

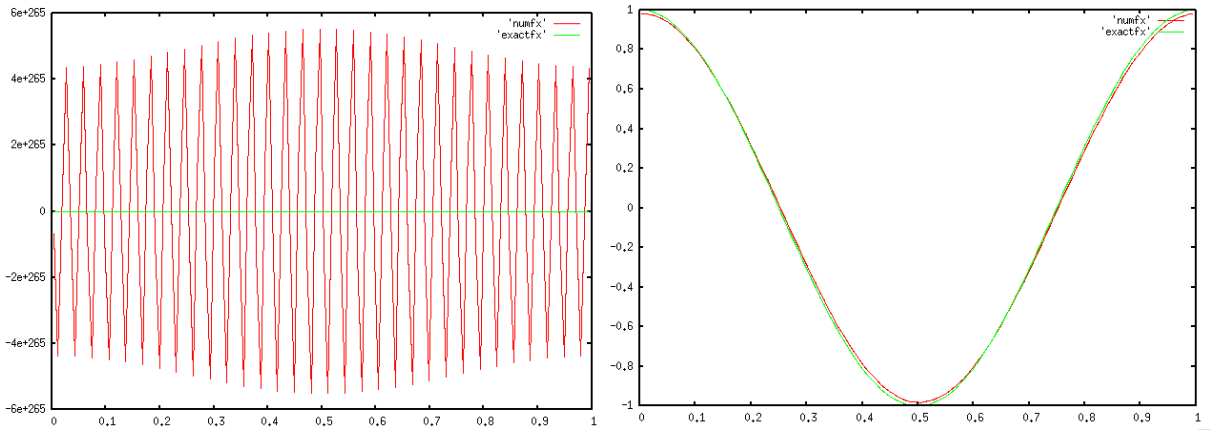
$$\frac{\Delta t\, \rho(M^{-1}A)}{\Delta x} \leq C,$$

where $\max\{|\lambda_j|\}_j = \rho(M^{-1}A)$. Applying the lemma 3.5.7, we have then the result. □

Example 3.5.9. *We here illustrate the stability result. We consider the Vlasov equation 3.1.1. If we suppose that the electric field vanishes at any time and position, we obtain the transport in space equation.*

$$\partial_t f + v\, \partial_x f = 0 \tag{3.5.15}$$

We consider the initial function $f_0(x, v) = \sin(\pi x)$*, the domain* $\Omega_x \times \Omega_v =]-1, 1[\times]-1, 1[$*. We use the centered flux and the following numerical parameters :* $\mathrm{d} = 2, M = 8, N_v = \mathrm{d}M + 1 = 17, N_x = 128, \wp = 0.6, T = 50, \beta = 1/2$*. With the RK2 scheme, the numerical solution becomes unstable (see Fig.3.2 left), while using the RK3 or RK4 scheme with the same parameters, we obtain a stable numerical solution (Fig.3.2 in the right).*



RK2 RK3

FIGURE 3.2 : Distribution function in $x-$ transport test case. Parameters : $\mathrm{d} = 2, M = 8, N_v = 17, N_x = 128, \wp = 0.6, T = 50, \beta = 1$. The numerical solution explose with the second order scheme (left) and remain stable with the third order scheme (right) (in time).

Stability for the viscous flux

Proposition 3.5.10. *(case $\kappa > 0$) The Runge-Kutta schemes for $k = 1, 2, 3, 4$ with the* ***viscous*** *flux are all stable when taking Δt sufficiently small :*

- *The RK1 scheme is stable iff the condition* $\Delta t = \wp \frac{\Delta x}{V_{max}}$ *with* $\wp \in (0,1)$ *and* $\Delta x/\Delta t > \kappa > V_{max}\wp$.
- *The RK2 scheme is stable if* $\Delta t = \wp \frac{\Delta x}{V_{max}}$ *with* $\wp \in (0,\sqrt{3})$ *and* $\Delta x/\Delta t > \kappa > (1/4)V_{\max}\wp^3$ *(not proved, numerically checked).*
- *The RK3 scheme is stable if* $\Delta t = \wp \frac{\Delta x}{V_{max}}$ *with* $\wp \in (0,1.5)$ *and* $(3/4)\Delta x/\Delta t > \kappa$ *(non optimal).*
- *The RK4 scheme is stable if* $\Delta t = \wp \frac{\Delta x}{V_{max}}$ *with* $\wp \in (0,2)$ *and* $(3/4)\Delta x/\Delta t > \kappa$ *(non optimal).*

We note that, at a given time step Δt, there is no condition on the diffusion parameter κ for the RK3 and RK4 schemes : it can be taken as small as we want. It is in accordance with the results of proposition 3.5.8 where the RK3 and RK4 schemes are proved to be stable with $\kappa = 0$.

Démonstration. According to lemma 3.5.6, the eigenvalues of $(\Delta t(\lambda_j\, S + \kappa\, C))$ are given

$$\mu_{j,i} = \frac{\Delta t \lambda_j}{\Delta x}\left(-\frac{\kappa}{\lambda_j}(1-\cos\frac{2\pi i}{N_x}) + I\sin\frac{2\pi i}{N_x}\right) \qquad \text{for } i = 1\cdots N_x.$$

Introducing the function

$$f_\pm(x) = -\frac{\kappa}{|\lambda_j|}(1 \pm \sqrt{1-x^2}),$$

the coefficient $\alpha_j = \Delta t|\lambda_j|/\Delta x$ and the variable $y = \alpha_j \sin(\theta)$, the eigenvalue condition (see proposition 3.5.4) reads : the curve $y \in [-\alpha_j, \alpha_j] \mapsto (\alpha_j f_\pm(y/\alpha_j) + Iy)$ have to be included in the stability domain $\mathcal{S}_k$ (for all j). In figure 3.5.2, we represent the eigenvalues for j such that $|\lambda_j| = \max|\lambda_j|_i$ and $\kappa = 0.5$, $\Delta x = 0.5$. We also plot the stability domain of the RK2 and RK4 schemes. The eigenvalues are located on an ellipse whose extreme points are given by $(0,0)$, $(-\frac{\Delta t\kappa}{\Delta x}, \pm\alpha_j)$ and $(-2\frac{\Delta t\kappa}{\Delta x}, 0)$. Thanks to the expressions of proposition 3.5.5, we have :

$$\begin{aligned} R_1(\mu) = 1 &\Leftrightarrow 2a + a^2 + b^2 = 0 \\ &\Leftrightarrow a_\pm = -1 \pm \sqrt{1-b^2}, \\ R_2(\mu) = 1 &\Leftrightarrow a_\pm = -1 \pm \sqrt{2\sqrt{b^2+1} - 1 - b^2}, \end{aligned}$$

where only the real roots are written. Since $(-\frac{\Delta t\kappa}{\Delta x}, \pm\alpha_j)$ is an extreme point of the ellipse of eigenvalues, one necessary condition to obtain stability is that :

$$\begin{aligned} \alpha_j &\leqslant \max_{(a,b)\in\mathcal{S}_1} b = 1, && \text{for the RK1 scheme,} \\ \alpha_j &\leqslant \max_{(a,b)\in\mathcal{S}_2} b = \sqrt{3}, && \text{for the RK2 scheme,} \end{aligned}$$

which give us the CFL condition : $\max\alpha_j = \Delta t V_{\max}/\Delta x \leqslant 1$ for the RK1 scheme and $\max\alpha_j = \Delta t V_{\max}/\Delta x \leqslant \sqrt{3}$ for the RK2 scheme. Since $(-2\frac{\Delta t\kappa}{\Delta x}, 0)$ is an extreme point of

the ellipse of eigenvalues, one necessary condition to obtain stability is that :

$$-2\frac{\Delta t\kappa}{\Delta x} \geqslant \min_{(a,b)\in\mathcal{S}_1} a = -2, \qquad \text{for the RK1 scheme,}$$
$$-2\frac{\Delta t\kappa}{\Delta x} \geqslant \min_{(a,b)\in\mathcal{S}_2} a = -2, \qquad \text{for the RK2 scheme,}$$

which give us the condition : $\Delta t\,\kappa \leqslant \Delta x$ for the RK1 and the RK2 scheme. For the RK1 and RK2 scheme, one another necessary condition is that the curvature of the eigenvalues curve is larger than the curvature of the stability boundary at the origin $(a, b) = (0, 0)$. The stability domain have the following expansions at the origin :

$$a_+ = -\frac{b^2}{2} + o(b^3), \qquad \text{for the RK1 scheme,}$$
$$a_+ = -\frac{b^4}{8} + o(b^5), \qquad \text{for the RK2 scheme,}$$

and for the eigenvalue curve :

$$\begin{aligned} x = \alpha_j f_-(y/\alpha_j) &= -\frac{\alpha_j\kappa}{2|\lambda_j|}\left(\frac{y}{\alpha_j}\right)^2 - \frac{\alpha_j\kappa}{4|\lambda_j|}\left(\frac{y}{\alpha_j}\right)^4 + o(y^6) \\ &= -\frac{\kappa}{2|\lambda_j|\alpha_j}y^2 + o(y^6). \end{aligned}$$

Hence, for the RK1 scheme, $\kappa/(2|\lambda_j|\alpha_j) > 1/2$ for all j and then $\kappa > \max|\lambda_j|\alpha_j = V_{\max}\wp$. For the RK2 scheme, the curvature condition gives : $\kappa > 0$. The asymptotic curves intersect at $y = \sqrt{\frac{4\kappa}{|\lambda_j|\alpha_j}}$. One possible natural condition to impose is :

$$\sqrt{\frac{4\kappa}{|\lambda_j|\alpha_j}} \geqslant \max_{(a,b)\in\text{the eigenvalue curve}} b = \alpha_j$$

This is equivalent to $\kappa \geqslant (1/4)V_{\max}\max\alpha_j^3$. We numerically check that it is valid for a large range of parameters. For the RK3 scheme, we can check that the rectangle $((-1.5,-1.5),(1.5,1.5),(0,1.5),(0,-1.5))$ is included into the stability domain $\mathcal{S}_3$ (see figure 3.1). Expressing that the extreme points of the ellipse of eigenvalues belong to this sub-domain give us the conditions $\max\alpha_j = \Delta t V_{\max}/\Delta x \leqslant 1.5$ with $\wp \in (0, 1.5)$ and $1.5 > 2\kappa\Delta t/\Delta x$. Similarly, the conditions for the RK4 scheme result from the fact that the rectangle $((-1.5,-2),(1.5,2),(0,2),(0,-2))$ is included into the stability domain $\mathcal{S}_4$ (see figure 3.1). □

3.5.3 Stability for the source term equation

In this section, we study the stability of the Runge-Kutta schemes for transport in the velocity direction. We consider a constant electric field $E = 1$. The reduced Vlasov equation simplifies to

$$\partial_t \mathbf{w} + \mathcal{M}^{-1}B(E)\,\mathbf{w} = 0. \tag{3.5.16}$$

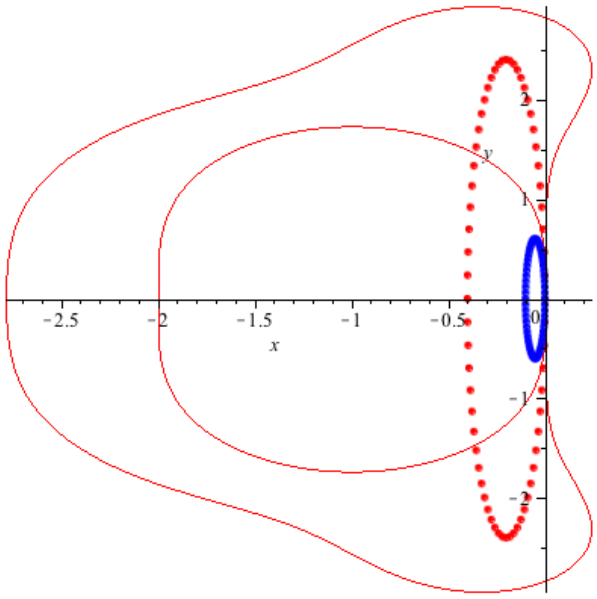


FIGURE 3.3 : Stability domain (in time) of order 2, 4 and the eigenvalues of matrix K_j with the viscous flux($\kappa = 0.5$). Parameters : $\Delta_x = 0.5$, $\Delta t = 0.05$ (blue points) and $\Delta t = 0.2$ (red points).

Still using the linearity of the equation, the k-th Runge-Kutta scheme writes

$$\mathbf{w}_i^{n+1} = -R_k(\mathcal{M}^{-1}B(E))\,\mathbf{w}_i^n, \quad \forall i = 1, \ldots, N_x$$

The Von-Neumann stability of the scheme is ensured if the eigenvalues of $\mathcal{M}^{-1}B$ lie in the stability domain $\mathcal{S}_k$. For simplicity reasons, we have computed the mass matrix $\mathcal{M}$ and B using Gauss-Lobatto quadrature instead of Gauss-Legendre. Matrix $\mathcal{M}$ in this case is diagonal so we can easily obtain the inverse matrix $\mathcal{M}^{-1}$. In Fig.(3.4), we plot the eigenvalues of $\mathcal{M}^{-1}B(E)$ numerically and the stability domain for the RK2 and RK4 schemes.

3.5.4 Stability of the full reduced Vlasov equation

Now, we will study the stability of the full reduced Vlasov equation (3.1.9). There existes a matrix of dimension $(N_v \times N_x)^2$ such that

$$\partial_t \mathbf{w} = \mathcal{A}\mathbf{w}. \tag{3.5.17}$$

As studied above, the stability of numerical scheme depends on the eigenvalues of matrix $\mathcal{A}$. It is difficult to determine theoricaly the eigenvalues of matrix $\mathcal{A}$. In this thesis, we just try to compute the matrix $\mathcal{A}$ and its eigenvalues numerically. Namely, in our Fortran 1D code, we compute the matrix $\mathcal{A}$ after the first time step, and then we copy this matrix in Maple in order to compute its eigenvalues. The results are represented in Fig.(3.5) : we plot the eigenvalues of this matrix in two cases : with centered flux(left) and viscous flux(right). We have chosen a time step Δt satisfying the stability conditions both of the x-transport and v-transport steps

$$\frac{\Delta t}{\Delta x V_{\max}} < 1, \quad \frac{\Delta t}{\Delta v \max E} < 1.$$

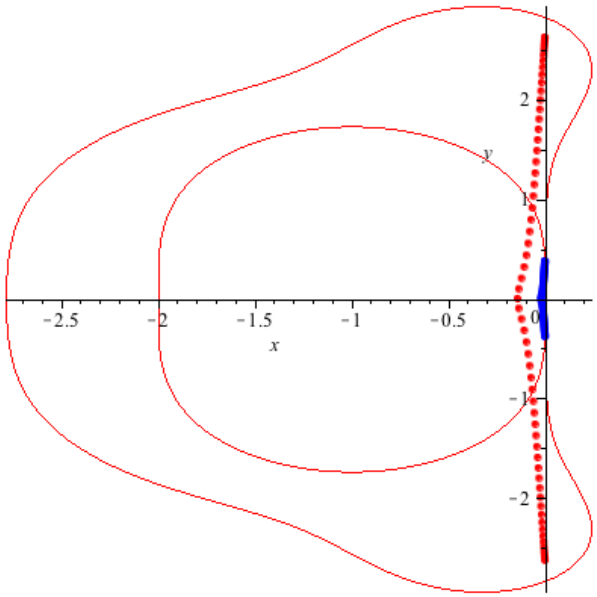


FIGURE 3.4 : Boundaries of the stability domains $\mathcal{S}_2$ and $\mathcal{S}_4$ and eigenvalues of the matrix $\mathcal{M}^{-1}B(E)$ in the complex plane (computed with Gauss-Lobatto integration rule). Parameters : d = 2, $M = 30$, $V_{\max} = 6$, $E = 1$, $\Delta t = 0.05$ (RK2) (the blue points), $\Delta t = 0.35$ (RK4) (red points).

We then observe that :

1. In the two cases we have stability of the numerical scheme with the RK3 or RK4 in time under some CFL condition.

2. For the RK2 scheme : for the viscous flux, we observe the stability of the scheme that is predicted by theory. In the case of centered flux, we observe small instabilities, that would grow when the final time increases.

3.6 Test cases

For testing our schemes, we consider several test cases. The first two tests are designed in order to validate the pure transport in the space and velocity direction. In particular, we will show that the correction (3.1.10), (3.1.11) is essential in order to obtain correct results. In our numerical experiments, the discretization parameters are $M = 20$, d = 5 and $N_x = 128$.

3.6.1 Transport test cases

The velocity transport equation

If we suppose that the distribution function f is space homogenous $\partial_x f = 0$ and the electric field is constant in time, namely $E(x,t) = E(x,t=0) = E_0(x)$, thus the Vlasov equation becomes

$$\partial_t f + E_0(x)\,\partial_v f = 0, \tag{3.6.1}$$

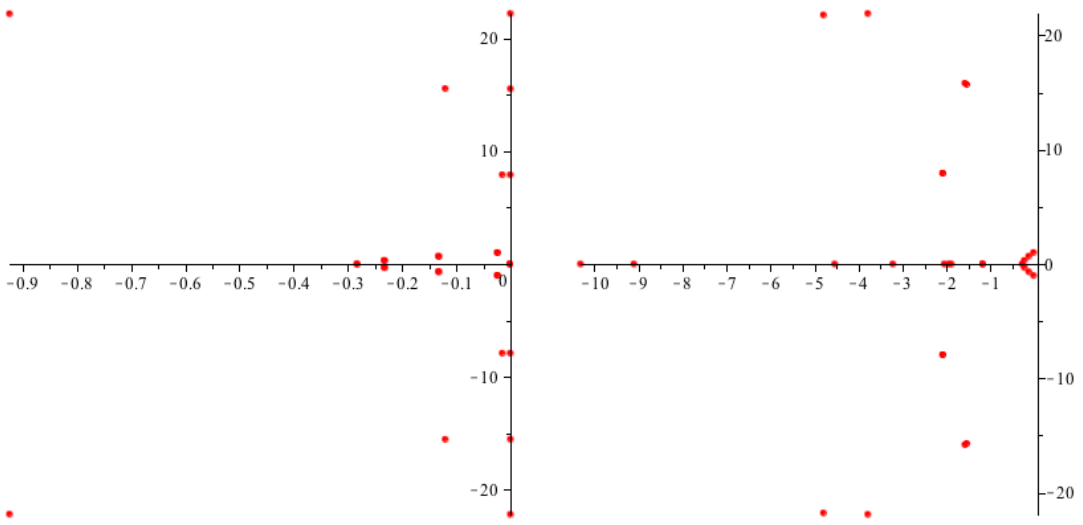


FIGURE 3.5 : Eigenvalues of matrix $\mathcal{A}$ with the centered flux (in the left) and with the viscous flux (in the right). Parameters : $\mathrm{d} = 2, M = 3, N_v = 7, N_x = 4$.

which is a v-transport equation. Consider the v-transport equation (3.6.1) with a given initial condition f_0. Thanks to the method of characteristics, the exact solution is given by

$$f(x, v, t) = f_0(x, v - E_0(x)t).$$

We consider the computation domain $v \in]-1, 1[$ and $x \in]0, 1[$ and the following space-homogenerous initial condition

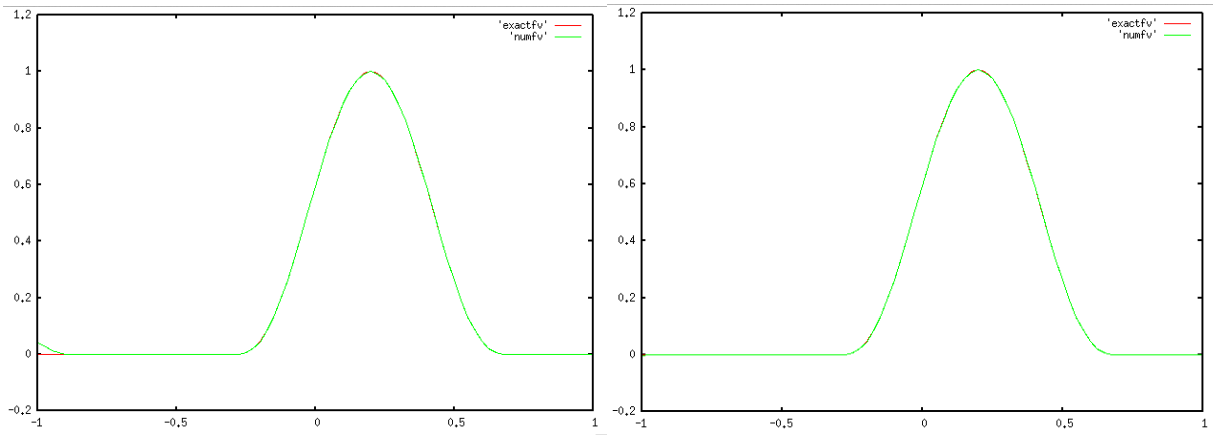
$$f_0(x, v) = \begin{cases} (1 + (-3 + (3 - 4v^2)4v^2)4v^2) & \text{if } v \in [-1/2, 1/2]; \\ 0 \quad \text{if not,} \end{cases}$$

with a constant electric field $E_0(x) = 1$. The parameters are fixed $N_x = 16$, $\mathrm{d} = 2$, $T = 0.2$.

Let us verify the choice of β in the weak formulation 2.2.4. With the parameters $\wp = 0.1$ and $M = 40$, we plot the numerical distribution function and the exact distribution function in Fig. 3.6. If the parameter β is chosen to be 0, we remark that the numerical solution differs from the exact solution at the left boundary of the velocity domain (on Fig.3.6 left). They do not not occur if we choose $\beta = 1/2$ or $\beta = 1$ (on Fig.3.6 right). This is due to the fact that the outflow boundary conditions is not taken into account if β equals 0 (see section 2.2).

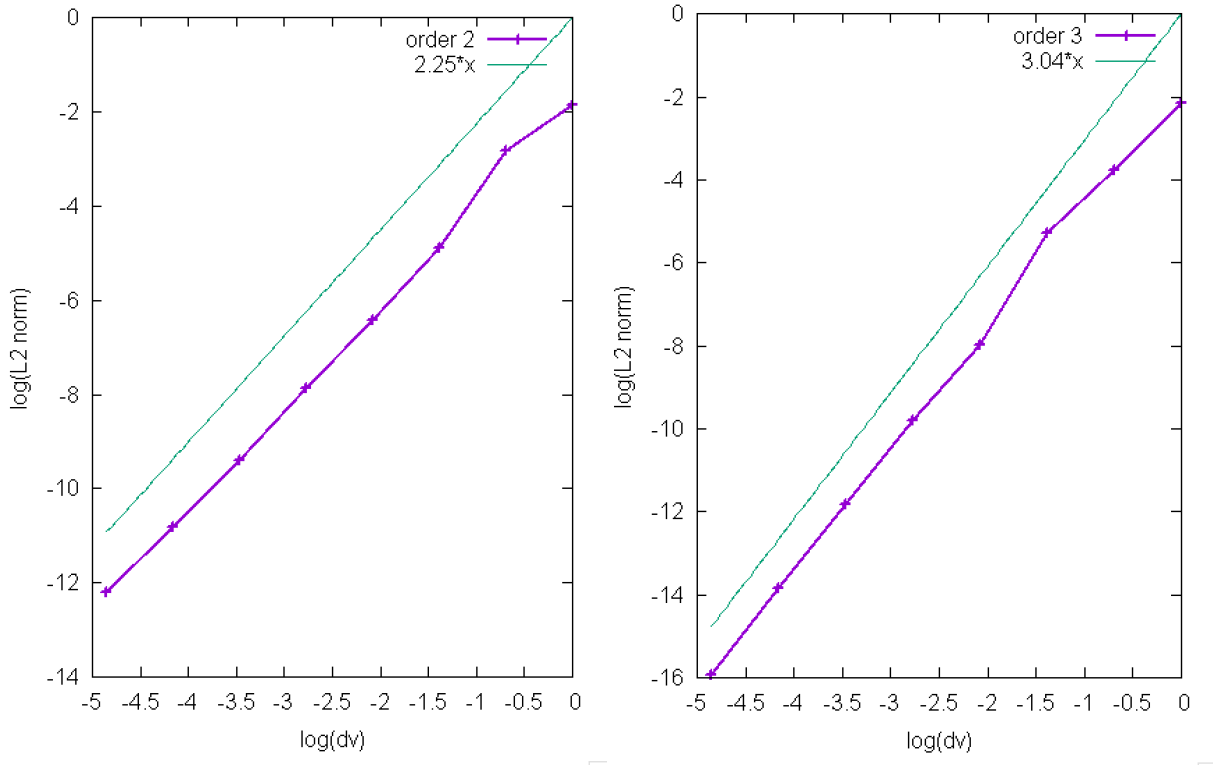
Now, let us study the order of convergence in velocity. With the same test case, we compare the effects of changing the degree of velocity polynomial $\mathrm{d} = 2$ and $\mathrm{d} = 3$. By using the viscous flux, the convergence rate in space is 1, it is the reason that we have to take $N_x = 128$ so that we can avoid the effect of convergence rate in space. With the RK4 scheme in time, theorically, we have the order 4 in time. The result we obtain is the convergence of order 2.25 and 3.04 as shown in Fig.3.7.

Remark 3.6.1. *Because of the proposition 2.5.6, the convergence rate with respect to velocity, in theory, is the degree* d *of the polynomial basis. Note that here we observe a slightly higher numerical convergence rate. In some other papers, they call this phenomenon the "super convergence". In our case, it is maybe due to the uniform mesh and the exact computation of the matrices* $\mathcal{M}, A, B$.



$\beta = 0$ $\beta = 1/2$ or $\beta = 1$

FIGURE 3.6 : Distribution function in v-transport test case. Parameters : d $= 2, M = 40, N_v = \mathrm{d}M + 1, N_x = 16, \wp = 0.1, T = 0.2$.



d = 2 d = 3

FIGURE 3.7 : Convergence in v-transport test case. Parameters : upwind flux, RK4, $\kappa = 0.005, \wp = 0.1, N_x = 128$ and $T = 0.2$.

The space transport equation

If we suppose that the electric field vanishes at any time and any position, the Vlasov equation becomes the x-transport equation

$$\partial_t f + v\, \partial_x f = 0 \tag{3.6.2}$$

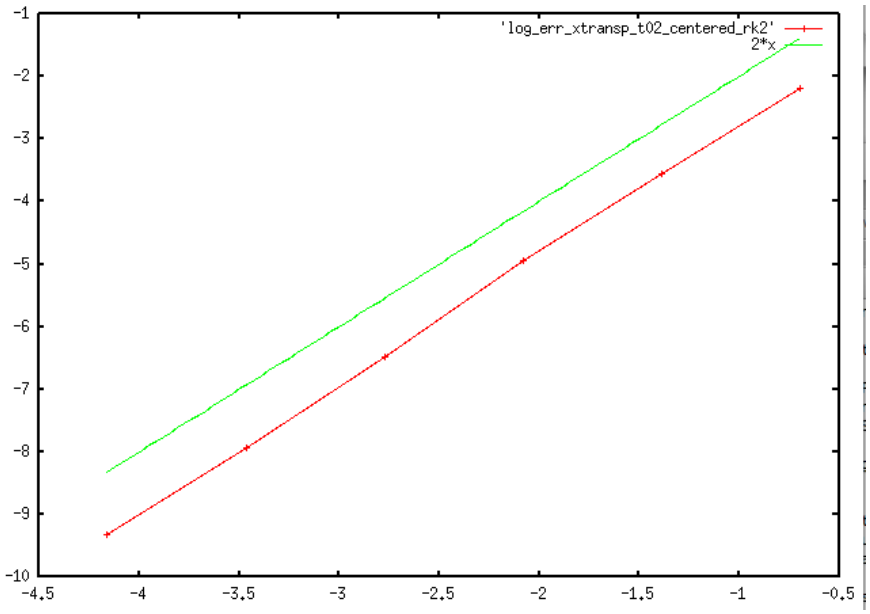


FIGURE 3.8 : Order of convergence in space for the x-transport test case of the Vlasov equation after $T = 0.2$. Parameters : d $= 2, M = 8$ and $\wp = 0.5$, $\beta = 1$ (weak-formula), centered flux, RK2

We consider the x-transport equation (3.6.2) with a given initial condition $f_0 = \sin(\pi x)$ and the computation domains $\Omega_x = \Omega_v =]-1, 1[$. The exact solution of this transport equation is given by

$$f(x, v, t) = f_0(x - vt, v).$$

With the parameters d $= 2, M = 8$ and $\wp = 0.5$, $\beta = 1$ (weak-formula), centered flux, RK3, and $N_x = 4, 8, \ldots, 128$, we obtain an order of convergence in space of 2 (case of centered flux) as in Fig. 3.8

3.6.2 Landau damping test cases

In this test case, we consider the following initial data

- The distribution function

$$f_0(x, v) = (1 + \varepsilon \cos(kx)) \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}}, \tag{3.6.3}$$

- the electric field

$$E_0(x) = \frac{\varepsilon}{k} \sin(kx),$$

- the space domain $\Omega_x =]0, L[$ with $L = 2\pi/k$.

We approximate the electric energy by

$$\mathcal{E}(t_n) = \sqrt{\int_{\Omega_x} E^2(x, t)\, dx} \simeq \sqrt{\triangle x \sum_{i=1}^{N_x} (E_i^n)^2} \tag{3.6.4}$$

For small ε, thanks to a linear approximation of the non-linear Vlasov-Poisson system, it is possible to compute an approximate analytic solution of the electric field. The details of the computation are given in [89]. The electric field is given by

$$E(x,t) = 4\varepsilon r e^{\omega_i t}\sin(kx)\cos(\omega_r t - \varphi), \tag{3.6.5}$$

where ω_i, ω_r are the real part and the imaginary part of ω, respectively. The numerical values of ω, r and φ are given in the following table

k	ω	$re^{I\varphi}$
0.5	$\pm 1.4156 - 0.1533I$	$0.3677e^{\pm I 0.536245}$
0.4	$\pm 1.2850 - 0.0661I$	$0.424666e^{\pm I 0.3357725}$
0.3	$\pm 1.1598 - 0.0126I$	$0.63678e^{\pm I 0.114267}$
0.2	$\pm 1.0640 - 5.510 \times 10^{-5} I$	$1.129664e^{\pm I 0.00127377}$

In addition, the distribution function can be approximated with a well validated method, such as the PIC method. We compare our numerical results with the PIC results and also with the analytic solution.

With the different choices of pairs of parameters (k, ε), we obtain different results. Now, let us consider the 3 following cases :

Landau damping 1

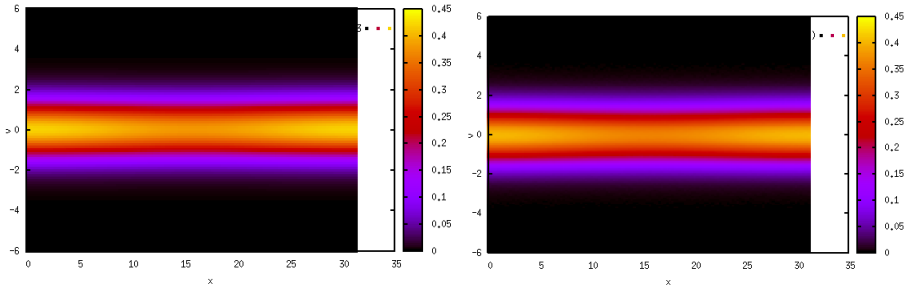
The value of parameters are $k = 0.2$ and $\varepsilon = 5 \times 10^{-2}$. We compare the distribution function of the reduced Vlasov-Poisson method and of the PIC method (taken from [24]). We plot the distribution function computed by the two methods at different times $t = 0$, $t = 20$, $t = 40$ and at $t = 100$. The results are on Figure 3.9

The reduced Vlasov approximation satisfies only an L^2 stability estimate (2.4.9). Such estimate does not ensure the positivity of f. Indeed, in our simulations we observe at some points slightly negative values of f. But the numerical results are anyway very satisfactory. We also plot the logarithm of the electric energy in order to compare the reduced Vlasov-Poisson method with the PIC method on Figure 3.10.

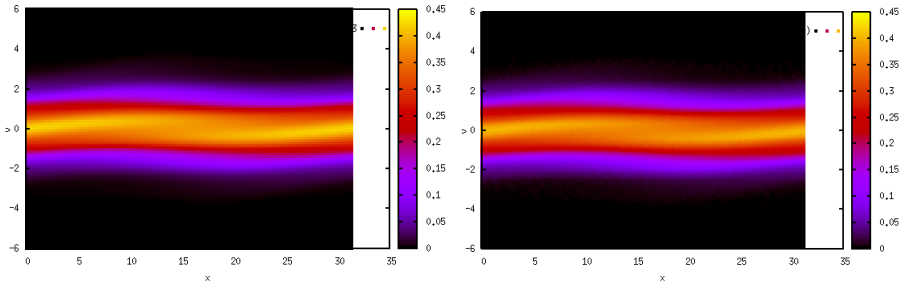
Remark 3.6.2.
- *In order to have the same scale with the results obtained with the PIC method, we plot Fig. 3.9 in the scale* $[0, 0.45]$.
- *We could avoid negative values of f by considering a non-linear entropy in the Vlasov equation 2.2.1 or by a WENO or TVD method (for more details, see [71, 83]).*
- *In comparing with the result obtained by PIC method, we observe that the distribution function and electric field obtained with our method are very similar to the ones obtained with the PIC method.*

Landau damping 2

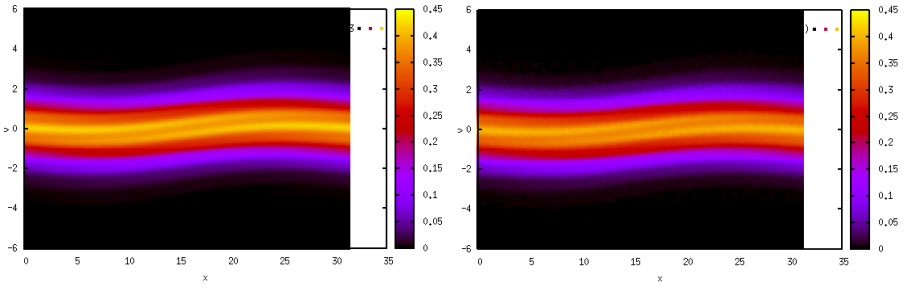
Now, we take the parameters $k = 0.5$ and $\varepsilon = 5 \times 10^{-3}$. In Figure 3.11, we plot the logarithm of the electric energy computed by reduced Vlasov-Poisson method and by the formula (3.6.5). We obtain the good damping rate (0.1533), which is predicted by theory



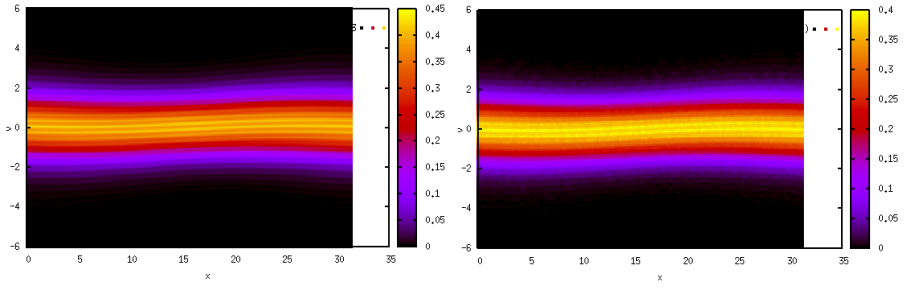
FV-FE method (t=0) PIC method (t=0)



FV-FE method (t=20) PIC method (t=20)



FV-FE method (t=40) PIC method (t=40)



FV-FE method (t=100) PIC method (t=100)

FIGURE 3.9 : The distribution function of the Landau damping 1 test case. Left : FV-FE method. Right : PIC method.

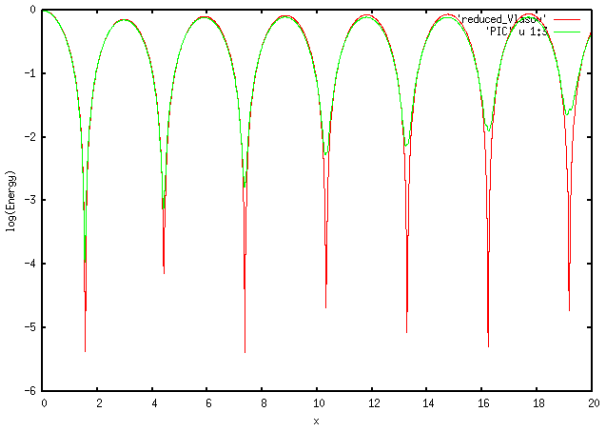


FIGURE 3.10 : The electric energy of the Landau damping 1 test case up to time $t = 20$, the green curve is computed with the PIC method and the red curve is computed with the reduced Vlasov-Poisson method.

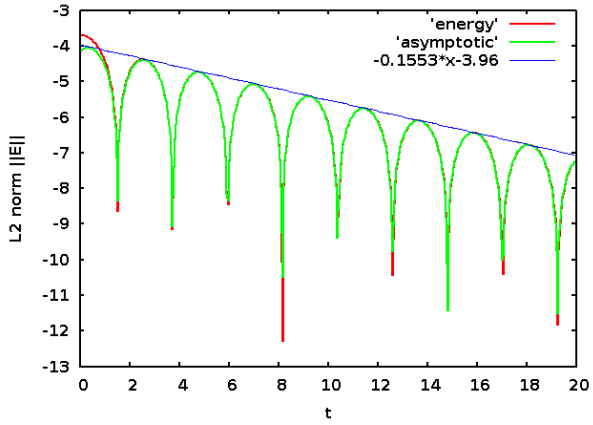


FIGURE 3.11 : Landau damping 2 test case : L^2 norm of the electric field $||E||_2$ as a function of time. The green curve "asymptotic" is the analytical solution and the red curve "energy" is computed with the reduced Vlasov-Poisson method, RK3 scheme with the slightly viscous flux. Parameters : d $= 2$, $M = 32$, $N_x = 64$, $\beta = 1$, $\kappa = 0.005$, $\Delta t = 0.05$.

(see Fig.3.11). As seen in the section 3.5, we just remark that with the Euler or Rung-Kutta 2 scheme, we have to decrease the CFL condition, (it means smaller time steps) for exemple $\Delta t = 0.005$ but not with the RK3 or RK4 scheme. For example, with the same parameter, d $= 2$, $M = 32$, $N_x = 64$, $\beta = 1$, the viscous flux with $\kappa = 0.005$, if the time step $\Delta t = 0.05$, the Euler or RK2 are unstable while the RK3 or RK4 schemes give good results.

We have also tested the reduced Vlasov-Ampère resolution. We have observed that it is not possible to use a fully explicit time integration for the Ampère equation (3.1.5). If we compute E_i^{n+1} from w_i^n by the first order explicit time integration, for instance, the results are not very precise and we observe an error increase as time goes by. We obtain better results if we use w_i^{n+1} for estimating $\partial_t E(x_i, t_n)$. This is a kind of semi-implicit scheme. The results are compared on Figure 3.12.

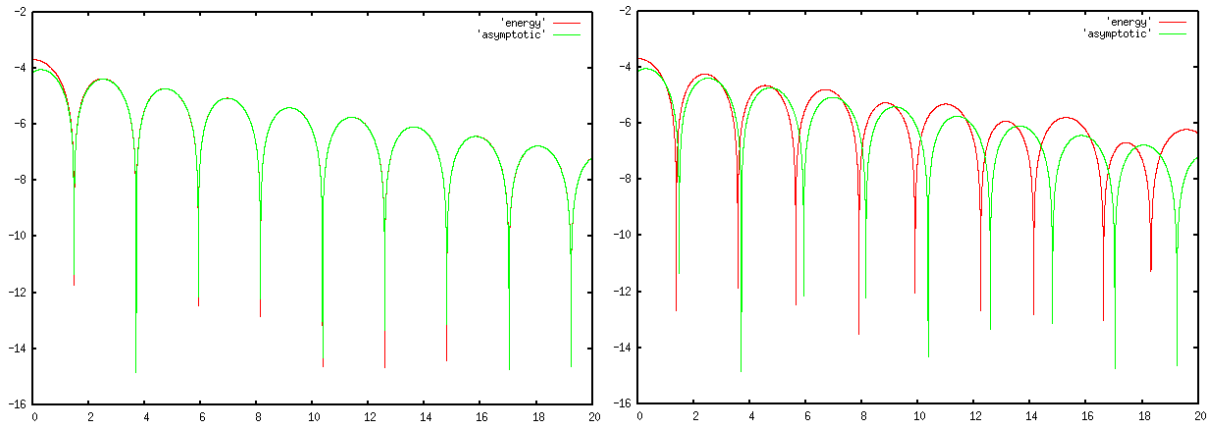


FIGURE 3.12 : The electric energy of the Landau damping test case up to time $t = 20$, the green curve "asymptotic" is computed with the formula (3.6.5) and the red curve "energy" is computed with the reduced Vlasov-Ampère method. Left : semi-implicit scheme. Right : explicit scheme.

Landau damping 3 (Strong landau damping)

In this case, we take the parameters $k = \varepsilon = 0.5$ in the equation (3.6.3). This choice of k makes the electrostatic Langmuir waves strongly Landau damped. In some others papers, for instance [38], this test case is called strong landau damping. The number of cells $N_x = 32$ in the $x-$ direction, and $N_v = 2 \cdot 32 + 1 = 65$ in v-direction (namely the degree d = 2 and the number of element $M = 32$). We plot the mass, the L^1 norm in Fig.3.13, the L^2 norm and the relative error of L^2 norm (Fig. 3.14) of the distribution function in order to observe the conservation. We compare the results between the Runge-Kutta 2 scheme with the viscous flux and the Runge-Kutta 4 with the centered flux. The time step here is 10^{-2} with the RK4, 10^{-3} with the RK2.

We can observe that : With the centered flux, we have order 2 in space, so the RK4 scheme with centered flux conserves better the L2 norm of the distribution function than the RK2 scheme with viscous flux (which is of order 1 in space)(see Fig.3.14). We can remark that the L^2 norm is not conserved exactly because we don't use a periodic condition in velocity. In contrast, the viscous flux introduces a numerical diffusion so it keeps in a better way the positivity of the distribution function (i.e. the L^1 norm) than the centered flux (see Fig.3.13).

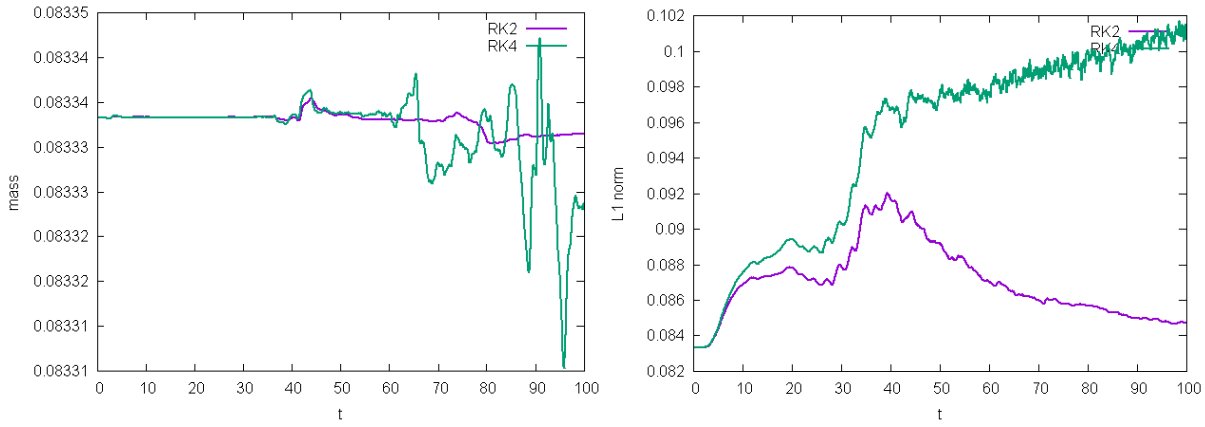
Remark 3.6.3.

- *The formulation used for the relative error in L^2 norm is given by*

$$\frac{\|\mathbf{w}(t)\|_{L^2} - \|\mathbf{w}_0\|_{L^2}}{\|\mathbf{w}_0\|_{L^2}}$$

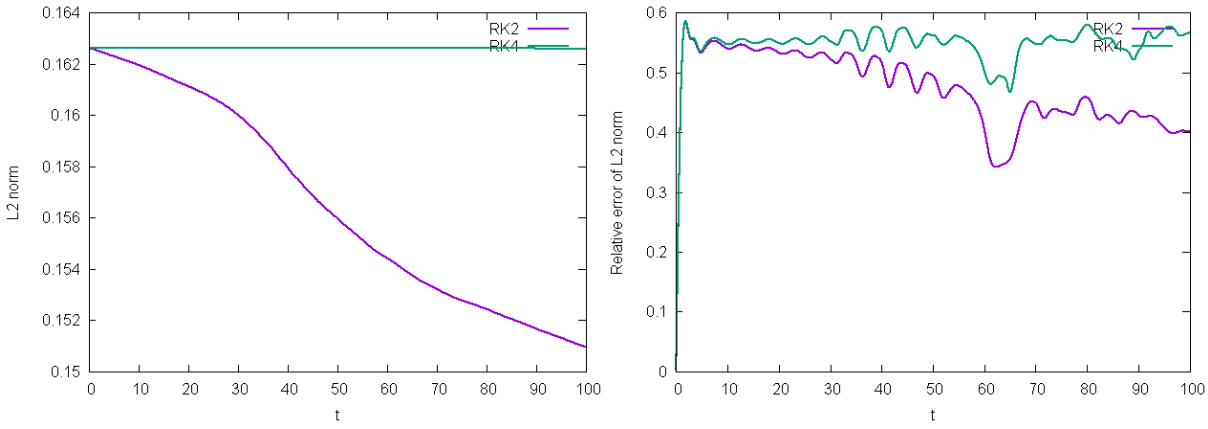
- *The results obtained with the RK4 method and a viscous flux is almost the same with the results obtained with the RK2 method and a viscous flux. For this reason, we don't display the curve of mass, L^1 and L^2 norm for the case RK4 with viscous flux.*

- *In case of using RK4, we can take a bigger time step, for instance, $\Delta t = 0.1$ for the above test case, and the numerical scheme remains stable.*



Mass L1 norm

FIGURE 3.13 : Strong landau damping test case : the mass and L^1 norm of the distribution function as functions of time. Comparison between the RK2 scheme viscous flux ($\kappa = 0.005$, $\Delta t = 10^{-3}$) and with the RK4 scheme centered flux ($\kappa = 0, \Delta t = 10^{-2}$). Numerical paramaters : $V = 6, N_x = 64, N_v = 65$.



L2 norm Relative error of L2 norm

FIGURE 3.14 : Strong landau damping test case : the L^2 norm (right) and relative error of L2 norm (left) of the distribution function as functions of time. Comparison between the RK2 scheme viscous flux ($\kappa = 0.005$, $\Delta t = 10^{-3}$) and with the RK4 scheme centered flux ($\kappa = 0, \Delta t = 10^{-2}$). Numerical paramaters : $V = 6, N_x = 64, N_v = 65$.

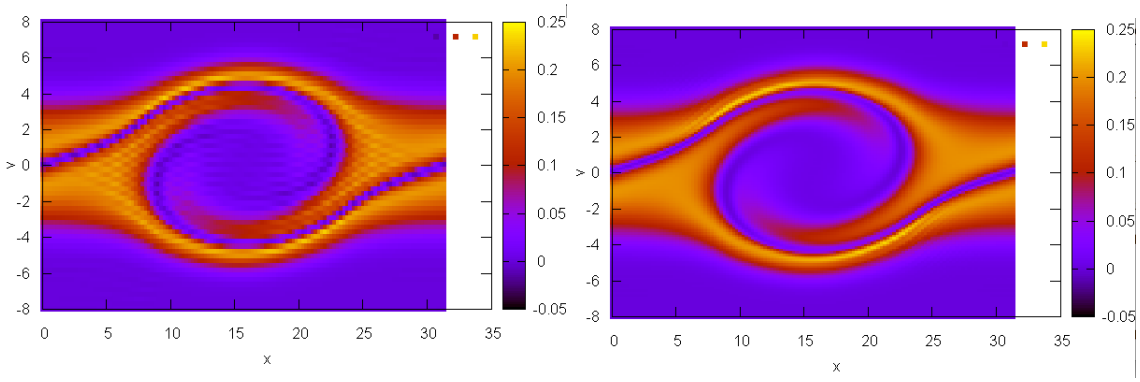


FIGURE 3.15 : The distribution function of the two-stream test case at time $t = 25$ computed with the reduced Vlasov-Poisson method with d = 2 and different values of M. Left : $M = 30$. Right : $M = 60$.

3.6.3 Two-stream instability

In this test case, the initial distribution function is given by

$$f_0(x,v) = (1 + \varepsilon \cos(kx)) \frac{1}{2\sqrt{2\pi}} \left(e^{-\frac{(v-v_0)^2}{2}} + e^{-\frac{(v+v_0)^2}{2}} \right),$$

in which the velocity v_0 is given. The value of parameters for this test case are $k = 0.2$, $\varepsilon = 5 \times 10^{-3}$ and $v_0 = 3$, $V = 8$.

Firstly, we remark that we have to take a large enough velocity discretization parameter N_v in order to reach good precision. There are two ways to do that : increasing the degree d, or increasing the number of elements M. For example, Figure 3.15 represents the distribution function at time $t = 25$ with a polynomial degree d = 2 and a number of element in velocity M equal to 30 and 60 respectively. We observe a better precision (less oscillations) with $M = 60$ than with $M = 30$. Note that with the reduced Vlasov method, we obtain a few slightly negative values of the distribution function.

Now, we plot the distribution function at times $t = 0$, $t = 15$, $t = 20$, $t = 25$ and $t = 50$ computed by PIC method or reduced Vlasov-Poisson method (with the RK3 scheme, viscous flux $\kappa = 0.005$) in Figures 3.16.

We also try the RK3 scheme with the centered flux and for the same test case as above (two streams instability). At time $t = 50$, we observe small numerical oscillations of the distribution function (Fig. 3.17 in the left). These oscillations are due to the fact that we have no upwind mechanism in the resolution of the transport equation with the centered numerical flux. The oscillations disappear when we use the upwind flux for the same RK3 scheme (Fig. 3.17 in the right).

In order to provide a better understanding of the dissipation, we also introduce in Appendix (C) a non-linear version of the reduced Vlasov approach. The non-linear approach allows to replace the energy estimate by an entropy estimate. It also allows to construct natural dissipative source term, which can be used for stabilizing the numerical method or for introducing a physical dissipative mechanism, such as collisions effect.

In chapter 8, we implement an upwind Discontinuous Galerkin (DG) method in order to introduce dissipation in the numerical method, while keeping high order.

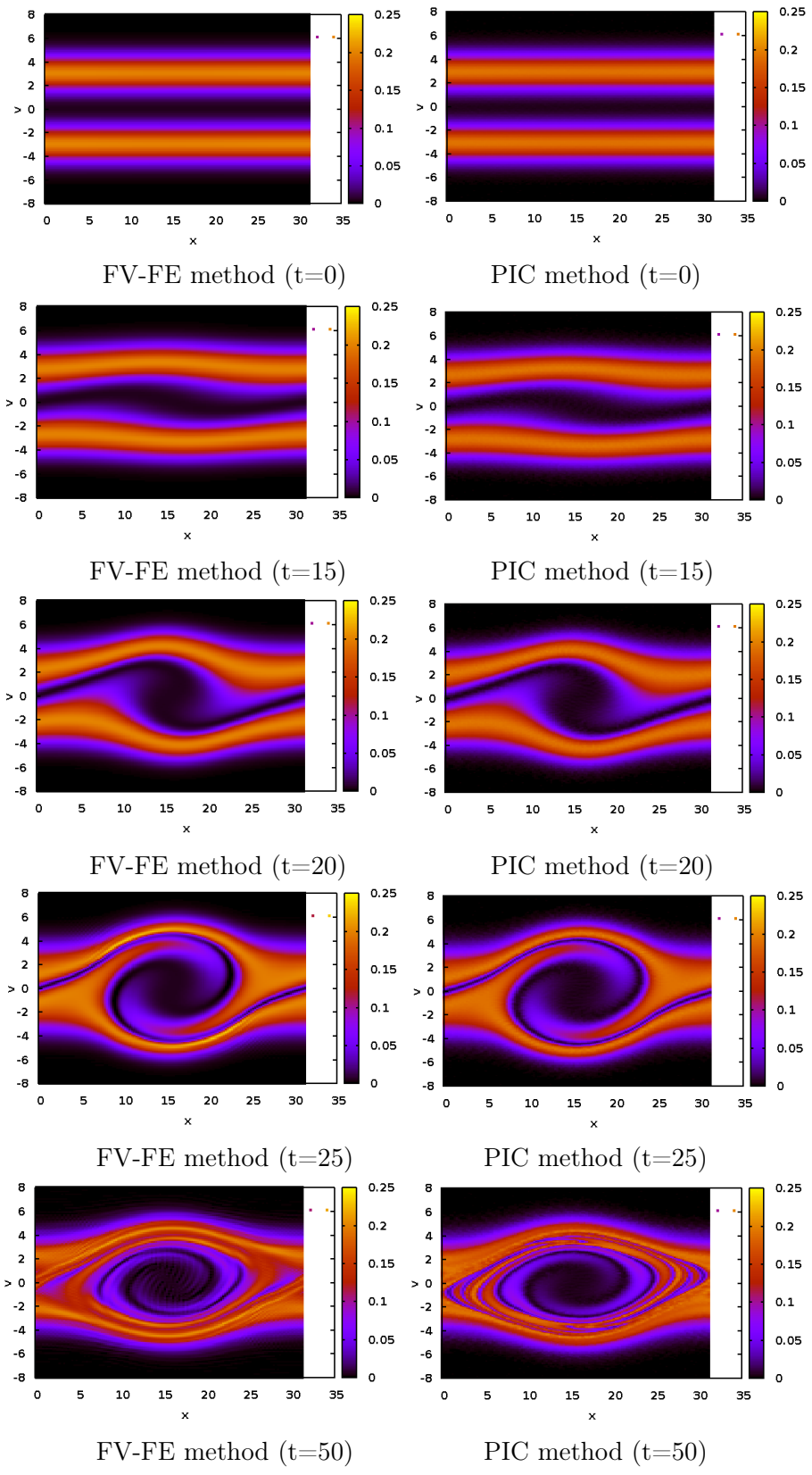
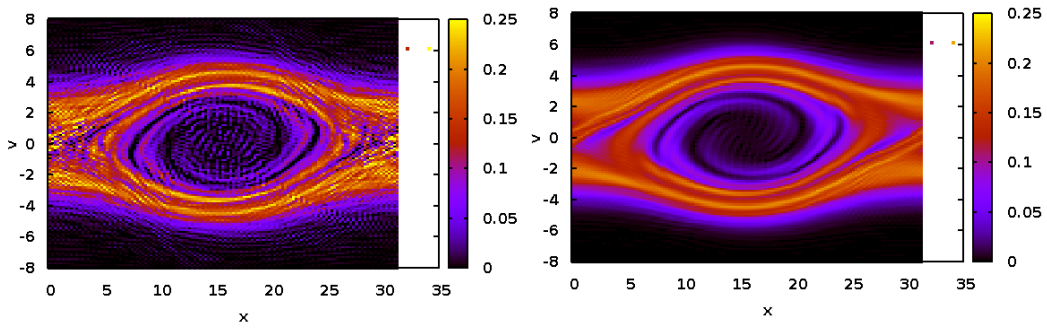


FIGURE 3.16 : The distribution function of the two-stream test case. Left : FV-FE method (RK3, viscous flux $\kappa = 0.005$, d $= 2, M = 64, N_x = 128, \beta = 1, \Delta t = 0.006$). Right : the PIC method.



centered flux viscous flux ($\kappa = 0.005$)

FIGURE 3.17 : The distribution function of the two-stream test case at time $t = 50$ computed with the reduced Vlasov-Poisson method, RK3 scheme, d $= 2, M = 64, N_x = 128, \beta = 1, \Delta t = 0.006$. Left : with the centered flux. Right : with the viscous flux ($\kappa = 0.005$).

3.6.4 Computation time

We also compare the computation time of the reduced Vlasov-Poisson method with the one of the PIC method. We consider the following data : $N_x = 256$, $V = 6$, $T_{\max} = 20$ the CFL $\wp = 0.6$ and the parameters $k = 0.2$ and $\varepsilon = 5 \times 10^{-2}$ for the landau damping test case. The number of time iteration is 1630. The reduced Vlasov-Poisson computation lasts $13,965$ seconds while the PIC computation, with approximately 130,000 particles, lasts 18 seconds. The graph of the electric energy is shown in Figure 3.10.

Chapitre 4

Hyperbolic approximation of the Fourier transformed Vlasov 1D equation

In this section, we consider a Fourier velocity transformation of the Vlasov equation. We then use the same method as in chapter 2 to construct a reduced model where the unknown depends on space and time instead of the full phase-space variables. The Fourier reduced model is a linear hyperbolic system, with non-linear source terms. We use the finite volume method to solve the new system.

As explained by Eliasson in [37], the reason for which we consider the Fourier transformed Vlasov-Poisson system instead of considering the original Vlasov-Poisson equation is due to the oscillations in velocity space (see section 1.4.1 on filamentation). Therefore, numerical simulations are polluted by aliasing effects (recurrence effect) when the oscillation wave length becomes too small to be captured by the mesh. In order to better control these large frequencies in velocity, we thus consider the Fourier transformed equation in the velocity direction.

4.1 Plasma mathematical model

We consider here the Vlasov 1D equation (7.1.6) with the initial condition and boundary condition which are defined in 3.1.1. For practical reasons, we will allow that the distribution function $f(x, v, t)$ and the electric field $E(x, t)$ take complex values, however, of course, only the real parts are physically relevant.

We consider a Fourier transformation with respect to the velocity variable (we note $I = \sqrt{-1}$)

$$\phi(x, \eta, t) = \int_{v=-\infty}^{+\infty} f(x, v, t) \exp(-I\eta v). \tag{4.1.1}$$

The Fourier velocity variable is denoted by $\eta \in \mathbb{R}$. The distribution function $\phi(x, \eta, t)$ satisfies the Fourier transformed Vlasov equation (see [37])

$$\partial_t \phi + I\partial_x \partial_\eta \phi + IE\eta\phi = 0. \tag{4.1.2}$$

In addition, the Poisson equation becomes

$$\partial_x E(x, t) = -1 + \phi(x, 0, t). \tag{4.1.3}$$

We call the new system 4.1.2 and 4.1.3 the Vlasov-Fourier equation. Several conserved quantities are associated to the Vlasov-Poisson system (see [37]). In particular, the total charge ρ, the total energy $\mathcal{E}$ and the L^2 norm of the distribution function are constant in time

$$\rho_{\text{tot}}(t) = \int f(x,v,t)dxdv = \int \phi(x,0,t)dx, \tag{4.1.4}$$
$$\mathcal{E}_{\text{tot}}(t) = \frac{1}{2}\int v^2 f(x,v,t)dxdv + \frac{1}{2}\int E(x,t)^2 dx = -\frac{1}{2}\int \partial_{\eta^2}\phi(x,0,t)dx + \frac{1}{2}\int E(x,t)^2 dx,$$
$$||\phi||_2^2(t) = \int f^2(x,v,t)dxdv = \int \phi^2(x,\eta,t)dxd\eta.$$

The Vlasov-Fourier representation (4.1.2)-(4.1.3) enables to have a better control of the high frequencies in velocity (see [37]). We then apply the reduction method described in chapter 2 for solving this system. We will see that we again obtain an hyperbolic approximation of the Vlasov equation.

4.2 Discretization of the Vlasov-Fourier equation with respect to the Fourier velocity variable

We will perform a semi-discretization of (4.1.2) with respect to the Fourier variable η in order to obtain a first order hyperbolic system set only in (x,t). We shall call this new system of equations the reduced Vlasov-Fourier model. We can expand the function ϕ on a basis of arbitrary functions depending on η. See for instance [67] and included references.

4.2.1 Continuous interpolation by the finite element method

In practice, we are not interested in the high frequencies oscillation in v. Therefore we shall assume that ϕ almost vanishes at the boundaries $\eta \to \pm\infty$. We consider thus a truncated domain $\eta \in [-\eta_{\max}, \eta_{\max}]$ and the following boundary conditions at $\pm\eta_{\max}$

$$I\partial_x\phi(x,\pm\eta_{\max},t) \mp \gamma\phi(x,\pm\eta_{\max},t) = 0. \tag{4.2.1}$$

We will that when $\gamma \geq 0$ then such conditions are dissipative. Other boundary conditions could be considered [37].

We suppose that the function $\phi(x,\eta,t)$ is well approximated by an expansion on the basis $\{\varphi_j\}_{j=1\cdots N_\eta}$ defined in 2.3.6.

$$\phi(x,\eta,t) = \sum_{j=1}^{N_\eta} w_j(x,t)\varphi_j(\eta), \tag{4.2.2}$$

Because of the interpolation property of the basis $\{\varphi_j\}_{j=1\cdots N_\eta}$

$$\varphi_i(N_j) = \delta_{ij}, \tag{4.2.3}$$

we have

$$\phi(x,N_i,t) = \sum_{j=1}^{N_\eta} w_j(x,t)\varphi_j(N_i) = w_i(x,t). \tag{4.2.4}$$

Therefore, we can approximate the initial condition in the following way

$$w_j(x,0) = \phi(x, N_j, 0) = \phi_0(x, N_j). \tag{4.2.5}$$

Considering equation (4.1.2) and boundary condition (4.2.1), we can consider the following weak formula of the problem : find $\phi(x,\eta,t)$ such that for all (continuous) test function $\varphi(\eta)$ we have

$$\int_\eta \partial_t\phi\varphi + \int_\eta I\partial_x\partial_\eta\phi\varphi + \int_\eta IE\eta\phi\varphi - \frac{1}{2}\varphi(\eta_{\max})I\partial_x\phi(\cdot,\eta_{\max},\cdot) + \frac{1}{2}\varphi(-\eta_{\max})I\partial_x\phi(\cdot,-\eta_{\max},\cdot) + \frac{1}{2}\varphi(\eta_{\max})\gamma\phi(\cdot,\eta_{\max},\cdot) + \frac{1}{2}\varphi(-\eta_{\max})\gamma\phi(\cdot,-\eta_{\max},\cdot) = 0. \tag{4.2.6}$$

This "semi-weak" formula is equivalent with the initial problem (4.1.2) supplemented with the boundary conditions (4.2.1). Indeed, if ϕ is a solution of (4.1.2) with the conditions (4.2.1), it is evident that ϕ is also a solution of (4.2.6). Reciprocally, if we suppose that ϕ is a solution of (4.2.6), because (4.2.6) is true for arbitrary test function. Thus for every function φ such that $\varphi(-\eta_{\max}) = \varphi(\eta_{\max}) = 0$, we obtain

$$\int_\eta \partial_t\phi\varphi + \int_\eta I\partial_x\partial_\eta\phi\varphi + \int_\eta IE\eta\phi\varphi = 0$$

and thus (4.1.2). Then taking test functions φ that do not vanish at $\eta = \eta_{\max}$ or $\eta = -\eta_{\max}$, we obtain the boundary conditions (4.2.1).

The factor $1/2$ in front of the boundary terms enables to ensure the hyperbolicity of the resulting equation. Indeed, we introduce the following matrices M, A and $B = B_E + D$ of dimension $N_\eta \times N_\eta$, whose coefficients are

$$\begin{aligned} M_{ij} &= \int_\eta \varphi_i\varphi_j \\ A_{ij} &= I\left[\int_\eta \varphi_i\varphi_j' - \frac{1}{2}\varphi_i(\eta_{\max})\varphi_j(\eta_{\max}) + \frac{1}{2}\varphi_i(-\eta_{\max})\varphi_j(-\eta_{\max})\right], \\ (B_E)_{ij} &= IE\int_\eta \eta\varphi_i\varphi_j, \quad D_{ij} = \frac{1}{2}\gamma[\varphi_i(\eta_{\max})\varphi_j(\eta_{\max}) + \varphi_i(-\eta_{\max})\varphi_j(-\eta_{\max})]. \end{aligned} \tag{4.2.7}$$

and we obtain the following equation

$$M\partial_t\mathbf{w} + A\partial_x\mathbf{w} + B\mathbf{w} = 0, \tag{4.2.8}$$

in which $\mathbf{w}(x,t)$ is the vector of N_η components

$$\mathbf{w} = (w_1, w_2, ..., w_{N_\eta})^T.$$

Obviously, the mass matrix M is positive hermitian. An integration by parts in η shows that A is hermitian (thanks to the factor $1/2$). Finally, B_E is skew-hermitian and D is diagonal non-negative. Then the system (4.2.8) is hyperbolic (i.e. that $M^{-1}A$ is diagonalizable with

real eigenvalues [85]) and energy dissipative (the proof is similar to the case of real distribution function, proposition 2.4.8 in chapter 3). Indeed, the L^2 norm of the distribution function is dissipated

$$\begin{aligned}\frac{d}{dt}||\phi||_2^2 &= -\int_x \mathbf{w}(x,t)^T D\mathbf{w}(x,t)dx \\ &= -\frac{\gamma}{2}\sum_{i=1}^{P}\left[\varphi_i^2(\eta_{\max})+\varphi_i^2(-\eta_{\max})\right]\int_x w_i^2(x,t)dx, \end{aligned} \tag{4.2.9}$$

where

$$||\phi||_2^2 = \int_x \mathbf{w}(x,t)^T M\mathbf{w}(x,t)dx = \int_{x,\eta\in[-\eta_{\max},\eta_{\max}]} \phi^2(x,\eta,t)dxd\eta.$$

In practice, to compute the matrices M, A, B we use the Gauss-Legendre integration and sparse matrix representations. The same computations as in chapter 3.

4.3 Finite volume schemes

As in the chapter 3, we apply finite volume scheme for the reduced Vlasov-Fourier equation. The space step is $\triangle x = L/N_x$ and the center of the cell C_i is $x_i = (i-1/2)\triangle x$. We also consider a sequence of times t_n, $n \in \mathbb{N}$, such that $t_0 = 0$ and $t_n = n\triangle t$, where $\triangle t$ satisfies the following CFL condition

$$\triangle t = \alpha\frac{\triangle x\triangle\eta}{d\pi},\ 0<\alpha\leq 1.$$

We now explain how to get heuristically such CFL condition. Indeed, because of the Fourier transform 4.1.1

$$\phi(x,\eta,t) = \int_{v=-\infty}^{+\infty} f(x,v,t)\exp(-I\eta v).$$

the Nyquist–Shannon sampling theorem requires that the sampling frequency is at least two times the maximum frequency of the signal. The sampling interval is $\Delta\eta$, thus the sampling frequency is $1/\Delta\eta$. The frequency of the signal is $v/2\pi$ and the maximum frequency is $v/2\pi$. We have then

$$\frac{1}{\Delta\eta} \geq 2\frac{V}{2\pi}$$

or

$$V \leq \frac{\pi}{\Delta\eta}$$

So, if we take

$$\Delta t = \alpha\frac{\triangle x\triangle\eta}{d\pi},\ 0<\alpha\leq 1.$$

the classical CFL condition is ensured (namely $V\Delta t \leq \Delta x$).

We consider a finite volume approximation of (4.2.8). We denote by $W_i(t)$ a piecewise constant approximation of W in each cell

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

We obtain the following semi-discrete (in space) approximation

$$M\partial_t \mathbf{w}_i = -\frac{F(\mathbf{w}_i, \mathbf{w}_{i+1}) - F(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\triangle x} - B\mathbf{w}_i.$$

where $(\mathbf{w}_L, \mathbf{w}_R) \mapsto F(\mathbf{w}_L, \mathbf{w}_R)$ denotes the numerical flux.

We then introduce a time discretization to compute $\mathbf{w}_i^n$

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

We use a time second order scheme given by the following algorithm

$$\begin{aligned}
\frac{\mathbf{w}_i^{n+1/2} - \mathbf{w}_i^n}{\triangle t/2} &= -\frac{F(\mathbf{w}_i^n, \mathbf{w}_{i+1}^n) - F(\mathbf{w}_{i-1}^n, \mathbf{w}_i^n)}{\triangle x} - M^{-1}B\mathbf{w}_i^n, \\
\frac{\mathbf{w}_i^{n+1} - \mathbf{w}_i^n}{\triangle t} &= -\frac{F(\mathbf{w}_i^{n+1/2}, \mathbf{w}_{i+1}^{n+1/2}) - F(\mathbf{w}_{i-1}^{n+1/2}, \mathbf{w}_i^{n+1/2})}{\triangle x} - M^{-1}B\mathbf{w}_i^{n+1/2}.
\end{aligned} \tag{4.3.1}$$

We consider several choices for the numerical flux $F(\mathbf{w}_L, \mathbf{w}_R)$. We consider the centered flux or a numerical flux with small numerical viscosity ("slightly upwinded flux"). The centered flux is given by

$$F(\mathbf{w}_L, \mathbf{w}_R) = M^{-1}A\frac{\mathbf{w}_L + \mathbf{w}_R}{2},$$

and the slightly upwinded flux

$$F(\mathbf{w}_L, \mathbf{w}_R) = M^{-1}A\frac{\mathbf{w}_L + \mathbf{w}_R}{2} - \frac{\delta}{2}(\mathbf{w}_R - \mathbf{w}_L). \tag{4.3.2}$$

where $\delta > 0$ is the numerical viscosity coefficient.

4.4 Test cases

We apply our numerical scheme for two test cases : the Landau damping and the two stream instability. We will compute the charge ρ, the total energy $\mathcal{E}$ and the L^2 norm of the distribution function in the computational domain $[0, L] \times [-\eta_{\max}, \eta_{\max}]$ see equation (4.1.4)). Only the last quantity is exactly conserved if $\gamma = 0$, (see equation (4.2.9). We will also compute the L^2 norm of the electric field. We are also interested in the distribution function in physical variable (x, v). The inverse Fourier transform reads

$$f(x, v, t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \phi(x, \eta, t)\exp(I\eta v)d\eta \quad \simeq \quad \frac{1}{2\pi}\int_{-\eta_{\max}}^{\eta_{\max}} \phi(x, \eta, t)\exp(I\eta v)d\eta. \tag{4.4.1}$$

We apply the rectangle method with oversampling for computing (4.4.1), in order to avoid Shannon aliasing. For this computation we can use a naive DFT computation instead of the FFT algorithm, because this step is applied only at the beginning and the end of the simulation.

In our numerical experiments, the discretization parameters are $N = 40$, $d = 5$, $N_x = 128$.

4.4.1 The Landau damping

The initial distribution function in the velocity Fourier variable writes

$$\phi(x,\eta,0) = \phi_0(x,\eta) = \int_{-\infty}^{+\infty} f_0(x,v)e^{-I\eta v}dv = (1+\varepsilon\cos(kx))e^{-\frac{\eta^2}{2}}. \tag{4.4.2}$$

We can estimate the time at which occurs the first recurrence in this case. Indeed, Landau-damping is a case with weak electric field. The Vlasov equation is thus almost the transport equation

$$\partial_t f + v\,\partial_x f = 0,$$

The method of charateristics states that the solution in this case is

$$\phi(x,\eta,t) = \phi_0(x-\eta t,\eta) = (1+\varepsilon\cos(kx-k\eta t))e^{-\frac{\eta^2}{2}}. \tag{4.4.3}$$

For each discretization point in $\eta_j = j\Delta\eta$, the function $\cos(kx - k\eta_j t)$ is periodic in time with period $2\pi/(kj\Delta\eta)$. In particular, the whole solution have period $2\pi/(k\Delta\eta)$, called the recurrence time. Let us consider the parameters $k = 0.5$ and $\varepsilon = 2\times10^{-4}$. In Figure 4.1, we compare the time evolution of the L^2-norm of the electric field obtained by our numerical scheme with the analytical solution. The scheme parameters are chosen as follows : $\delta = 0.01$ (slightly upwinded flux) and $\gamma = 5$ (dissipative η boundary condition). In that case, the theoretical recurrence time equals : $t = \pi N_\eta/(k\eta_{\max}) = 2\pi N_\eta/\eta_{\max} \approx 31,4$. The two curves coincide up the recurrence time and then the recurrence phenomena occurs (when the solution reaches the boundary in the η direction, (see [37]). In Figure 4.2, we observe that the electron charge, the total energy and the squared L^2 norm are approximately conserved up to the recurrence time.

4.4.2 Two-stream instability

The initial distribution function ϕ_0 in the Vlasov-Fourier case is given by

$$\phi_0(x,\eta) = (1+\varepsilon\cos(kx))e^{-\frac{1}{2}\eta^2}\cos(\eta v_0). \tag{4.4.4}$$

The value of parameters for this test case are $k = 0.2$, $\varepsilon = 5\times10^{-3}$ and $v_0 = 3$. The distribution function is plotted at times $t = 25$ and $t = 50$ in Figures 4.3 and 4.4. We compare the Vlasov-Fourier method with the slightly upwinded flux ($\delta = 0.005$ and $\gamma = 0$) and the Particle-In-Cell (PIC) method. At time $t = 50$ (Figure 4.4), the two methods capture the filamentation of the solution but they are slightly different. Increasing the numerical dissipation when taking $\delta = 0.05$ (Figure 4.4 bottom left), the oscillations are smoothed in the whole domain. Using the boundary dissipation in the η direction when taking $\gamma = 5$ (Figure 4.4 bottom right), the oscillations on the edges of the distribution support are removed : they indeed resulted from the L^2 norm conservation.

The Vlasov-Fourier method with the centered flux is actually unstable. In Figure 4.5, the real part of the distribution function is displayed. As time increases, oscillations propagate from the $\eta = 0$ line to the boundaries $\eta = \pm\eta_{\max}$. For the centered flux, unphysical oscillations develop. In Figure 4.6, we can observe that the scheme remains unstable even when using the dissipative boundary conditions in the η direction.

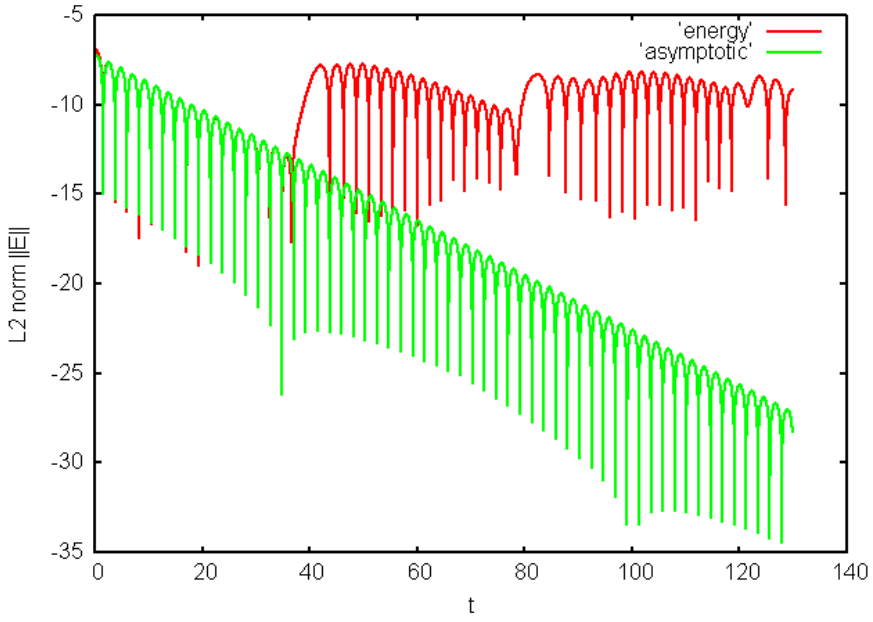


FIGURE 4.1 : Landau damping test case : L^2 norm of the electric field $||E||_2$ as a function of time. The green curve "asymptotic" is the analytical solution and the red curve "energy" is computed with the Vlasov-Fourier method with the slightly upwinded flux ($\delta = 0.01$, $\gamma = 5$).

4.5 Conclusion

The numerical results show the stability of numerical method under a CFL condition for RK3 ou RK4 with centered flux. We still observe the oscillation phenomenon which is due to the boundary condition. It would be interesting to test the "outflow boundary condition" mentioned in [37] and compare it with the boundary condition that we used in this chapter. In the future, it would also be interesting to apply approximate transparent boundary conditions approaches for neglecting in a finer way the velocity high frequencies. We also plan to apply this method in the two-dimensional case.

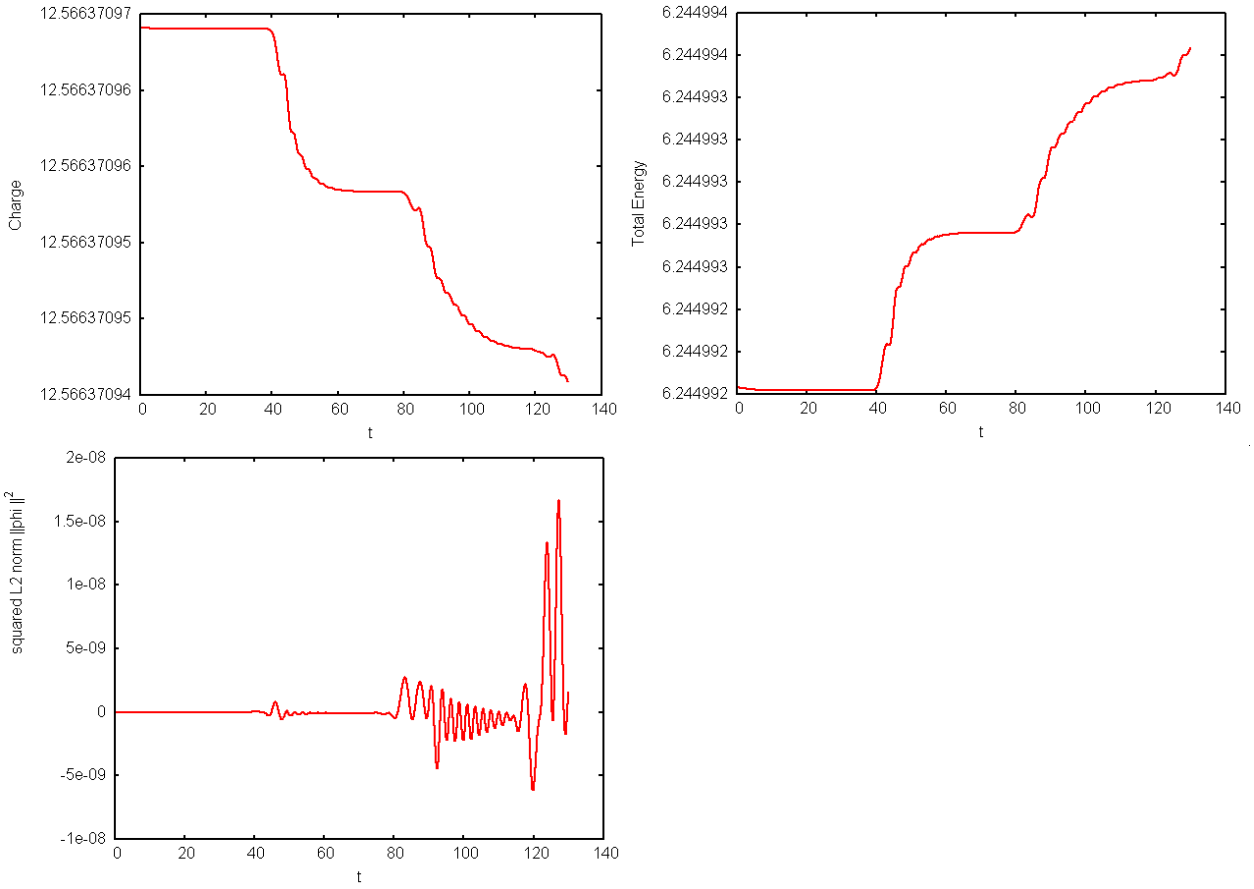
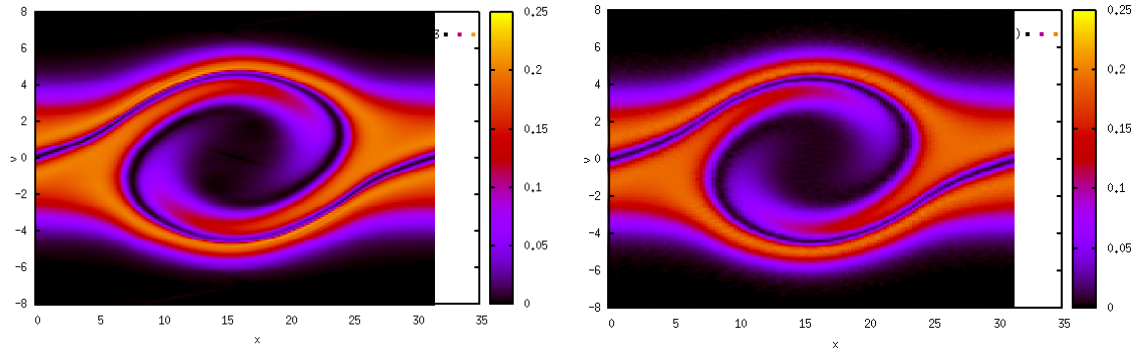


FIGURE 4.2 : Landau damping test case : charge ρ (top left), total energy $\mathcal{E}$ (top right) and squared L^2 norm $||\phi||_2^2$ (bottom) as functions of time - computed with the reduced Vlasov-Fourier method with the slightly upwinded flux ($\delta = 0.01$, $\gamma = 5$).



Vlasov-Fourier method ($\delta = 0.005$, $\gamma = 0$) PIC method

FIGURE 4.3 : Two-stream instability test case : distribution function $f(x, v, t)$ at time $t = 25$ in the (x, v) phase space. Left : Vlasov-Fourier method with the slightly upwinded flux ($\delta = 0.005$). Right : the PIC method.

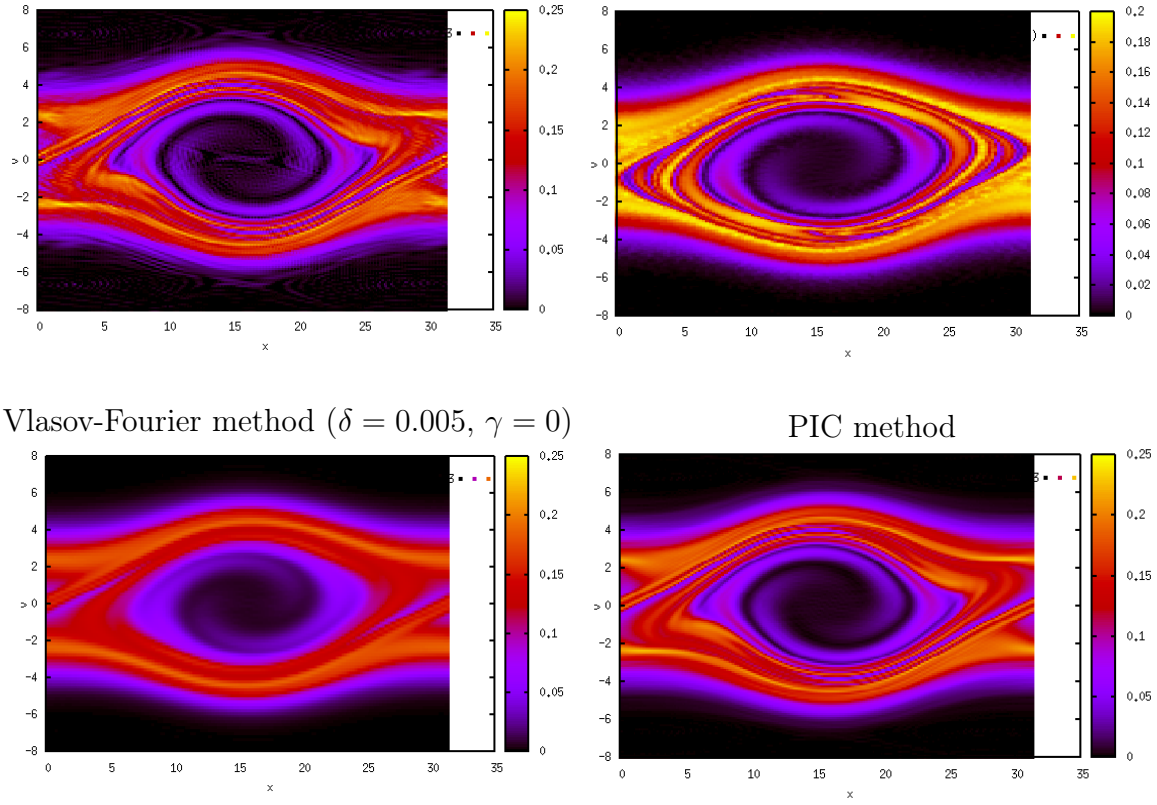
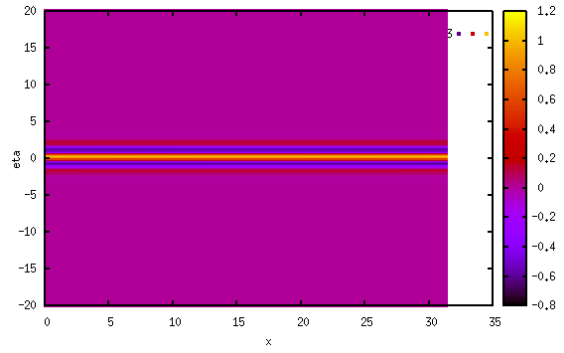
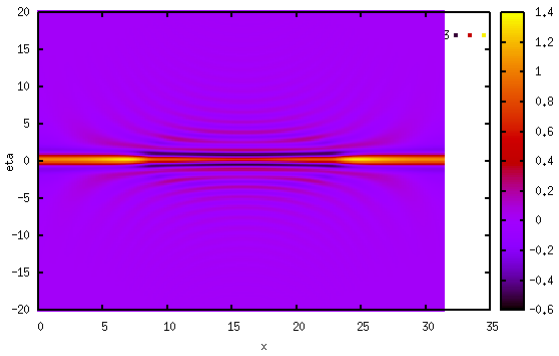


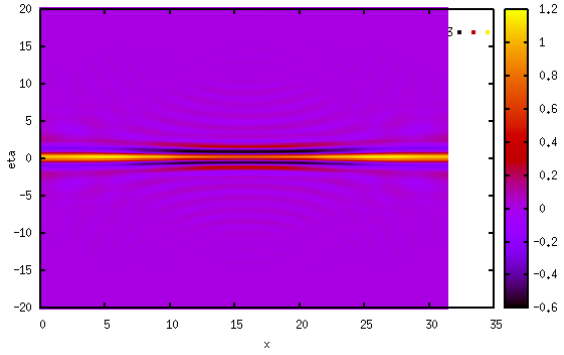
FIGURE 4.4 : Two-stream instability test case : distribution function $f(x, v, t)$ at time $t = 50$ in the (x, v) phase space. Comparison of the Vlasov-Fourier method with the slightly upwinded flux with the PIC method.



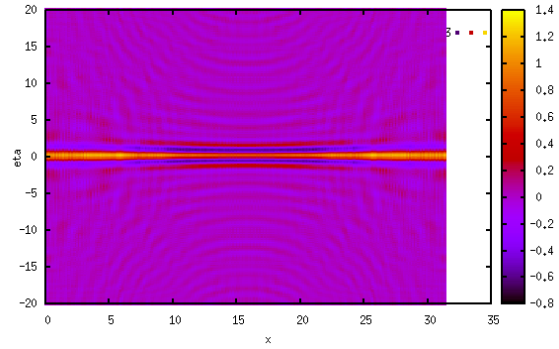
slightly upwinded flux : $t = 0$



slightly upwinded flux : $t = 25$

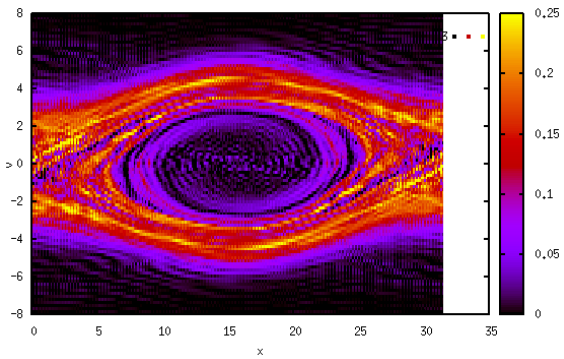


slightly upwinded flux : $t = 50$

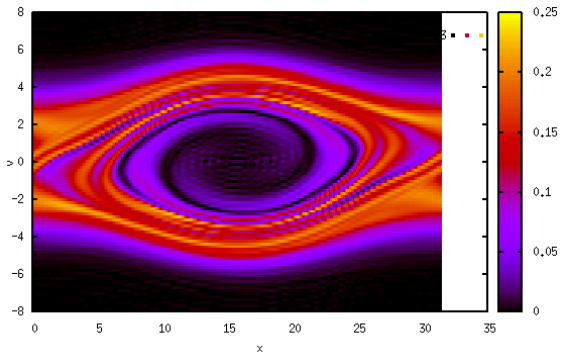


centered flux : $t = 50$

FIGURE 4.5 : Two-stream instability test case : real part of the Fourier-transformed distribution function $\Re(\phi(x, \eta, t))$ in the (x, η) phase space. Comparison of the slightly upwinded flux ($\delta = 0.05$) with the centered flux for the reduced Vlasov-Fourier method.



centered flux ($\delta = 0$, $\gamma = 0$)



centered flux ($\delta = 0$, $\gamma = 5$)

FIGURE 4.6 : Two-stream instability test case : distribution function $f(x, v, t)$ at time $t = 50$ in the (x, v) phase space. The Vlasov-Fourier method using the centered flux with and without dissipative boundary conditions.

Troisième partie

Mise en oeuvre en dimension 2 et parallélisation

Chapitre 5

Numerical simulation of the reduced 2D Vlasov-Poisson system

In chapters 3 and 4, we have studied the Vlasov equation in the one-dimensional case.

In this chapter, we apply our approach to the two-dimensional Vlasov-Poisson system. In two dimensions, there are more variables than in the one-dimensional, so the computations are more expensive in time and memory . In order to save the computation time and memory, we apply a parallelization to our algorithms : we implement a distributed-memory parallel version of the algorithm with MPI library. We also use the `SELALIB` library (Modular library for the kinetic and gyrokinetic simulation of tokamak plasmas by the semi-lagrangian method) (see for instance [3]). The approach could be generalized to higher dimensions, relativistic cases and Vlasov-Maxwell systems.

We consider the reduced Vlasov equation (2.4.3) of chapter 2. The main ingredients for the numerical discretization are the finite volume discretization in space and the Runge-Kutta method for the time discretization.

We consider two different types of discretization points and integration points in velocity and we compare the results obtained from the two cases. Namely, one is based on a uniform mesh in velocity with Gauss-Legendre integration points (as seen in chapters 3 and 4) and another is constructed with the Gauss-Lobatto points used for both discretization and integration points (mass lumping). We show that the two methods have different behavior as regards the recurrence phenomenon (in the Landau test-case) : for the Gauss-Lobatto points, the recurrence time is indenpendent of the degree of the polynomial basis and depends only on the number of elements while for the uniform points it depends on the two parameters.

5.1 Finite volume approximation in space

5.1.1 Spatial mesh

We consider a finite volume approximation. We assume that the spatial domain $]0, L[\times]0, L[$ is split into $N_{x_1} \times N_{x_2}$ cells. The space steps are defined by $\Delta x_1 = L/N_{x_1}$ and $\Delta x_2 = L/N_{x_2}$. The cell C_{kl} is the set $](k-1)\Delta x_1, k\Delta x_1[\times](l-1)\Delta x_2, l\Delta x_2[$ for $k = 1 \dots N_{x_1}$ and $l = 1 \dots N_{x_2}$. For practical reasons, we also consider additional cell rows and columns at $k = 0$, $k = N_{x_1}+1$, $l = 0$, $l = N_{x_2}+1$ for applying the periodic boundary condition : at the beginning

of each time step, we copy the data from the opposite sides to the additional cells in order to apply the boundary condition. The center of the cell C_{kl} is $x_{kl} = ((k - \frac{1}{2})\Delta x_1, (l - \frac{1}{2})\Delta x_2)$.

5.1.2 Piecewise constant approximation

We are looking for a piecewise constant approximation $(w_{kl}(t))_{k,l}$ of the vector $\mathbf{w}(x,t)$

$$w_{kl}(t) \simeq \mathbf{w}(\mathbf{x},t),\ \mathbf{x} \in C_{kl}. \tag{5.1.1}$$

Each w_{kl} is a vector of dimension $N_{\mathbf{v}}$. Since the j^{th} component of the initial condition is given by $(w(x,0))_j = f_0(x, N_j)$ for each $1 \leqslant j \leqslant N_{\mathbf{v}}$, the vectors are initialized as follows

$$(w_{kl}(0))_j = f_0(x_{kl}, N_j).$$

5.1.3 Finite volume scheme

We apply a finite volume approximation (semi-discrete in space) of equation (2.4.3) :

$$\begin{aligned} M\partial_t w_{kl} = &-\frac{F(w_{k,l}, w_{k+1,l}, \nu^1) - F(w_{k-1,l}, w_{k,l}, \nu^1)}{\Delta x_1} - \frac{F(w_{k,l}, w_{k,l+1}, \nu^2) - F(w_{k,l-1}, w_{k,l}, \nu^2)}{\Delta x_2} \\ &- B(E)w_{k,l}. \end{aligned} \tag{5.1.2}$$

where $F(w_a, w_b, \nu)$ denotes the numerical flux of the finite volume scheme in the spatial direction $\nu \in \mathbb{R}^2$. Our regular mesh implies that we only consider the directions $\nu^1 = (1,0)$ and $\nu^2 = (0,1)$. We here consider the following numerical flux

$$F(w_L, w_R, \nu) = A_i\nu_i\frac{w_L + w_R}{2} - \frac{\kappa}{2}(w_R - w_L),$$

with $\kappa \geq 0$. As described in the chapter 3, the case $\kappa = 0$ corresponds to the centered scheme while the case $\kappa > 0$ corresponds to an upwind scheme. The parameter $\kappa > 0$ is a numerical viscosity parameter that allows to add stabilization to the numerical scheme.

5.2 Time discretization

We finally consider a sequence of times t_n, $n \in \mathbb{N}$, such that $t_0 = 0$ and $t_n = n\triangle t$, where $\triangle t$ is the time step. We note $(w_{kl}^n)_{k,l}$ the time and space discretization of the vector $\mathbf{w}(\mathbf{x},t)$ at time t_n

$$w_{k,l}^n \simeq w_{k,l}(t_n), \quad \mathbf{x} \in C_{k,l}.$$

We use classical time integration solver. We consider either the second order Runge-Kutta scheme (RK2), the third order Runge-Kutta scheme (RK3) or the fourth-order Runge-Kutta scheme (RK4) (see (3.3.2), (3.3.3) and (3.3.4) from chapter 3). High-order schemes enable to obtain stability properties even for weakly upwind flux.

In order to ensure the stability of the schemes, the time step should satisfy a CFL condition. The CFL condition takes the following form

Proposition 5.2.1. *(CFL condition) Stability for (RKj) with $j = 2, 3$ or 4 is ensured if*

$$\bigcup_{\lambda_k \in Spec(\mathcal{M}^{-1}A)} Spec(-\lambda_k C + \kappa D) \cup Spec(\mathcal{M}^{-1}B(E)) =: \mathcal{R}(\Delta x, \Delta v) \subset \mathcal{S}_j(\Delta t), \qquad (5.2.1)$$

where $Spec(\mathcal{M})$ denotes the set of all the eigenvalues of the matrix $\mathcal{M}$, matrices C, D are respectively the centered and upwind matrices

$$C = \frac{1}{2\Delta x}\begin{pmatrix} 0 & -1 & 0 & & 0 & 1 \\ 1 & 0 & -1 & & 0 & 0 \\ 0 & 1 & 0 & & 0 & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & & 0 & -1 \\ -1 & 0 & 0 & & 1 & 0 \end{pmatrix}, \qquad D = \frac{1}{2\Delta x}\begin{pmatrix} -2 & 1 & 0 & & 0 & 1 \\ 1 & -2 & 1 & & 0 & 0 \\ 0 & 1 & -2 & & 0 & 0 \\ & & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & & -2 & 1 \\ 1 & 0 & 0 & & 1 & -2 \end{pmatrix}$$

and $\mathcal{S}_j$ is the Runge-Kutta stability region defined by

$$\mathcal{S}_j(\Delta t) = \left\{ \mu \in \mathbb{C} : \left\| 1 + \Delta t\, \mu + \frac{\Delta t^2}{2!}\mu^2 + \cdots + \frac{\Delta t^j}{j!}\mu^j \right\| \leq 1 \right\}$$

Thanks to the Gershgorin theorem, we can obtain upper bound of the different eigenvalues (see annex), but they are not optimal. In Fig.5.1, we have represented the eigenvalues (numerically computed) of the matrice $\mathcal{M}^{-1}B(\mathbf{E})$ where the integral are computed with the Gauss-Lobatto method and the stability domain $\mathcal{S}_4$ (related to Runge-Kutta 4 method). Due to the boundary term in (2.4.5), the eigenvalues have non zero real part. In that case, stability condition 5.2.1 implies that the imaginary parts of all elements λ of $\mathcal{R}(\Delta\mathbf{x}, \Delta\mathbf{v})$ should satisfy : $|\Im(\lambda)| \leq 2\sqrt{2}/\Delta t$. In practice, we consider more general CFL stability condition of the form :

$$\triangle t = \wp \min\left(\frac{\Delta x_1}{V_{\max}}, \frac{\Delta x_2}{V_{\max}}, \frac{\beta_d \Delta v_1}{\mathbf{E}_{\max}}, \frac{\beta_d \Delta v_2}{\mathbf{E}_{\max}} \right), \quad \text{with } 0 < \wp \leq 1. \qquad (5.2.2)$$

where $\mathbf{E}_{\max}$ is the infinite norm of the electric field $\mathbf{E}$ and $\beta_d > 0$ is a coefficient that depends on the eigenvalues of the matrix $B(\mathbf{E})$. Remark that as said in the proposition 3.5.8 the centered flux ($\kappa = 0$) is always unstable when used with the second order Runge-Kutta scheme.

5.3 Subdomain parallelism - MPI library

MPI (Message Passing Interface) is a library that helps the computations on distributed-memory parallel machines. Many processors are performing the task in parallel. They exchange messages in order to synchronize theirs results. The communications among processes, which have separate address spaces, may take time. It is thus important to minimize communications between the process. Because we solve a four dimensional PDE, the numerical computations can become very time and memory consuming. We have implemented a distributed-memory parallel version with the MPI library of our finite volume algorithm in the library Selalib, in order to perform computations within a reasonable time. We use a domain decomposition algorithm for the finite volume scheme. Domain decomposition methods

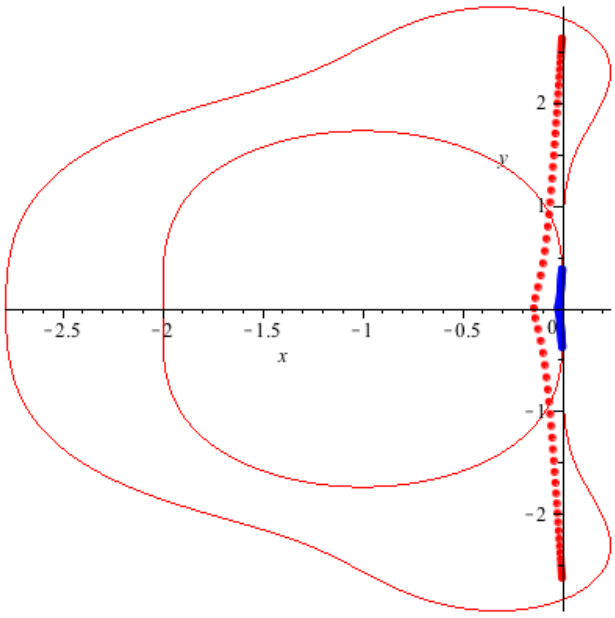


FIGURE 5.1 : Boundaries of the stability domains $\mathcal{S}_2$ and $\mathcal{S}_4$ and eigenvalues of the matrix $M^{-1}B(E)$ in the complex plane (computed with Gauss-Lobatto integration rule). Parameters : $d = 2$, $M = 30$, $V_{\max} = 6$, $E = 1$, $\Delta t = 0.05$ (RK2), $\Delta t = 0.35$ (RK4).

are well adapted to MPI paradigm because communications occur only at the boundaries of the subdomains. The volume of the transferred data is small compared to the computations performed in the volumes of the subdomains. The space domain $\Omega_x =]0, L[\times]0, L[$ is split into $q \cdot q$ subdomains along the x_1, x_2 direction. At the end of each step of the time integration algorithm each subdomain $s = 1 \cdots q^2$ exchanges its upper and lower rows and its rightmost and leftmost columns with its four neigbours. In other words this is an implementation of a domain decomposition method using ghost points. We use the periodic boundary condition in x direction.

For instance, consider a Cartesian mesh with $5\times5 = 25$ nodes, 5 in each direction x_1, x_2, as in Figure 5.2. We for instance parallelize the code in the x_2 direction : we thus consider slices in the x_2 direction and one processor is assigned to each slices. When we want to compute the numerical flux at one node (for instance node 6), part of the data are already available by the processor (data at nodes $5, 7, 1$) but, for nodes at the boundary of the subdomain, it requires data computed by the other processor (data at node 11). These data are transferred from one processor to the other thanks to the "point to point communication" module in the library `SELALIB`. With the subdomain decomposition by slices, we only need to assigned two ports to each processor, one port used for the communication with the processor up above and one port for the communication with processor down below.

5.4 Test cases

In this section, we perform several test-cases to study the accuracy and the performance of the method. We study all the one-dimensional test-cases mentioned in chapter 3. On classical one-dimensional test-cases (transport, Landau damping), we first establish the numerical

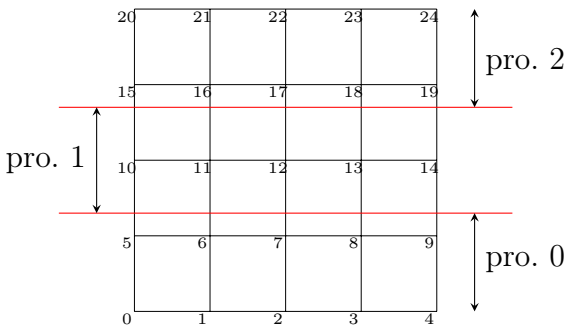


FIGURE 5.2 : MPI process distribution

accuracy of the method. We then focus on the numerical diffusion of the method. We finally discuss the parallel scalability performance on two dimensional test-case. All these test-cases are periodic in space. Therefore, taking advantage of the cartesian mesh, the Poisson equation is solved by using the FFT algorithm : this requires only O(N^2 logN) operations (with $N = \max(N_{x_1}, N_{x_2})$) in 2D.

Convergence properties of the schemes are presented only for the uniform discretization points in velocity. We obtain similar results for the Gauss-Lobatto points.

5.4.1 One dimensional test cases

Numerical Convergence Rates (1D)

In order to study the order of convergence of this method in two dimensions, we use the 1D-velocity and space transport equation. Details of the two test cases are given in Chapter 3. We obtain the same result as those of Chapter 3, which ensures the correctness of the order of convergence. For example, for the velocity transport equation (see Fig.5.3), we report the error between the exact and the numerical solutions with respect to $\Delta \mathbf{v} = 2V/M$. The degree of the finite element basis in velocity is taken equal to d = 3. We take a small value $\wp = 0.01$ in the CFL condition (see 5.2.2) in order to reduce the time integration error. We can observe that the line has slope 3.3, which almost matches the theoretical $O(\Delta \mathbf{v}^3)$ order of convergence (see proposition 2.5.6). Note that for bigger value of Δt, we recover the order 2 convergence (which is now limited by the time integration accuracy).

Landau damping $1D$ test cases

We use the test cases : Landau damping of section 3.6.2 and two-stream instability of section 3.6.3 (in chapter 3) in order to verify the 2D code.

For example, we test the 1D strong Landau damping test case. The test is given in 3.6.3 with the parameter $k = \varepsilon = 0.5$. The number of cells $N_{x_1} = 32$ in the x_1-direction, and $2 \cdot 32 + 1 = 65$ in the v_1-direction (namely the degree d = 2 and the number of element $M = 32$). We plot the mass, the L^1 and L^2 norm of the distribution function in order to verify the conservation. We compare two methods : the Runge-Kutta of order 2 scheme with the viscous flux and the Runge-Kutta order 4 scheme with the centered flux. The time step is 7.10^{-3} with the RK4 scheme and 2.10^{-3} for the RK2 approach. We present the results

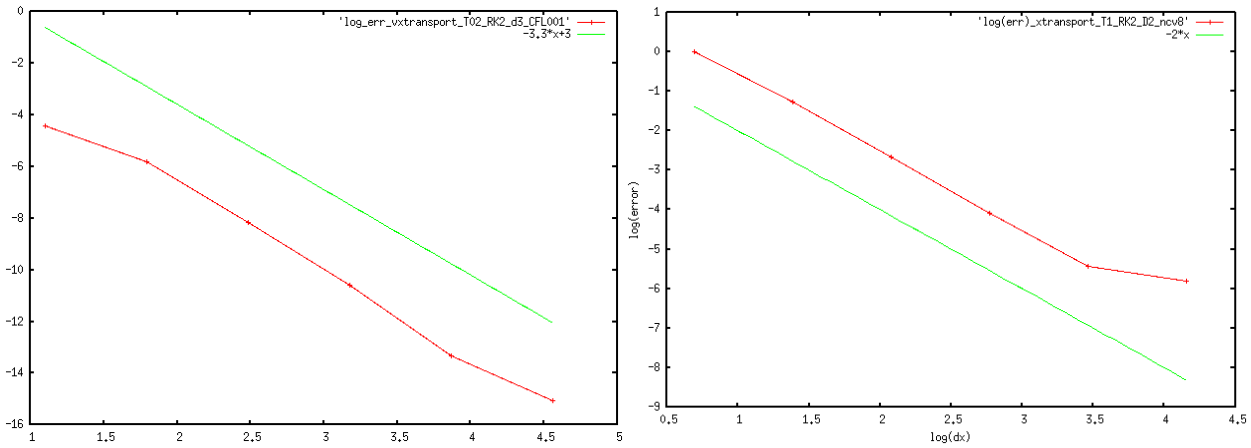


FIGURE 5.3 : (Transport test-cases) Left : L^2 error between exact and numerical solution of the distribution function with respect to Δv in log scale. Parameters : d $= 3$, $V = 6$, $\wp = 0.01$, $T = 0.2$. Right : L^2 error between exact and numerical solution with respect to Δx in log scale. Parameters : $V = 1$, $\wp = 0.01$, $T = 1$, d $= 2$, $M = 8$.

obtained for both uniform discretization points (left) and the Gauss-Lobatto discretization (right) in Figures 5.4, 5.5 and 5.6. We can make the same remarks as in section 3.6.2 (for the 1D code) : the viscous flux enables to better preserve the positivity of the distribution function and thus leads to better conservation of L^1 norm (see Fig. 5.4). However, the L^2 norm decreases in time with the viscous flux while remaining constant for the centered flux (see Fig. 5.6). When using the Gauss-Lobatto points, we obtain intermediate results as regards conservation properties : the L^2 dissipation is less important for the RK2-viscous flux and the mass is better conserved when using the RK4-centered flux.

We also plot the space integrated distribution function in this test case (Fig. 5.7 and Fig. 5.8) obtained with uniform discretization points in velocity (we obtain similar results with the Gauss-Lobatto discretization points). In Fig. 5.7, we plot the space integrated distribution function at time $10, 20, 30, 40, 50, 60, 70, 80$ with the Runge-Kutta 2 scheme using slightly upwinded flux, $\Delta t = 2.5 \times 10^{-3}$. We plot the same thing in Fig.5.8 but for the RK2 scheme with centered flux. With the centered flux (Fig.5.8), we observe more oscillations than with the slightly upwinded flux (Fig.5.7). It is due to the fact that the RK2 with centered flux is unstable, while the RK2 with the viscous flux can be made stable with an adequate time step (see proposition 3.5.10 in chapter 3).

5.4.2 Landau damping (2D)

Landau damping 1

In this test case, the initial condition is set to

$$f_0(\mathbf{x}, \mathbf{v}) = \frac{1}{2\pi}\Big(1 + \varepsilon \cos(k_1 x_1)\cos(k_2 x_2)\Big)\, e^{-\frac{(v_1^2+v_2^2)}{2}},$$

where $\varepsilon = 5.10^{-3}$ and the wave numbers are $k_1 = k_2 = 0.5$. The length of the periodic box in the physical space is $L_1 = L_2 = 4\pi$.

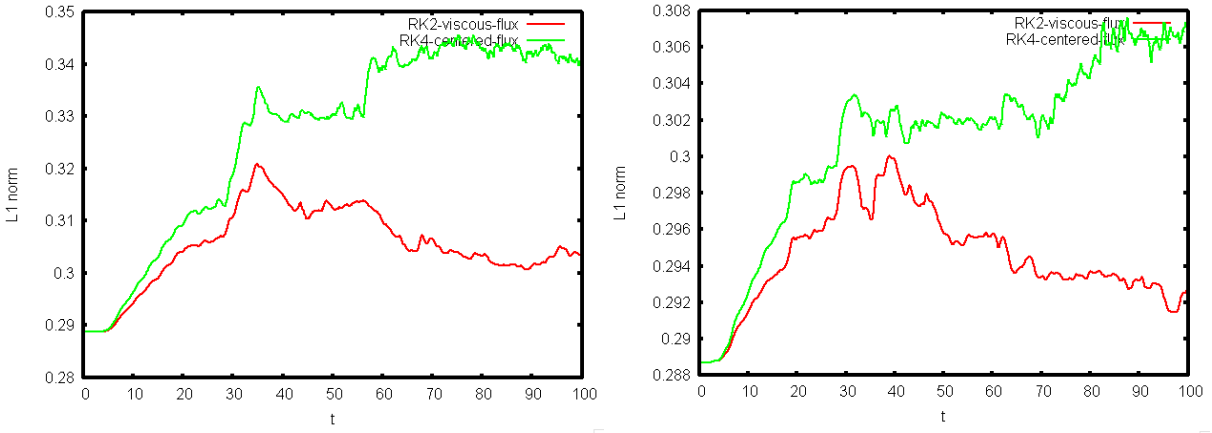


FIGURE 5.4 : Strong landau damping test case : the L^1 norm of the distribution function as functions of time. Comparison between the RK2 scheme upwinded flux ($\kappa = 0.005$, $\Delta t = 2 \times 10^{-3}$) and with the RK4 scheme centered flux ($\kappa = 0$, $\Delta t = 5 \times 10^{-2}$). Numerical paramaters : $V_{\max} = 6$, $N_{x_1} = 64$, $N_{v_1} = 65$ dof. Left : uniform discretization points, right : Gauss-Lobatto discretization points (in velocity).

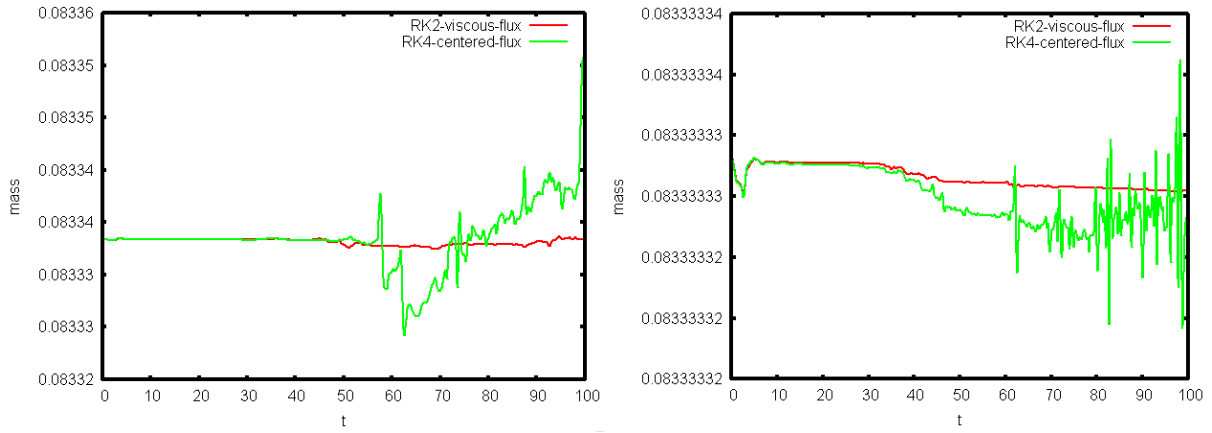


FIGURE 5.5 : Strong landau damping test case : the mass of the distribution function as functions of time. Comparison between the RK2 scheme upwinded flux ($\kappa = 0.005$, $\Delta t = 2 \times 10^{-3}$) and with the RK4 scheme centered flux ($\kappa = 0$, $\Delta t = 5 \times 10^{-2}$). Numerical paramaters : $V_{\max} = 6$, $N_{x_1} = 64$, $N_{v_1} = 65$ dof. Left : uniform discretization points, right : Gauss-Lobatto discretization points (in velocity).

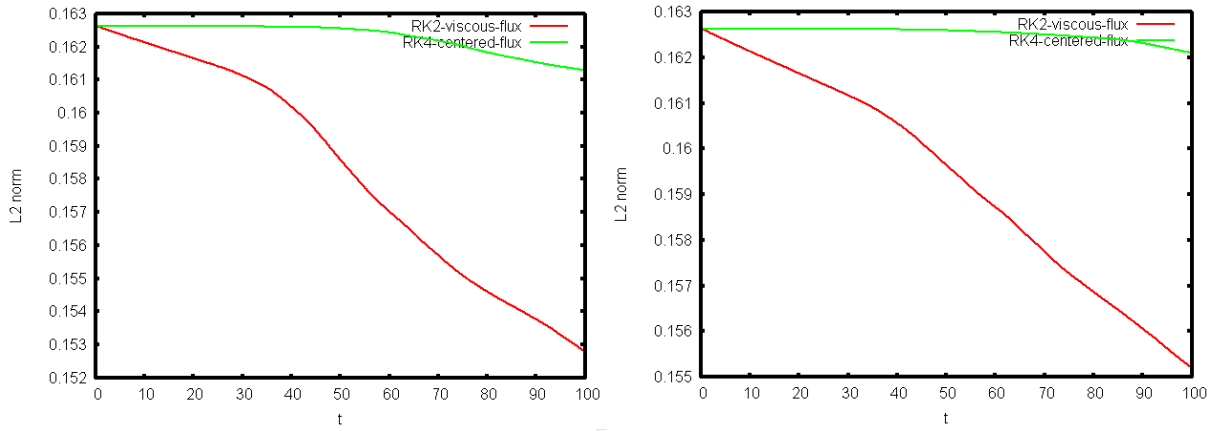


FIGURE 5.6 : Strong landau damping test case : The L^2 norm of the distribution function as functions of time. Comparison between the RK2 scheme upwinded flux ($\kappa = 0.005$, $\Delta t = 2 \times 10^{-3}$) and with the RK4 scheme centered flux ($\kappa = 0$, $\Delta t = 5 \times 10^{-2}$). Numerical paramaters : $V_{\max} = 6$, $N_{x_1} = 64$, $N_{v_1} = 65$ dof. Left : uniform discretization points, right : Gauss-Lobatto discretization points (in velocity).

In Fig. 5.9, we report the evolution of the electric energy obtained by this method. The number of cells in each direction are $N_{x_1} = N_{x_2} = 32$, the degree is set to 2 and the number of elements in v_1 and v_2 is $M = 32$ (therefore, there are $2.32 + 1 = 65$ cells in v_1 and v_2 direction). The damping rate here is 0.394 which is predicted by theory.

Landau damping 2 - Comparison between uniform and Gauss-Lobatto discretization points in velocity

We can also consider the following initial condition

$$f_0(x, v) = \frac{1}{2\pi}(1 + \varepsilon \cos(k_1 x_1 + k_2 x_2))e^{-\frac{(v_1^2+v_2^2)}{2}}.$$

The choice of parameters is : $\varepsilon = 2.10^{-4}$ and the wave numbers are $k_1 = k_2 = 0.5/\sqrt{2}$, the initial function can be written as

$$f_0(x, v) = \frac{1}{2\pi}(1 + \varepsilon \cos(k(x_1 + x_2))e^{-\frac{(v_1^2+v_2^2)}{2}}.$$

The length of the periodic box in the physical space is $L_1 = L_2 = 4\pi\sqrt{2}$, which is chosen to be large enough in order to contain one wave length. The velocity domain is $[-6, 6]$.

In Figures 5.10 and 5.11, we show that the recurrence time (due to aliasing) increases as soon as we increase the number of elements or the degree of polynomials in velocity in the case of **uniform discretization points in velocity**. In Figure 5.10, we set the degree of polynomials basis equal to d = 2 and we make vary the number of elements : we can see the recurrence phenomenon in the electric field occurs at about $t = 47.4$ for the case of 16 elements in velocity and at about $t = 94.8$ for the case of 32 elements in velocity. In Figure 5.11, we set the number of elements and make vary the degree of polynomials : the recurrence time equals $t = 47.4$ for d = 2 and $t = 64$ for d = 4.

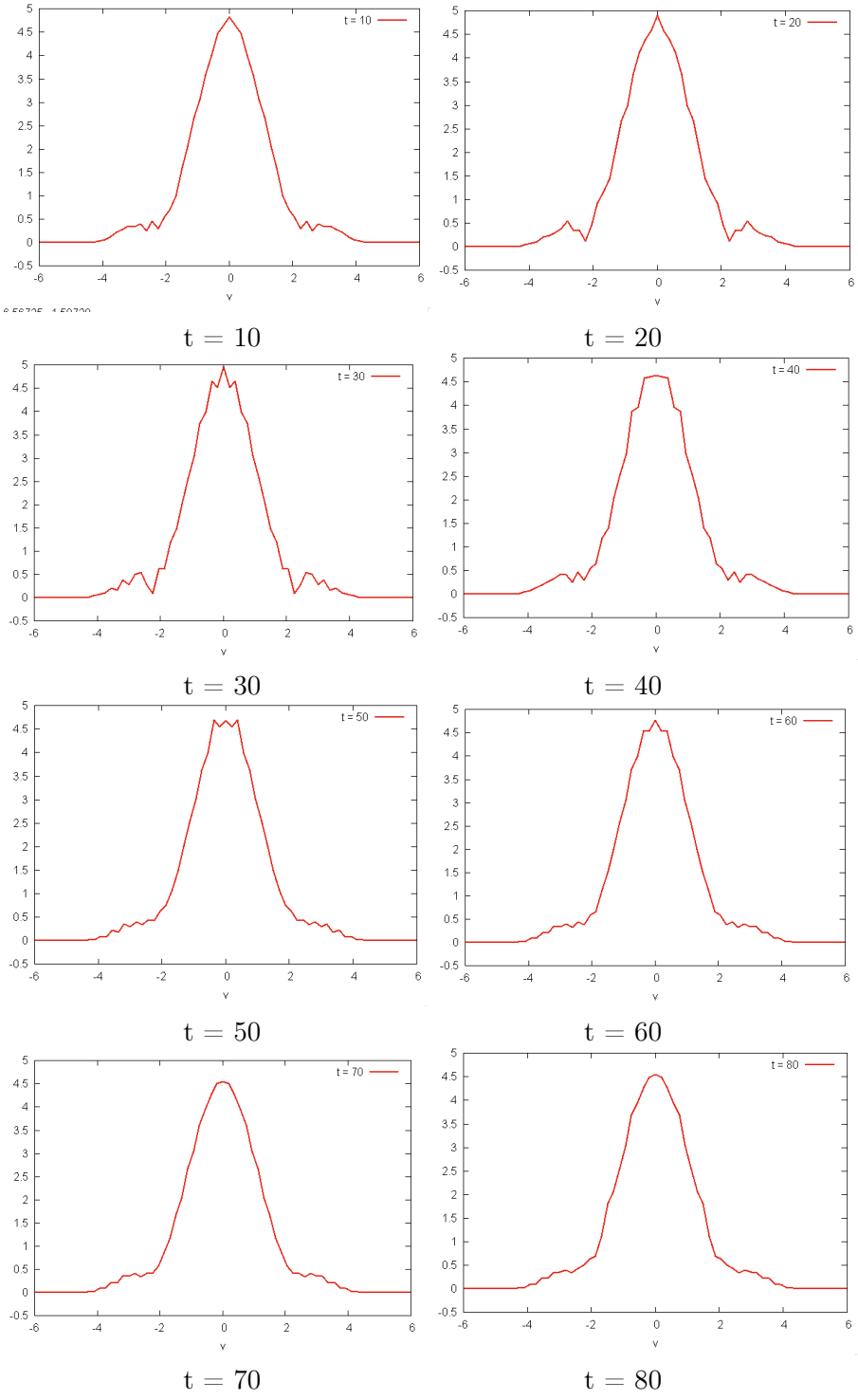


FIGURE 5.7 : Strong landau damping test case : the space integrated distribution function with RK2 scheme, viscous flux ($\kappa = 0.001$, $\Delta t = 2.5 \times 10^{-3}$) . Numerical paramaters : $V_{\max} = 6$, $N_{x_1} = 64$, $N_{v_1} = 65$ dof. Uniform discretization in velocity.

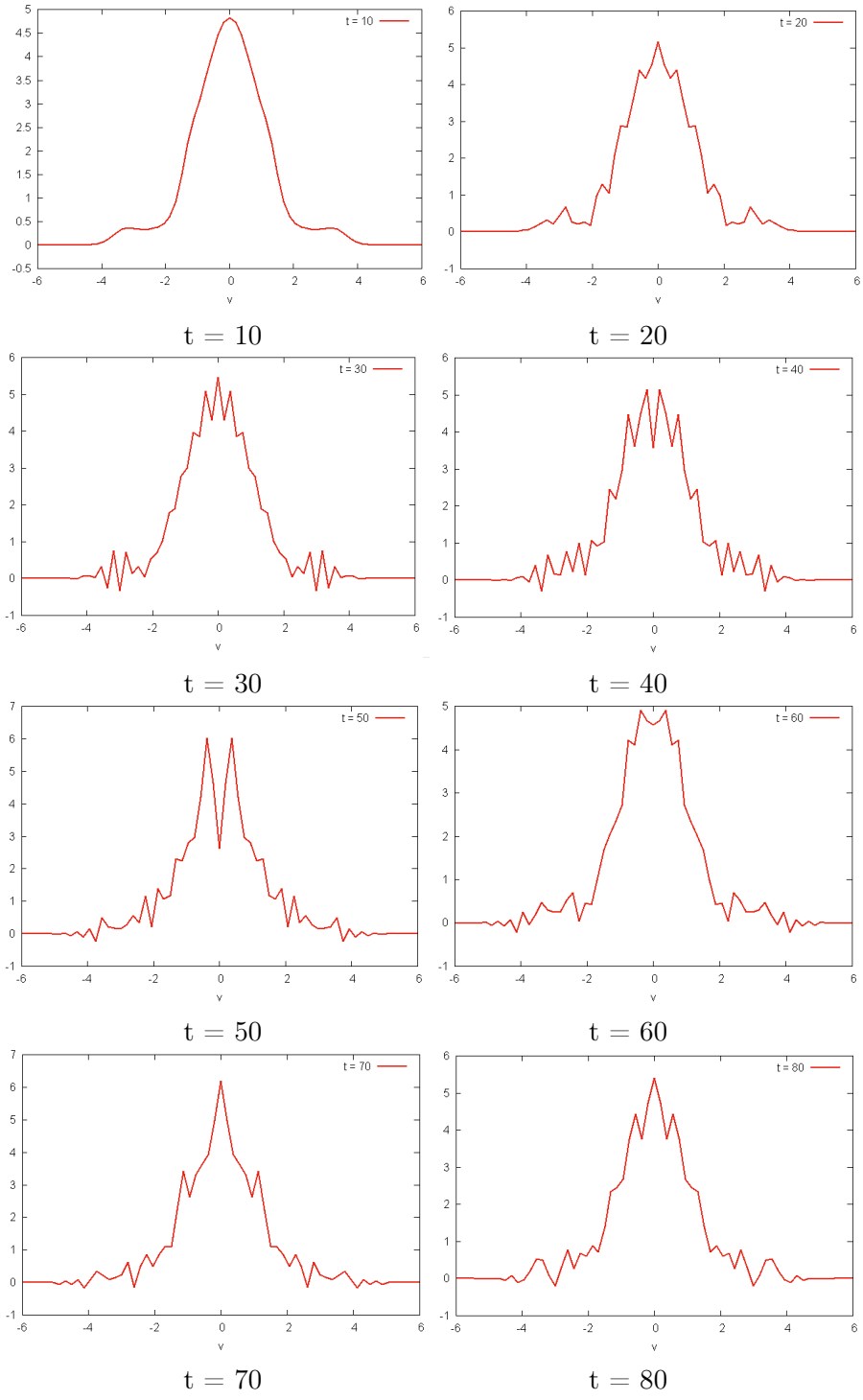


FIGURE 5.8 : Strong landau damping test case : the space integrated distribution function with RK2 scheme using centered flux, $\Delta t = 2.5 \times 10^{-3}$. Numerical paramaters : $V_{\max} = 6$, $N_{x_1} = 64$, $N_{v_1} = 65$ dof. Uniform discretization in velocity.

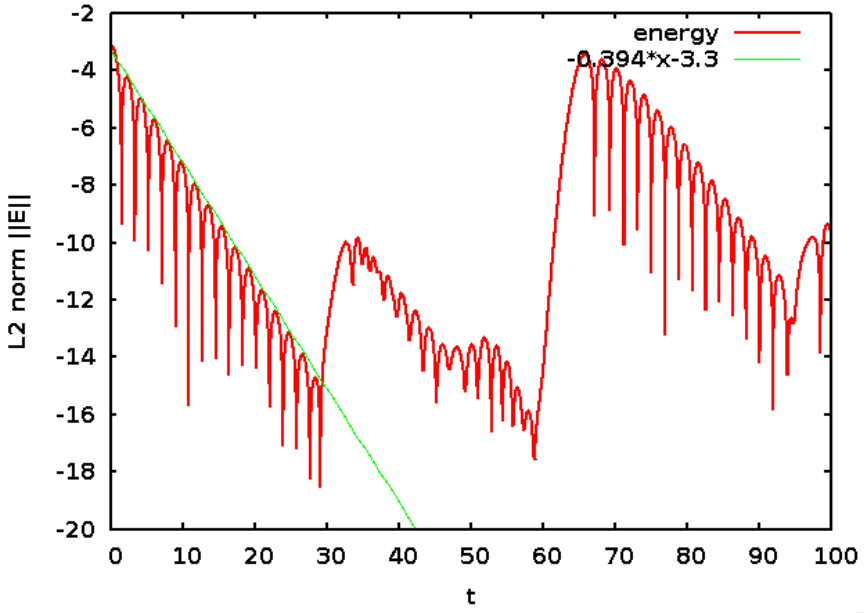


FIGURE 5.9 : The electric energy of the Landau damping 2D test case up to time $t = 100$. Parameters : d $= 2, M = 32, N_{\mathbf{x}} = 64, \Delta t = 0.01, T = 100$ RK3 scheme with centered flux, 64 processors. Computations time : 18964s on Curie

.

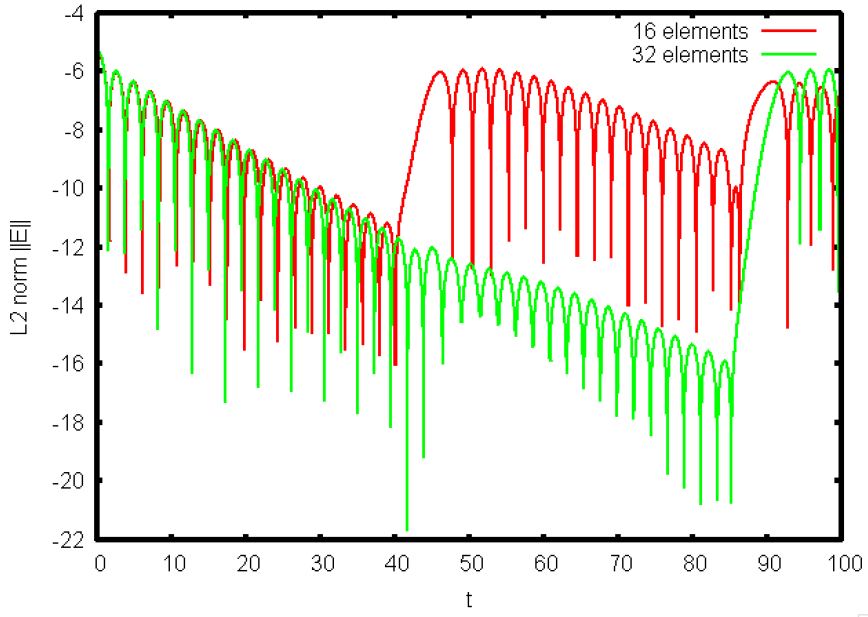


FIGURE 5.10 : Recurrence phenomena in the electric field up to time $T = 100$, comparison between the different choice of the number of elements in velocity : $M = 16$ (the red curve) and $M = 32$ (the green one). The uniform points discritization and Gauss-Legendre integration in velocity. Parameter : d $= 2$, $N_{x_1} = N_{x_2} = 16$, RK2 viscous flux ($\varepsilon = 0.005$)

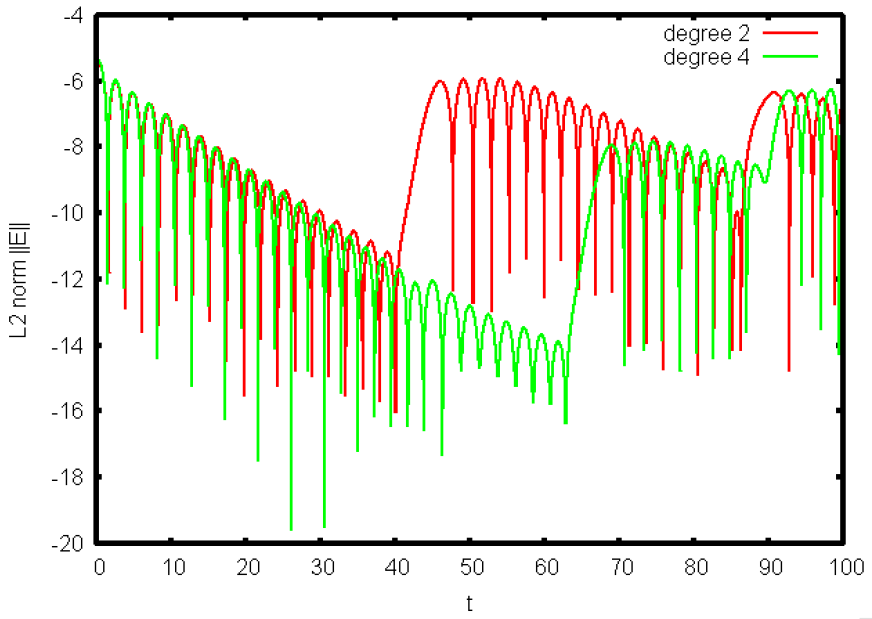


FIGURE 5.11 : Recurrence phenomenon in the electric field up to time $T = 100$, compare between the case degree of polynomials base is d = 2 (red) and in the case d = 4 (green). The uniform points discretization and Gauss-Legendre integration in velocity. Parameter : $M = 16$, $N_{x_1} = N_{x_2} = 32$, RK2 viscous flux ($\varepsilon = 0.005$), $\Delta t = 0.005$.

In the case of **Gauss-Lobatto discretization points** in velocity, we first set the degree of the polynomial basis (d = 2) and make vary the number of elements in velocity. We observe in Fig. 5.12 that the recurrence time is larger as we increase the number of elements : the recurrence time takes place at time $t = 11.8$ for $M = 8$, at time $t = 23.7$ for $M = 16$ and time $t = 47.3$ for $M = 32$. We then make vary the degree of polynomials while keeping the number of elements in the velocity equal to $M = 32$. In Fig.5.13, we remark that the recurrence phenomenon appears at the same time (at about $t = 40$) for degrees d = 2 and d = 4.

The numerical results allow us to conjecture that the recurrence phenomenon is always satisfying the analytic formula (see for instance [38]) : $T_{\text{rec}} = 2\pi/k\triangle v = 4\sqrt{2}\pi/\Delta v$, where $\Delta v = 12/M$ when using Gauss-Lobatto points and with $\Delta v = 12/M/d$ when using uniform points.

Remark 5.4.1. *We can also compare* ***the computation time*** *when using the two types of discretization points. We consider the following parameters :* $N_v = 65^2$ *(*$d = 2, M = 32$*),* $N_{x_1} = N_{x_2} = 32$*. The codes are executed on the irma-hpc2 machine hosted at the IRMA laboratory (Institut de Recherche Mathématique Avancée, UMR 7501, Strasbourg). For* 100 *iterations, the code with Gauss-Lobatto discretization spends* 363 *seconds while the code with uniform discretization points (with Gauss-Legendre quadrature rules) spends* 765.6 *second. Therefore, the code with Gauss-Lobatto points in velocity is about 2.1 times faster than the code with uniform discretization points. This is mainly due to the fact that the involved matrices (A, M) are diagonal in the case of Gauss-Lobatto points and thus no matrix inversion is required (see next chapter).*

The tests above have certified the validity of our code. Now we will assess the parallel

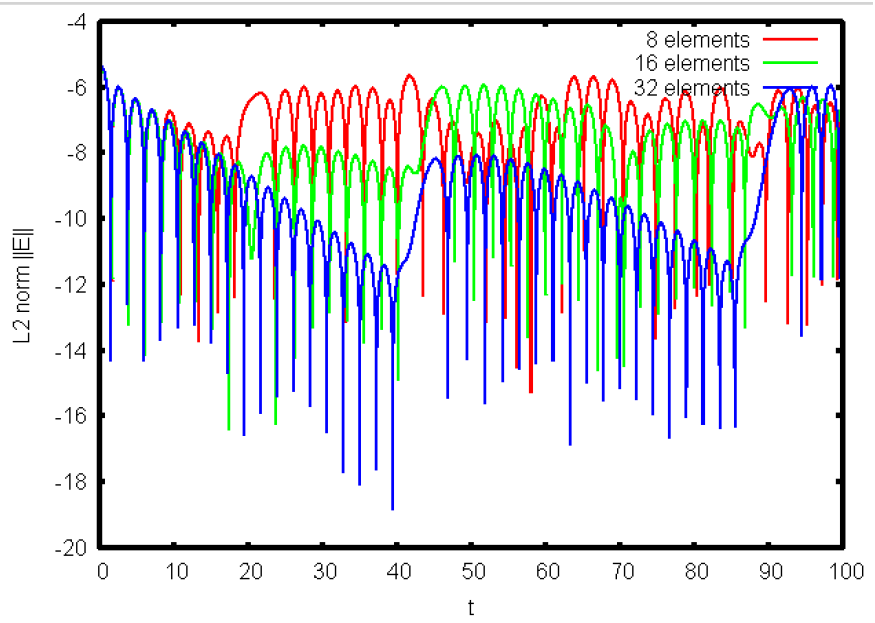


FIGURE 5.12 : Recurrence phenomena in the electric field up to time $T = 100$, comparison between the different choice of the number of elements in velocity : $M = 8$ (the red curve), $M = 16$ (the green one) and $M = 32$ (the blue one). The Gauss-Lobatto points discritization and integration in velocity. Parameter : d $= 2$, $N_{x_1} = N_{x_2} = 16$, RK2 viscous flux ($\varepsilon = 0.005$)

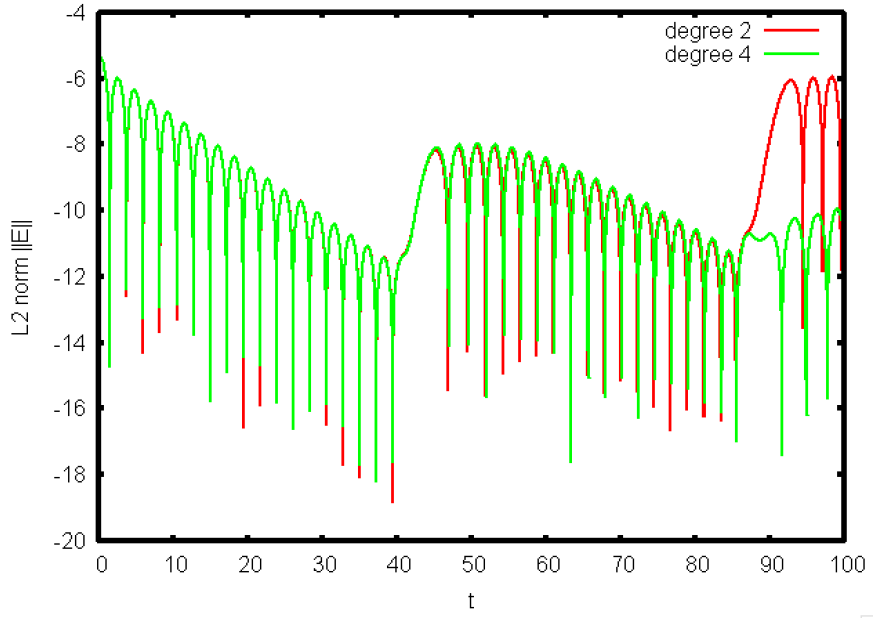


FIGURE 5.13 : Recurrence phenomenon in the electric field up to time $T = 100$, compare between the case degree of polynomials base is d $= 2$ (red) and in the case d $= 4$ (green). The Gauss-Lobatto points discritization and integration in velocity. Parameter : $M = 32$, $N_{x_1} = N_{x_2} = 32$, RK2 viscous flux ($\varepsilon = 0.005$), $\Delta t = 0.005$.

Number of processor units (CPU)	8	16	64	128
computation time (second)	7398	3673	943	491
speed-up (relative)		2.01	3.89	1.92
speed-up		16.11	62.76	120.54

TABLE 5.1 : Computational time and speedup for the 2D **finite-volume** code. Computed on the supercomputer Curie (TGCC). Parameters : 100 iterations, $N_v = 65^2$ ($d = 2, M = 32$), $N_{x_1} = N_{x_2} = 256$.

efficency.

5.4.3 Parallel code performance

We consider the Landau damping 2D test case. We here study the parallel performance of the code as the number of cores increases. We consider the following problem size : $N_v = 65 \times 65$ (d = 2, $M = 32$) points in velocity and $N_{x1} = N_{x2} = 256$ points in each spatial direction. We here use the uniform points (and Gauss-Legendre quadrature formula). The total number of discretization points equals :

$$nb_{points} = (32 \times 2 + 1)^2 \times 256^2 = 276\,889\,600.$$

We execute 100 iterations of the code in Curie, a supercomputer of GENCI and operated into the TGCC by CEA (see [1]). We choose the number of processors $nb_{\text{processors}} = 4, 8, 16, 32, 64, 128, 256$, knowing that each nodes of the supercomputer is composed of 32 cores (4 eight-core CPUs). We provide the computation time and the speedup in table 5.1 : the speed up obteained is good. The speedup is defined as the computational time with one processor divided by the computational time with $nb_{\text{processors}}$. Remark that with this choice of grid size, the one-processor computations cannot be realized because of limited memory for one core. In a perfect situation, with infinitely fast MPI communications, the speedup for $nb_{\text{processors}}$ processors would be $nb_{\text{processors}}$. We observe a speedup very closed to the ideal one.

As introduced in [6], we can also quantify the performance of the code with the efficiency, that is the number of advections per second and per processor (in million), defined by

$$\text{eff} := \frac{s * nb_{points} * nb_{iterations}}{T * 10^6 * nb_{\text{processors}}}, \tag{5.4.1}$$

in which s number of sup steps per time iteration, nb_{points} is the number of discretization points (in space and velocity) and T is the computational time. Considering the second order splitting discretization combined with the RK2 scheme , we have 8 sub-steps per time iterations (2 times the spatial and velocity advections).With the previous computation (with the same number of discretization points), we obtain an efficiency equals to : eff = 1.75. Note that it is less efficient than an optimized full semi-Lagrangian scheme in one dimension where efficiency of order 29 can be reached (see [6]). This may be due to the transport in velocity that requires a matrix multiplication for our scheme and also to the fact that we here include the computation of the electric potential at each time step.

5.5 Conclusion

In this chapter, we have implemented the finite volume method for the two-dimensional reduced Vlasov equation using subdomain paralellization techniques (MPI tool) : good We also compare two different kinds of discretizations points in velocity : uniform and Gauss-Lobatto points. We have observed that the Gauss-Lobatto points leads to smaller recurrence time. However, with the Gauss-Lobatto discretization points, the code is about twice faster since the matrices involved in the schemes are diagonal and it enables to use semi-Lagrangian schemes to solve the diagonal hyperbolic equation (see next chapter).

Chapitre 6

Semi-Lagrangian approach for the 2D reduced Vlasov-Poisson equation

Ce chapitre a été écrit en collaboration avec Philippe Helluy, Laurent Navoret, Malcolm Roberts, Sébastien Guisset, Michel Massaro et Michael Gutnic. Il a fait l'objet d'un projet de la recherche au CEMRACS (Centre d'Été Mathématique de Recherche Avancée en Calcul Scientifique) 2014 et a la publication suivante :

Lagrangian/Eulerian solvers and simulations for Vlasov-Poisson

Nous avons ajouté 2 *annexes : la première annexe décrit la méthode de semi-Lagrangienne avec plus de détails (notamment sur le splitting en temps) et la deuxième annexe présente les performances du code en terme de parallélisation.*

Lagrangian/Eulerian solvers and simulations for Vlasov-Poisson

Abstract

We construct a hyperbolic approximation of the Vlasov equation using a method of reduction [53, 80, 52] in which the dependency on the velocity variable is removed. The reduction relies on a semi-discrete finite element approximation in the velocity variable. We apply Gauss-Lobatto numerical integration in velocity space, reducing the hyperbolic system to a system of transport equations for which the transport velocities are the Gauss-Lobatto points. The transport equations are coupled through a zero-order term that represents the electromagnetic forces. We solve the resulting system by a splitting approach : the homogeneous transport equations are solved by a split semi-Lagrangian method and the source term is applied independently. We also present preliminary comparisons with another transport solver based on the discontinuous Galerkin method.

Résumé

Au moyen d'une méthode de réduction décrite dans [53, 80, 52] nous construisons une approximation hyperbolique de l'équation de Vlasov dans laquelle la dépendance en vitesse

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

6.1 Introduction

The Vlasov-Poisson system is a popular model for the numerical simulation in plasma physics. Solving the Vlasov-Poisson equation is challenging as it is composed of a time-dependent transport equation in a six-dimensional $(\mathbf{x}, \mathbf{v})$ phase-space coupled with the electric potential equation. Some popular methods for studying this equation are the particle-in-cell (PIC) method [14] or the semi-Lagrangian approach [41].

In a previous work [53] (see also [80], [52]), we constructed a reduced Vlasov-Poisson system where the dependency in the velocity variable $\mathbf{v}$ is removed. The principle is to approximate the distribution function $f(\mathbf{x}, \mathbf{v}, t)$ in the velocity variable $\mathbf{v}$ with a finite element interpolation ; this semi-discretization transforms the Vlasov equation into a hyperbolic system for which the unknowns system are the values of f at the interpolation nodes in $\mathbf{v}$. The hyperbolic system contains a zero order source term that represents the electric force.

Here, we apply the same strategy as in [53] but we replace the exact numerical integration by Gauss-Lobatto integration in the finite element approximation. This simplifies the nature of the hyperbolic system, reducing it to a system of transport equations for which the velocities are the Gauss-Lobatto points. The spatial transport and velocity source term are solved separately. The transport equation is solved by a semi-Lagrangian approach [27], [41]. We present numerical results on the classic test cases of Landau damping and the two-stream instability and evaluate the conservation properties of the scheme. Finally, we provide preliminary comparisons between the semi-Lagrangian transport solver and a recently developed discontinuous Galerkin (DG) transport solver.

6.2 Model reduction of the Vlasov-Poisson equation

We consider the two-dimensional Vlasov equation

$$\frac{\partial f}{\partial t}(\mathbf{x}, \mathbf{v}, t) + \mathbf{v} \cdot \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{v}, t) + \mathbf{E} \cdot \nabla_{\mathbf{v}} f(\mathbf{x}, \mathbf{v}, t) = \mathbf{0}. \tag{6.2.1}$$

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

$$\mathbf{E} = -\nabla_{\mathbf{x}} \Phi, \text{ with } -\Delta_{\mathbf{x}} \Phi = \rho - \overline{\rho}. \tag{6.2.2}$$

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

$$\rho(\mathbf{x},t)=\int_{\mathbf{v}} f(\mathbf{x},\mathbf{v},t)d\mathbf{v},\quad \overline{\rho}(t)=\frac{\int_{\mathbf{x},\mathbf{v}} f(\mathbf{x},\mathbf{v},t)d\mathbf{x}d\mathbf{v}}{|\Omega_x|}. \tag{6.2.3}$$

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

$$\mathbf{E}(\mathbf{x},t)\cdot\mathbf{n}_v(\mathbf{v})|_{\mathbf{v}\in\partial\Omega_v}<0\Rightarrow f(\mathbf{x},\mathbf{v},t)|_{\mathbf{v}\in\partial\Omega_v}=0. \tag{6.2.4}$$

Using the notation $(\mathbf{E}\cdot\mathbf{n}_v)^-=\min(\mathbf{E}\cdot\mathbf{n}_v,0)$, condition (6.2.4) is equivalent to

$$(\mathbf{E}\cdot\mathbf{n}_v)^- f(\mathbf{x},\mathbf{v},t)=0\quad\text{on }\partial\Omega_v. \tag{6.2.5}$$

Note that the boundary condition is trivially satisfied when $\mathbf{E}\cdot\mathbf{n}_v\geq 0$. The equations (6.2.1)-(6.2.2) are supplemented by an initial condition

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

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

$$f(\mathbf{x},\mathbf{v},t)\simeq\sum_{j=1}^{N_v} w_j(\mathbf{x},t)\varphi_j(v)=w_j(\mathbf{x},t)\varphi_j(\mathbf{v}). \tag{6.2.6}$$

(The convention of summation on repeated indices is used.) Several different bases can be used, see for instance [67] and references therein. Multiplying the Vlasov equation (6.2.1) by φ_i, integrating with respect to $\mathbf{v}$

$$\partial_t w_j\int_{\Omega_v}\varphi_i\varphi_j+\nabla_{\mathbf{x}}w_j\cdot\int_{\Omega_v}\mathbf{v}\varphi_i\varphi_j+\mathbf{E}w_j\int_{\Omega_v}\varphi_i\nabla_{\mathbf{v}}\varphi_j=0. \tag{6.2.7}$$

Let $(\mathbf{E}\cdot\mathbf{n}_v)^+=\max(\mathbf{E}\cdot\mathbf{n}_v,0)$. Using Green's formula two times, and applying the boundary conditions (6.2.5), we have

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

Finally

$$\partial_t w_j \int_{\Omega_v} \varphi_i\varphi_j + \nabla_{\mathbf{x}} w_j \cdot \int_{\Omega_v} \mathbf{v}\varphi_i\varphi_j + \mathbf{E}w_j \int_{\Omega_v} \varphi_i \nabla_{\mathbf{v}}\varphi_j - w_j \int_{\partial\Omega_v} (\mathbf{E}\cdot\mathbf{n}_v)^- \varphi_j\varphi_i = 0. \quad (6.2.8)$$

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

$$M\partial_t \mathbf{w} + A^1\partial_{x_1}\mathbf{w} + A^2\partial_{x_2}\mathbf{w} + B(\mathbf{E})\mathbf{w} = \mathbf{0}. \quad (6.2.9)$$

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

$$M_{i,j} = \int_{\Omega_v} \varphi_i\varphi_j, \quad A^1_{i,j} = \int_{\Omega_v} v^1\varphi_i\varphi_j, \quad A^2_{i,j} = \int_{\Omega_v} v^2\varphi_i\varphi_j, \quad (6.2.10)$$

and

$$B(\mathbf{E})_{i,j} = \mathbf{E}\cdot\int_{\Omega_v} \varphi_i\nabla_{\mathbf{v}}\varphi_j - \int_{\partial\Omega_v} (\mathbf{E}\cdot\mathbf{n}_v)^-\varphi_j\varphi_i. \quad (6.2.11)$$

The above procedure applies to any choice of velocity basis. As in [53, 80], we use a finite element interpolation basis, for other choices we refer to eg [67]. To construct the function basis, we first consider a regular square mesh of Ω_v made of finite elements of degree d. The basis functions are continuous on $\overline{\Omega}_v$ and polynomial of degree d in each square finite element. Gauss-Lobatto integration points are used in each element to discretize the variation of f with $\mathbf{v}$. Let M be the number of elements in each velocity direction, so that the number of Gauss-Lobatto points in the mesh is $N_v = (Md+1)^2$. We denote these points $\{N_i\}_{i=1}^{N_v}$. The basis functions satisfy the interpolation property

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

where δ_{ij} is the Kronecker delta. From the finite element construction, we also have access to the velocity mesh connectivity : for a given finite element index k, $0 \le k \le M^2$ and a local Gauss-Lobatto point index ℓ, $0 \le \ell \le (d+1)^2$, we are able to recover the global index i, $0 \le i \le N_v$, of the Gauss-Lobatto point. In this case, we also use the notation $N_i = N_{k,\ell}$. For more details we refer to [53], where the construction of the finite element basis is fully detailed in the one-dimensional case.

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

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

Then by using the Gauss-Lobatto quadrature, we obtain

$$\int_{Q_k} h(\mathbf{v})d\mathbf{v} \simeq \sum_{\ell=1}^{(d+1)^2} \omega_{k,\ell} h(N_{k,\ell}), \quad (6.2.12)$$

where $\omega_{k,\ell}$ is the weight associated to Gauss-Lobatto point $N_{k,\ell}$ in the quadrature formula. Finally, applying formula (7.4.10) we obtain that the matrices M and A^k are diagonal and given by

$$M_{i,i} = \sum_{N_i = N_{k,\ell}} \omega_{k,\ell} > 0, \quad (6.2.13)$$

and

$$A^k_{i,i} = M_{i,i} V^k_i, \quad \mathbf{V}_i = (V^1_i, V^2_i). \tag{6.2.14}$$

The matrix $B(\mathbf{E})$ is sparse but not diagonal; with the standard numbering of the finite element interpolation points its bandwidth is $Md+1$. We can also rewrite the system (6.2.9) as

$$\partial_t \mathbf{w} + M^{-1}A^1 \partial_{x_1} \mathbf{w} + M^{-1}A^2 \partial_{x_2} \mathbf{w} + M^{-1}B(E)\mathbf{w} = \mathbf{0}. \tag{6.2.15}$$

where $M^{-1}A^1$ and $M^{-1}A^2$ are diagonal. We call equation (6.2.15) the reduced Vlasov model. We observe that thanks to the choice of the basis functions and the numerical quadrature, the reduced Vlasov model is simply a system of transport equations that are coupled through the source term $M^{-1}B(\mathbf{E})\mathbf{w}$.

6.3 Numerical methods for solving the reduced Vlasov model

In this section, we present three methods to solve the reduced Vlasov model on Cartesian meshes. The finite volume method (subsection 6.3.1) and the semi-Lagrangian method (subsection 6.3.2) rely on a directional-splitting method to replace the $2D$ problem by a pair of one-dimensional problems. Let $\mathbf{w}^n$ be an approximation of $\mathbf{w}$ at time t_n, the time discretization is

$$\frac{\mathbf{w}^* - \mathbf{w}^n}{\Delta t} + M^{-1}A^1 \, \partial_{x_1} \mathbf{w}^n = 0, \tag{6.3.1}$$

$$\frac{\mathbf{w}^{**} - \mathbf{w}^*}{\Delta t} + M^{-1}A^2 \, \partial_{x_2} \mathbf{w}^* = 0, \tag{6.3.2}$$

$$\frac{\mathbf{w}^{n+1} - \mathbf{w}^{**}}{\Delta t} + M^{-1}B(\mathbf{E})\mathbf{w}^{**} = 0. \tag{6.3.3}$$

where $\mathbf{w}^*$ and $\mathbf{w}^{**}$ are intermediate unknowns. The discontinuous Galerkin method (subsection 6.3.3) can be applied without having to apply the above splitting technique.

6.3.1 The finite volume method

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

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

We obtain the following numerical scheme

$$\frac{\mathbf{w}^{n+1}_i - \mathbf{w}^n_i}{\Delta t} = -\frac{\mathbf{F}(\mathbf{w}_i, \mathbf{w}_{i+1}) - \mathbf{F}(\mathbf{w}_{i-1}, \mathbf{w}_i)}{\Delta x}. \tag{6.3.4}$$

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

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

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

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

For more details we refer to [53].

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

6.3.2 The semi-Lagrangian method

We use the same spatial mesh as in subsection 6.3.1. Since A^1 is diagonal, system (6.3.1) is a time-discretization of N_v one-dimensional transport equations with constant velocities V_i which evolves according to

$$\partial_t \mathbf{w} + V_i \, \partial_x \mathbf{w} = \mathbf{0}. \tag{6.3.5}$$

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

$$\mathbf{w}(t^{n+1}, x) = \mathbf{w}(t^n, x - V_{1,i} \, \Delta t). \tag{6.3.6}$$

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

$$\mathbf{w}_i^{n+1} = [\mathbf{\Pi} \mathbf{w}^n] \big(x_i - V_{1,i} \, \Delta t\big), \tag{6.3.7}$$

where $\mathbf{\Pi} \mathbf{w}^n$ is an interpolation function built from the points $(x_i, \mathbf{w}_i^n)$. Several interpolation methods have been studied (see eg [27]). We perform an interpolation π using either a cubic spline (with accuracy $\mathcal{O}(\Delta x^3)$) or a Lagrange polynomial with $2r+1$ points (with accuracy $\mathcal{O}(\Delta x^{2r+1})$) which we denote

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

We refer to [91, 41] for more details. One of the main advantage of the semi-Lagrangian method is that the size of the time step is not bounded above by a stability condition. Consequently, the time-step size is only limited by the required precision and the stability condition of the finite-element method used in $\mathbf{v}$.

The Strang splitting can be modified in order to obtain second order accuracy in time [91, 41].

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

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

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

The discontinuous Galerkin (DG) method is a generalization of the finite volume method for solving systems of type (6.2.15). The computational domain Ω_x is covered with a mesh ; this mesh need not be structured or conforming, nevertheless in practice we choose a structured and conforming hexahedron mesh because the geometry of Ω_x is very simple. In general, each cell L of the mesh is obtained from a second order polynomial transformation that maps the reference cube $\hat{L}$ onto L. The cells of our mesh can thus be "curved" hexahedrons, even if this feature is not exploited in this work. In each cell L of the mesh, the field is approximated by polynomial basis functions

$$\mathbf{w}(\mathbf{x},t) = \mathbf{w}_L^j(t)\psi_j^L(\mathbf{x}), \quad \mathbf{x} \in L. \tag{6.3.8}$$

The numerical solution satisfies the DG approximation scheme

$$\int_L \partial_t \mathbf{w}\psi_i^L - \int_L \mathbf{F}\big(\mathbf{w},\mathbf{w},\nabla\psi_i^L\big) + \int_{\partial L} \mathbf{F}(\mathbf{w}_L,\mathbf{w}_R,\mathbf{n}_{LR})\psi_i^L = 0 \tag{6.3.9}$$

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

$$\mathbf{F}(\mathbf{w},\mathbf{w},\mathbf{n}) = A^k n^k \mathbf{w}. \tag{6.3.10}$$

We apply Gauss-Lobatto numerical integration in $\mathbf{x}$ using nodal basis functions $\psi_i^L(G_k^L) = \delta_{i,k}$. The Gauss-Lobatto integration points and associated weights are noted G_k^L and ω_k^L and the quadrature formula is $\int_L h(X)dX \simeq \sum_k \omega_k^L h(G_k^L)$. The Gauss-Lobatto points are first defined on the reference cube by tensor products of one-dimensional points. They are then mapped to L by the geometric transformation. Similarly, the nodal basis function ψ_i^L are obtained from tensor products of one-dimensional Lagrange polynomials and mapped from $\hat{L}$ to L. In the end, we perform time integration of a system of ordinary differential equations. For a similar approach (but with Gauss-Legendre points instead of Gauss-Lobatto points), we refer to the PhD of Thomas Strub [94] (see also [52]).

We have implemented the DG algorithm in the `OpenCL` framework, which is a software environment for programming GPUs or multicore accelerators. Unlike `CUDA`, which is only available for NVIDIA GPUs, `OpenCL` allows access to multicore accelerators of many brands. We will not describe here the philosophy of `OpenCL` and the full details of the DG implementation, but refer readers to previous works [95, 52]. We have also implemented an `OpenMP` version of the same algorithm for use when `OpenCL` is not available and for verification and comparisons. The resulting software `schnaps` can be found at `schnaps.gforge.inria.fr`.

In contrast to our previous works we have also provided additional features, such as a macrocell strategy for managing the DG mesh : the connectivity of the mesh is described at a macro level with H20 quadratic hexahedrons[2] with control points on each vertex and in the middle of each edge, allowing for curved elements in physical space via a non-linear transformation. Then, each macrocell is split into subcells that share the same macro geometry data. In the implementation of the DG algorithm, a macrocell is associated to a single

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

`OpenCL` kernel, subcells are associated to `OpenCL` work-groups, and finally the Gauss points are associated to `OpenCL` work-items. This organization reduces memory access to the mesh connectivity. The development of `schnaps` is still in progress. In this paper, we shall only present preliminary results.

6.4 Numerical results

6.4.1 Convergence rate

We first verify that the convergence orders of transport solvers corresponds to theory. Equation (6.2.9) is solved with an electric field equal to zero leading a vanishing source term $B(\mathbf{E})w = 0$. We compare the exact and the approximate solution by translating the initial conditions using the equation

$$f(\mathbf{x}, \mathbf{v}, t) = f(\mathbf{x} - t\mathbf{v}, \mathbf{v}, 0). \tag{6.4.1}$$

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

$$f(\mathbf{x}, \mathbf{v}, t) = f(\mathbf{x}, \mathbf{v} - t\mathbf{E}, 0). \tag{6.4.2}$$

`SELALIB` SL solver

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

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \sin(\pi x). \tag{6.4.3}$$

We used a second-order time-stepper with $\Delta t = 0.01$ and advanced the system to $t = 0.5$ and $M = 17$ grid points in v_1. The spatial convergence agreed well with the theoretical convergence rate for a cubic spline interpolation. Results are shown in Figure 6.1.

The initial condition for the velocity transport case was

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \begin{cases} (1 - 2v)^3(1 + 2v)^3 & \text{if } v < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{6.4.4}$$

where $v = \sqrt{v_1^2 + v_2^2}$. The electrical field was set to $\mathbf{E} = \mathbf{v}_1$. We used a second-order time-stepper with $\Delta t = 0.01$ and advanced the system to $t = 0.2$. We observe a convergence close to the theoretical value of 2 for second order finite elements. Results are shown in Figure 6.1.

`schnaps` DG solver

The initial condition used for this test was

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{v^2}{\sigma}} \begin{cases} e^{\frac{-1}{1-4r^2}} & \text{if } r < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{6.4.5}$$

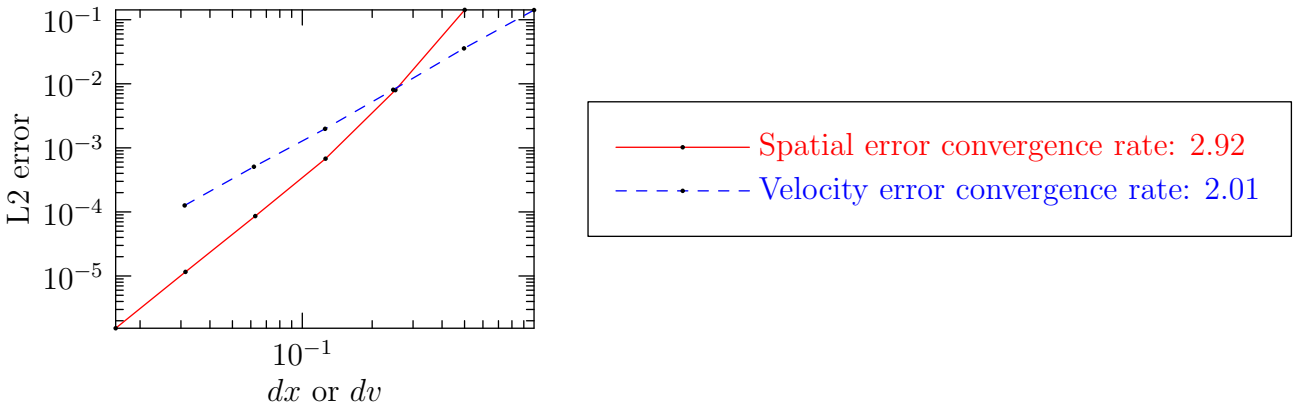


FIGURE 6.1 : The relative L^2 error between the numerical solution and the analytic solution as a function of grid spacing for `SELALIB`. An interpolation spline of order 3 is used in space and a second order finite element method is used for the velocity transport.

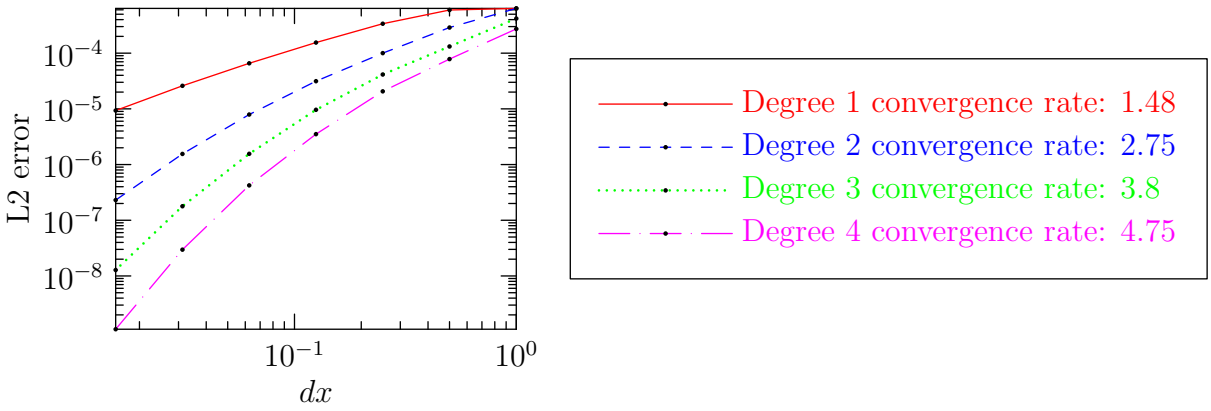


FIGURE 6.2 : The relative L^2 error between the numerical solution and the analytic solution as a function of grid spacing for `schnaps` using Gauss-Lobatto interpolation with polynomial degree between 1 and 4.

where $r = \sqrt{x_1^2 + x_2^2}$, $v = \sqrt{v_1^2 + v_1^2}$, and $\sigma = 0.2$. This initial condition consists of a Gaussian in the velocity multiplied by a C^∞ function with support in $B_{\mathbf{x}}(\mathbf{0}, 1/2)$. The tests were done with 30 velocity components in direction $\mathbf{v}_1$ and $\mathbf{v}_2$, the computational domain was $\Omega = \Omega_v \times \Omega_x = (-1, 1)^4$, and the maximum time was $t = 0.4$. For each combination of degree and spatial resolution the fourth-order Runge-Kutta time step was decreased by a factor of two until subsequent error values differed by no more than 1%, thereby guaranteeing that the error was independent of the choice of time-step. The results are shown in Figure 6.2 and we observe that the convergence rate in the L^2 norm is slightly larger than to the polynomial order d. This results again complies with the theory. Let us observe that if we had used Gauss-Legendre integration, we would have reached a convergence rate of $d + 1$. Gauss-Lobatto points were chosen so as to allow better performance with respect to memory access and to reduce the implementation cost.

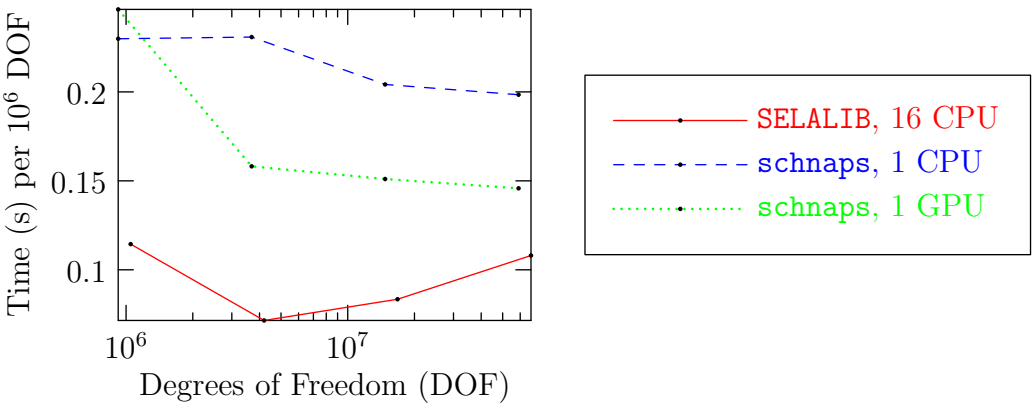


FIGURE 6.3 : Computation time per second-order time-step per 10^6 degrees of freedom for `SELALIB` with 16 CPUs and `schnaps` with 1 CPU or 1 graphics card.

6.4.2 Performance of the SL and DG transport solvers

In this section we compare the performance of the semi-Lagrangian and the DG transport solvers. We consider only the transport in $\mathbf{x}$, ie the source term in (6.3.3) is deactivated. At first sight, the semi-Lagrangian method should have a clear advantage; the scheme is universally stable and there is no error due to time discretization. However, memory access is more complicated for the semi-Lagrangian method than for the DG algorithm, and the parallelization of the semi-Lagrangian method relies on the transposition of the computational grid (`MPI_Alltoall`). We compare the speed of the two codes using parameters which could be considered realistic for research simulations. Velocity was discretized with 32 points in both dimensions (30 for `schnaps` due to a current technical limitation) and between 32 and 256 points in both spatial directions. The initial condition was a Gaussian in the velocity multiplied by a 6^{th} degree C^2 piecewise defined polynomial with compact support in $B_{\mathbf{x}}(\mathbf{0}, 1/2)$. The particular form of the initial condition is

$$f(\mathbf{x}, \mathbf{v}, t)|_{t=0} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{v^2}{\sigma}} \begin{cases} \frac{35}{16}(1-2r)^3(1+2r)^3 & \text{if } r < 1/2 \\ 0 & \text{otherwise.} \end{cases} \tag{6.4.6}$$

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

The computational domain for this test is $\Omega = \Omega_x \times \Omega_v = (-1, 1)^4$. We evolved the system until time $t = 0.4$ to be consistent with both the homogeneous Dirichlet boundary conditions imposed in `schnaps` and the periodic boundary conditions imposed in `SELALIB`. The comparison of computational time per second-order time-step per million degrees of freedom is shown in Figure 6.3. The three implementations, namely `SELALIB`, `schnaps-C`, and `schnaps-OpenCL`, demonstrate roughly similar computation time per second-order time-step for the case studied. Since the time-step of the DG code is bounded by a stability condition, a semi-Lagrangian method may be a better choice if it can be shown that the error due to time discretization using a large value of dt does not significantly alter the results.

6.4.3 Comparison of the SL and FV methods

In this section we benchmark the spatial semi-Lagrangian / velocity finite-element method with an upwind finite-volume method developed previously [53, 80]. We consider the full Vlasov system as given in equation (6.2.15).

Precision

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

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

$$f_0(\mathbf{x}, \mathbf{v}) = (1 + \varepsilon \cos(k x_1)) \frac{1}{\sqrt{2\pi}} e^{-\frac{v_1^2}{2}}, \tag{6.4.7}$$

where $k = 0.5$ and $\varepsilon = 5 \times 10^{-3}$. The spatial domain is periodic with length $L = 2\pi/k$. An analytic solution for the linearized problem is given in [89]. We plot the electric energy

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

of the analytic solution, of the classic finite volume method, and of the semi-Lagrangian method on Figure 6.4. Both numerical methods capture the theoretical damping rate well. We observe that the solution obtained by the semi-Lagrangian method is closer to the analytic solution than the solution of the classic finite volume method. This is due to the fact that semi-Lagrangian method (order 3 in space) is more precise than the finite volume method (order 1 in space).

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

$$f_0(\mathbf{x}, \mathbf{v}) = (1 + \varepsilon \cos(k x_1)) \frac{1}{2\sqrt{2\pi}} \left(e^{-\frac{(v_1 - v_0)^2}{2}} + e^{-\frac{(v_1 + v_0)^2}{2}} \right), \tag{6.4.8}$$

where $k = 0.2$, $\varepsilon = 5 \times 10^{-3}$ and $v_0 = 3$. The distribution function at time $t = 25$ and $t = 50$ for the two methods is compared in Figure 6.5. We observe that both methods do a good job at capturing the filamentation in phase-space, with the semi-Lagrangian / finite-element method being slightly more precise. The results are quite similar since they both use the same discretization in velocity.

Stability

The time-step of semi-Lagrangian / finite element method is only constrained by a stability condition in the $\mathbf{v}$-direction, since the semi-Lagrangian method in the $\mathbf{x}$-direction does not restrict the time-step. In order to compare the stability of the SL and the FV methods, we

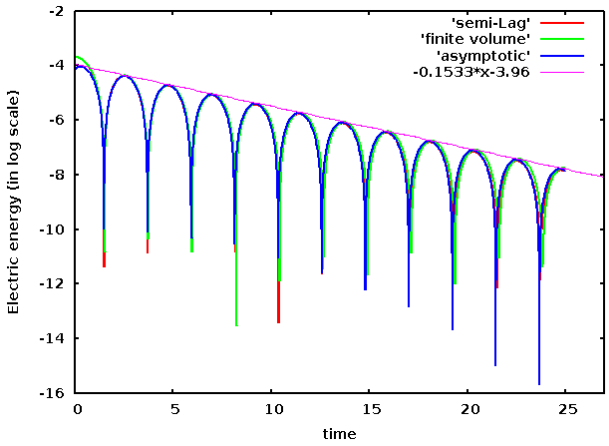


FIGURE 6.4 : The electric energy of the Landau damping 1D test as a function of time. The blue curve labelled "asymptotic" is the exact solution of the linearized equation, the red curve is from the semi-Lagrangian/finite element method and the green curve is from the finite volume method.

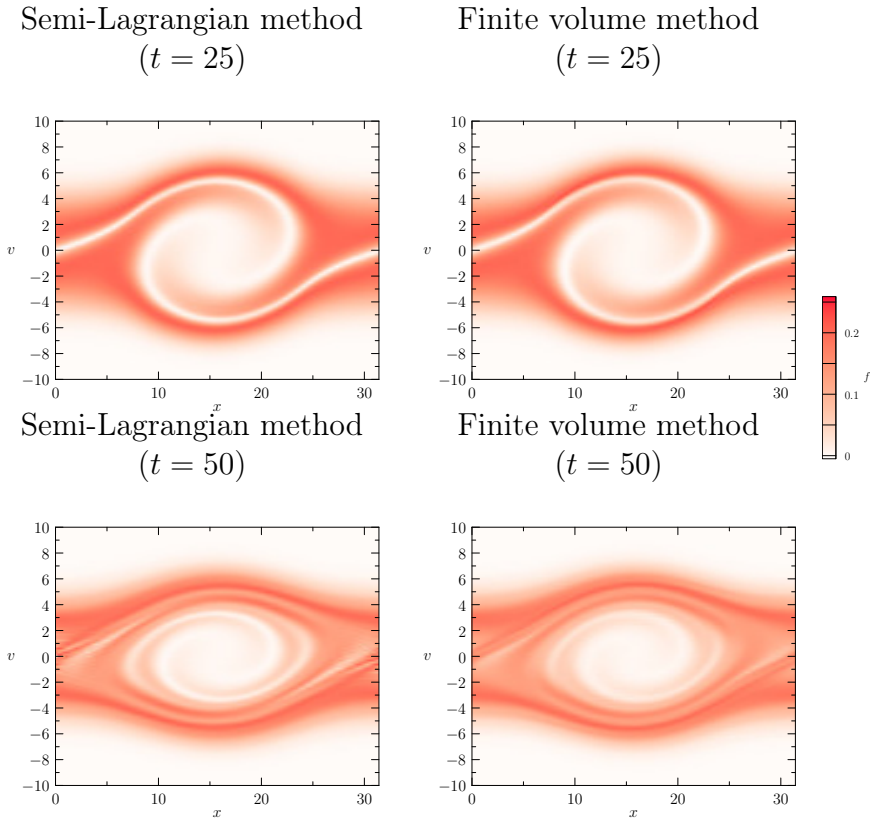


FIGURE 6.5 : The distribution function $f(\mathbf{x}, \mathbf{v}, t)$ for the double-stream instability test-case at time $t = 25$ (on the top) and at $t = 50$ (in the bottom) in the (x_1, v_1) phase space. Results from the semi-Lagrangian method (left) and the classic finite volume method (right with slightly upwinded flux $\kappa = 0.008$) are shown.

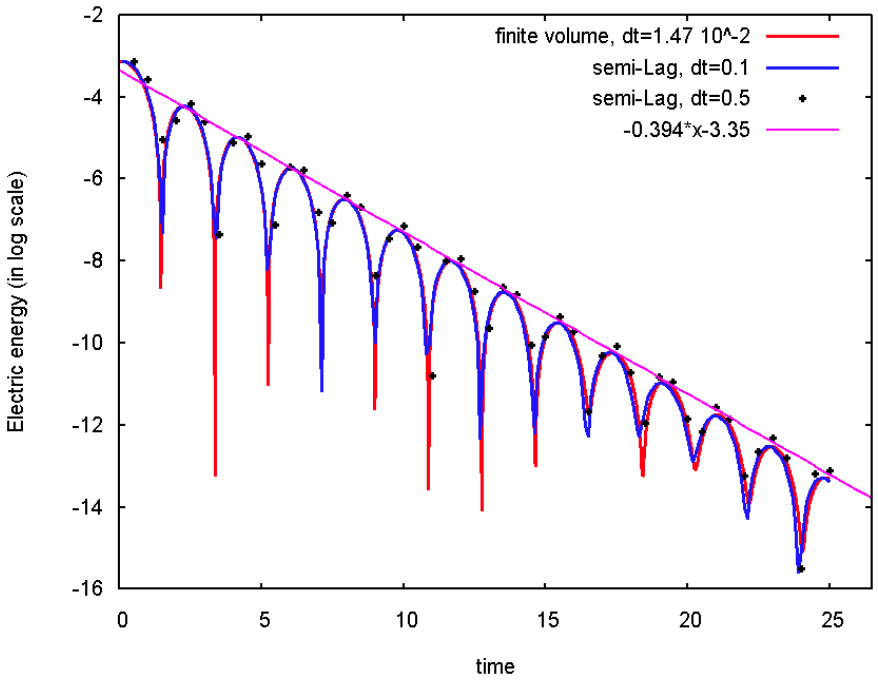


FIGURE 6.6 : The electric energy of the Landau damping 2D test case as a function of time. The red curve is computed using the classic finite volume method with $\Delta t = 1.47 \times 10^{-2}$, the blue curve is computed with the semi-Lagrangian/finite element method with $\Delta t = 0.1$, and the black points are computed with computed with the semi-Lagrangian/finite element method with $\Delta t = 0.5$.

can fix the number of cells and then vary the time step. We consider a 4D Landau damping test case. The initial condition is

$$f_0(\mathbf{x}, \mathbf{v}) = \frac{1}{2\pi}(1 + \varepsilon \cos(k_1 x_1) \cos(k_2 x_2)) e^{-\frac{(v_1^2 + v_2^2)}{2}}, \tag{6.4.9}$$

where $\varepsilon = 5 \times 10^{-3}$, the wave numbers $k_1 = k_2 = 0.5$ and $L_1 = L_2 = 4\pi$. The theoretical damping rate of the energy field is 0.394 [42]. We fix the number of cells at 32×32 in space and the degree 2 with 32 elements in each dimension of velocity. With the finite volume method, we have to use the time step small enough to ensure the stability of the energy field, for example here Δt is 1.47×10^{-2}. For this test-case, the electric field is very weak, so the stability condition in the $\mathbf{v}$-direction does not significantly restrict the time step. Large time step can thus be used for the semi-Lagrangian / finite element method and this produces qualitatively reasonable results even with $\Delta t = 0.1$.

Conservation

The Vlasov equation conserves the L^1 and L^2 norms of the distribution function f. To test the conservation properties of the semi-Lagrangian/finite element and the finite-volume methods, we consider the strong Landau damping 1D test case with initial condition given by (6.4.7) but with the parameter $\varepsilon = 0.5$ and $k_x = 0.5$. The number of grid points in x_1 is 32 and the number of grid points in v_1 is 65. We take the time step $\Delta t = 0.125$ for the semi-Lagrangian method and $\Delta t = 0.025$ for the finite volume method. The mass is defined

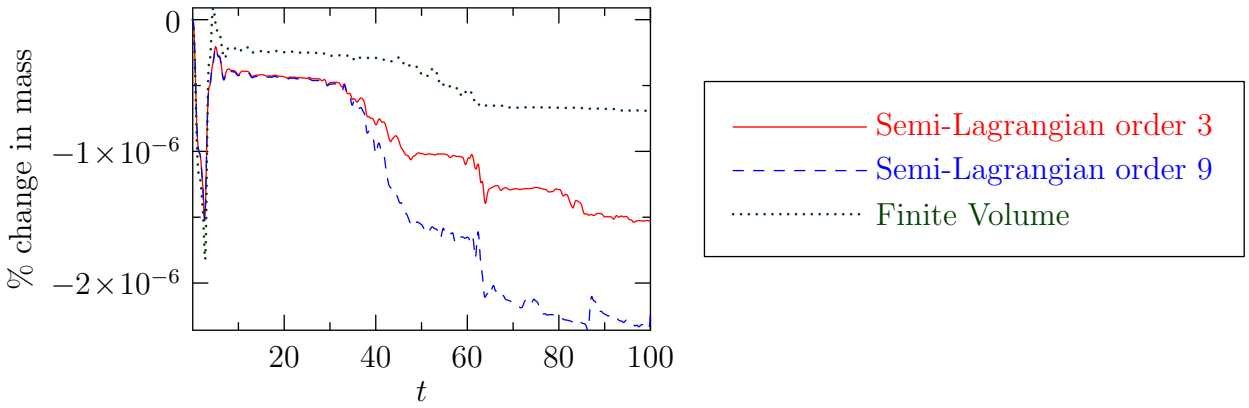


FIGURE 6.7 : Time evolution of the change in mass as function of time for the strong Landau damping 1D test case. The semi-Lagrangian schemes (order 3 : cubic spline interpolation, order 9 : Lagrange interpolation) and the finite volume scheme ($\kappa = 0.02$) are compared.

as

$$\int_{\Omega_{\mathbf{x}}\times\Omega_{\mathbf{v}}} f(\mathbf{x},\mathbf{v},t)\,d\mathbf{x}\,d\mathbf{v} \tag{6.4.10}$$

and its time evolution is shown in Figure 6.7. Due to the boundary condition in velocity, global mass is a priori not a conserved quantity, though the initial density quickly decays to zero as v increases, so the mass loss should not be very large. For this test-case, the three method (semi-Lagrangian cubic spline, semi-Lagrangian Lagrange order 9, finite volume) leads to conservation of mass up to a relative error of $\mathcal{O}(10^{-8})$. The time evolution of the L^1 and L^2 norms of f are shown in Figure 6.8. The finite volume method has better conservation of the L^1 norm for the cases studied here. Indeed, as the finite volume scheme is more diffusive (already observed in section 6.4.3), it better preserves the positivity of the distribution function. The L^2 norm is better conserved by the semi-Lagrangian/finite element scheme than by the finite volume scheme. The large numerical dissipation of the finite volume scheme leads to a relative error of $\mathcal{O}(0.1)$ in the L^2 norm. As observed in [41], the Lagrange interpolation of order 9 should be used in order to obtain numerical dissipation lower than the cubic spline interpolation.

6.4.4 Comparison of Semi-Lagrangian and discontinuous Galerkin Method

The two methods are compared for (x, v) with $x \in (-1,1)\times(-1,1)$ and $v \in (-1,1)\times(-1,1)$. The initial conditions are given by

$$f(x,v,t=0) = \frac{1}{\sigma\sqrt{2\pi}} e^{-e^{\frac{\|v\|^2}{\sigma}}} e^{\frac{-1}{1-4\|x\|^2}} 1_{\{x,\|x\|\leq 0.5\}} \tag{6.4.11}$$

where $\sigma = 0.2$. The cross-sectional profiles for this initial condition are shown in Figure 6.4.4. The semi-Lagrangian simulations imposed periodic boundary conditions whereas the discontinuos Galerkin simulations imposed homogeneous Dirichlet boundary conditions. The different boundary conditions are consistent with eachother when considering the advection of an initial velocity distribution f with compact support in $B(0, 0.5)$ for final times $t_{\max} < 0.5$, as is the case in out simulations.

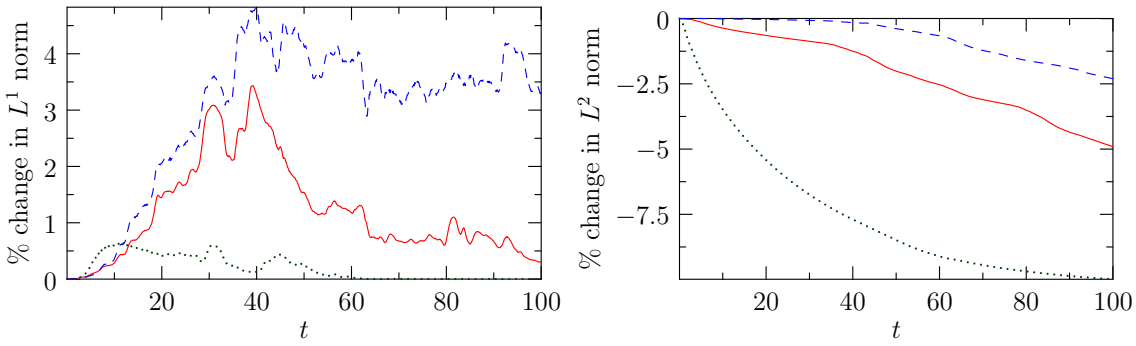


FIGURE 6.8 : Time evolution of the change of the L^1 (left) and L^2 (right) norms of f as a function of time for the strong Landau damping 1D test case. Results from the semi-Lagrangian/finite element method are shown using interpolation using cubic spline of order 3 (red, solid) and Lagrange interpolation of order 9 (blue, dashed), and the finite volume method with $\kappa = 0.02$ (green, dotted).

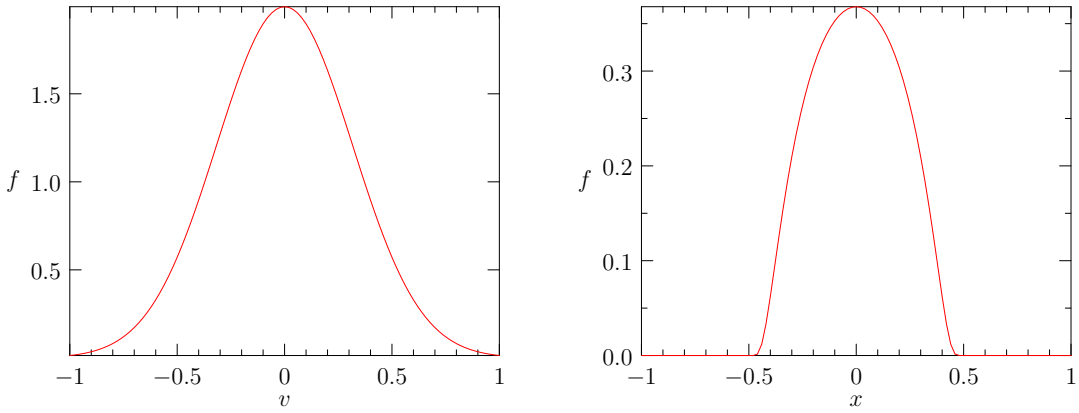


FIGURE 6.9 : Velocity (left) and spatial (right) profiles for the initial conditions.

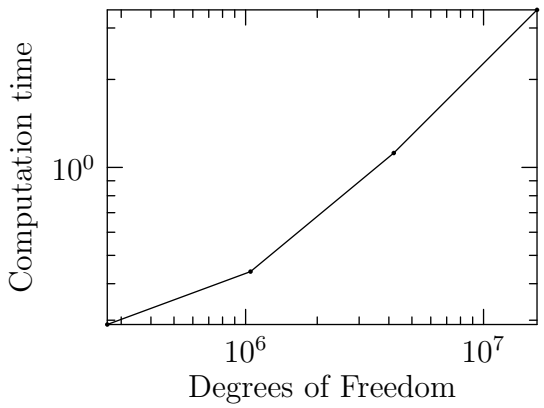


FIGURE 6.10 : Time per RK2 time-step for `SELALIB`.

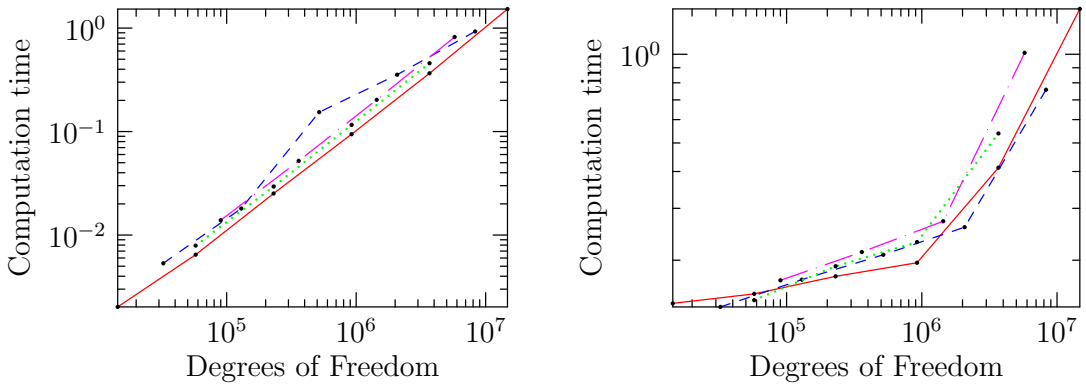


FIGURE 6.11 : Time per RK2 time-step for `schnaps` using `OpenMP` (left) and `OpenCL` (right) with 900 velocity components. The interpolation degree is 1 (red, solid), 2, (blue, dashed), 3 (green, dotted), or 4 (long-dashed, purple).

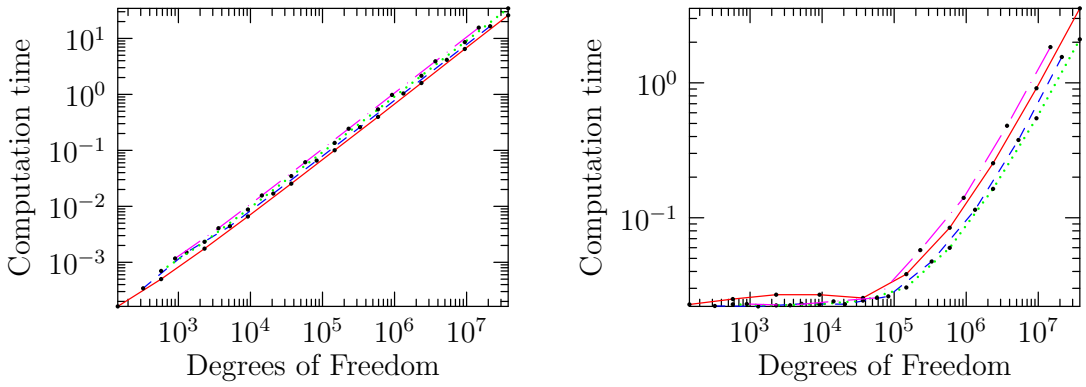


FIGURE 6.12 : Time per RK2 time-step for `schnaps` using `OpenMP` (left) and `OpenCL` (right) with 9 velocity components. The interpolation degree is 1 (red, solid), 2, (blue, dashed), 3 (green, dotted), or 4 (long-dashed, purple).

6.5 Conclusion and Future Work

6.5.1 Conclusion

In this paper, we presented and compared several high order methods to efficiently solve the reduced Vlasov model (6.2.15) in space.

The semi-Lagrangian / finite element (SL/FE) `SELALIB` implementation was compared with the discontinuous Galerkin (DG) method implemented in `schnaps` using a test case which consisted of transport of a multi-valued velocity field. The SL/FE and DG method showed the appropriate convergence rates. The speeds of the two implementations were compared. `schnaps` had better performance per CPU, making it a priori faster, but `SELALIB` has less stringent stability requirements on the time step, which will allow it to simulations faster for certain configurations. The SL/FE method was compared with a finite volume method using the 1D double-stream instability test-case and the 2D Landau damping test case for the Vlasov-Poisson equation. This allowed us to successfully validate the Vlasov-Poisson implementation in `SELALIB`. The SL/FE method was able to use much larger time-steps than the finite volume method, making the SL/FE method more efficient for the test cases considered.

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

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

6.5.2 Future Work

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

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

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

Acknowledgements

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

Annexe 6.A Semi-Lagrangian method

6.A.1 Principle of the semi-Lagrangian method

For the sake of completeness, we present the general framework of the semi-Lagrangian method for a general transport equation

$$\partial_t f + a(\mathbf{x}, t) \cdot \nabla_{\mathbf{x}} f = 0 \tag{6.A.1}$$

where $a(\mathbf{x}, t) \in \mathbb{R}^d$ is the advection field. The characteristics of this equation are the solutions $X(t; x, s)$ to the differential equations

$$\frac{dX(t; x, s)}{dt} = a(X(t; x, s), t), \tag{6.A.2}$$

$$X(s; x, s) = x. \tag{6.A.3}$$

If we have $X(t; x, s)$ the solution of (6.A.3) then by taking the derivative of $f(X(t), t)$ with respect to t, we then obtain

$$\frac{d}{dt}\Big[f(X(t; x, s), t)\Big] = \frac{\partial f}{\partial t}(X(t; x, s), t) + \frac{dX(t; x, s)}{dt} \cdot \nabla_x f(X(t; x, s), t) \tag{6.A.4}$$

$$= \frac{\partial f}{\partial t}(X(t; x, s), t) + a(X(t; x, s), t) \cdot \nabla_x f(X(t; x, s), t) = 0. \tag{6.A.5}$$

The distribution function f is thus constant along the characteristics. So the principle of the semi-Lagrangian method is (see [89, 27] for more details) as follows : to compute the f_i^{n+1} an approximation of $f(t^{n+1}, x_i)$ at point x_i,

1. find the foot of the characteristic at time t^n that equals x_i at time t_{n+1} : it corresponds to the point $X(t_n; x_i, t_{n+1})$. We thus have : $f(t^{n+1}, x_i) = f\big(t^n, X(t_n; x_i, t_{n+1})\big)$. That why it is also called the backward semi-Lagrangian method. In practice, the computation of these backward points requires to use backward numerical scheme to the differential equations (6.A.3).

2. since f^n is actually known only on the discrete nodes, we have to interpolate the values $(f_i^n)_i$ to have an approximation of $f\big(t^n, X(t_n; x_i, t_{n+1})\big)$. We thus set : $f_i^{n+1} = [\mathbf{\Pi} f^n](X(t_n; x_i, t_{n+1}))$ where $\mathbf{\Pi}$ is the interpolation operator.

The scheme simplifies when the advection field $a(\mathbf{x}, t)$ is constant. Let us consider the one-dimensional case : $f(t, x)$ is the solution to the following one dimensional transport equation

$$\partial_t f + a\, \partial_x f = 0, \quad f(t = 0, x) = f_0(x). \tag{6.A.6}$$

with $a \in \mathbb{R}$ is the constant advection velocity, on the bounded domain $[0, L]$ with periodic boundary conditions. Since f is constant along the characteristic, the exact solution is given by

$$f(t, x) = f\big(s, x - a(t - s)\big), \quad \forall t, s \geq 0. \tag{6.A.7}$$

We split the spatial domain $[0, L]$ into N cells with $N + 1$ nodes $x_i = i\Delta x$ where $\Delta x = L/N$. We also consider a sequence of times t_n, $n \in \mathbb{N}$, such that $t_0 = 0$ and $t_n = n\Delta t$, with $\Delta t > 0$.

Still denoting f_i^n the approximation of f at time t_n and at node x_i, the semi-Lagrangian scheme writes :

$$f_i^{n+1} = \mathbf{\Pi} f^n(x_i - a\Delta t).$$

This can be written in matrix form : $f^{n+1} = Cf^n$, where $f^n \in \mathbb{R}^N$ is the vector of unknowns and $C \in M_N(\mathbb{R})$ is a matrix depending on the chosen interpolation. Since the mesh is uniform and the boundary conditions are periodic, we can show that the matrix C is a circulant matrix. Several interpolation operator have been studied : for instance, the cubic spline interpolation, the Lagrange interpolation.

Let us return to our Vlasov equation (6.2.15) : applying the spitting method (in space), solving the transport part of this equation is equivalent to solving the following two transport equations

$$\partial_t \mathbf{w} + M^{-1} A^k \partial_{x_k} \mathbf{w} = 0, \tag{6.A.8}$$

with $k = 1, 2$. Since the matrices $M^{-1}A_k$ are diagonal (with the use of the Gauss-Lobatto points) and the diagonal entries equal

$$(M^{-1}A_k)_{ii} = V_i^k, \tag{6.A.9}$$

where V_j^k are the Gauss-Lobatto points. Equation (6.A.10) is thus equivalent to solve P transport equations with constant advection

$$\partial_t \mathbf{w}^j + V_j^k \partial_{x_k} \mathbf{w}^j = 0 \quad j = 1 \dots N_v. \tag{6.A.10}$$

6.A.2 A time step discretisation

In this section, we detail the algorithm for the 2D reduced Vlasov-Poisson system for each time step. We can consider the first order or second ordre splitting scheme

1. The first order in time procedure

- Transport in x_1 over Δt

$$\partial_t \mathbf{w}_i + V_{1,i} \partial_{x_1} \mathbf{w}_i = 0, \quad i = 1 \dots P \tag{6.A.11}$$

- Transport in x_2 over Δt

$$\partial_t \mathbf{w}_i + V_{2,i} \partial_{x_2} \mathbf{w}_i = 0, \quad i = 1 \dots P \tag{6.A.12}$$

- Update E by solving the Poisson equation (FFT) and then solve over Δt the following equation

$$\partial_t \mathbf{w} + M^{-1} B(E) \mathbf{w} = 0 \tag{6.A.13}$$

2. The second order in time procedure

- Transport in x_1 over $\Delta t/2$

$$\partial_t \mathbf{w}_i + V_{1,i} \partial_{x_1} \mathbf{w}_i = 0, \quad i = 1 \dots N_v \tag{6.A.14}$$

- Transport in x_2 over $\Delta t/2$

$$\partial_t \mathbf{w}_i + V_{2,i} \partial_{x_2} \mathbf{w}_i = 0, \quad i = 1 \dots N_v \tag{6.A.15}$$

- Update E by solving the Poisson equation (FFT) and then using the Runge-Kutta second order to solve the source term over Δt

$$\partial_t \mathbf{w} + M^{-1} B(E)\mathbf{w} = 0 \tag{6.A.16}$$

- Transport in x_1 over $\Delta t/2$

$$\partial_t \mathbf{w}_i + V_{1,i}\partial_{x_1}\mathbf{w}_i = 0, \quad i = 1 \dots N_v \tag{6.A.17}$$

- Transport in x_2 over $\Delta t/2$

$$\partial_t \mathbf{w}_i + V_{2,i}\partial_{x_2}\mathbf{w}_i = 0, \quad i = 1 \dots N_v \tag{6.A.18}$$

6.A.3 Conservation properties

We have the following proposition

Proposition 6.A.1. *[89] In the 1D case with a uniform mesh and pediodic boundary conditions, the semi-Lagrangian scheme with cubic spline or Lagrange interpolation conserves exactly the discrete mass ρ and the discrete total current $\mathbf{J}_{tot}$.*

$$\rho = \sum_{i=1\dots N, j=1\dots N_v} \Delta x \Delta v f_i^j,$$

$$\mathbf{J}_{tot} = \sum_{i=1\dots N, j=1\dots N_v} \Delta x \Delta v f_i^j V_j,$$

during the advection in space.

Therefore, in the time splitting algorithm described in section 6.A.2, the advections in space are conservative. Since the advection in velocity is also conservative (see prop. (2.4.8)), the whole algorithm is conservative. Note that, to obtain this conservation properties, it constraints to use 2D Cartesian meshes, uniform in each direction.

Annexe 6.B Parallel code performance

We now study the (strong) scalability of the code. We execute numerical simulations of the semi-Lagrangian scheme on the Curie supercomputer (TGCC, Genci) with the following problem size : $N_v = 65 \times 65$ points in velocity ($d = 2, M = 32$) and $N_{x_1} = N_{x_2} = 256$ points in each space direction. We perform 100 iterations of the code. In table 6.1, we observe that the speed-up is almost optimal.

We can compare these results with the results obtained with the finite volume code in section 5.4.3. We observe that the computation time of the semi-Lagrangian code is two time faster than the finite volume code. This is partly due to the use of the Gauss-Lobatto points for the semi-Lagrangian code instead of the Gauss-Legendre points (for finite-volume code). Indeed, with Gauss-Legendre integration points, at each time step and each spatial node, we have to do a matrix vector mutliplication with the $N_v \times N_v$ matrices $M^1 A^k$. With the Gauss-Lobatto points, there is no such multiplication since the system is diagonalize.

Number of processor units (CPU)	8	16	64	128
computation time (second)	7398	3673	943	491
speed-up (relative)		2.01	3.89	1.92
speed-up		16.11	62.76	120.54

TABLE 6.1 : Computational time and speedup for the 2D **semi-Lagrangian** code. Computed on the supercomputer Curie (TGCC). Parameters : 100 iterations, $N_v = 65^2$ ($d = 2, M = 32$), $N_{x_1} = N_{x_2} = 256$.

Let us compute the efficiency of the code as introduced in 5.4.3, that is the number of advection per processors and per second (in million). Considering the second order splitting discretization (see section 6.A.2), we have 8 sub-steps per time iterations (4 steps for the advection in space and 4 advections in velocity when using the RK2 scheme). Therefore, the efficiency of the code equals : eff = 3.5 with 256×256 grid points in space and 65×65 grid points in velocity. This is twice larger than the efficiency of the finite-volume code but it is still ten times smaller than the efficiency of the full 1D semi-Lagrangian scheme [6]. This confirms that the lack of efficiency comes from the advection in velocity since the order of magnitude of the efficiciency does not depend on the spatial solver (semi-Lagrangian or finite-volume). Note that both code are parallelized with MPI but the finite volume code uses point to point communication (with subdomain decomposition) while the semi-Lagrangian code uses transposition.

Quatrième partie

Application au système Vlasov-Maxwell et au modèle Drift-Kinetic

Chapitre 7

Reduced Vlasov Maxwell simulation

Ce chapitre a été écrit en collaboration avec Philippe Helluy, Laurent Navoret et Anaïs Crestetto. Il fait l'objet d'un article intitulé : Reduced Vlasov-Maxwell simulations.

Reduced Vlasov-Maxwell simulations

Abstract

In this paper we review two different numerical methods for Vlasov-Maxwell simulations. The first method is based on a coupling between a Discontinuous Galerkin (DG) Maxwell solver and a Particle-In-Cell (PIC) Vlasov solver. The second method only uses a DG approach for the Vlasov and Maxwell equations. The Vlasov equation is first reduced to a space-only hyperbolic system thanks to the finite element method. The two numerical methods are implemented using OpenCL in order to achieve high performance on recent Graphic Processing Units (GPU).

Introduction

The Maxwell-Vlasov system is a fundamental model in physics. It can be applied to plasma simulations, charged particles beam, astrophysics, *etc.* The unknowns are the electromagnetic field, solution of the Maxwell equations and the distribution function, solution of the Vlasov equation. The two systems of equations are coupled because the motion of particles generates an electric current at the right hand side of the Maxwell equations, while the electromagnetic field accelerates the particles in the Vlasov equation. The Maxwell equations are a system of linear hyperbolic equations. Today, they are routinely solved in industrial applications with commercial software. Several numerical methods exist : finite difference, finite elements. In this paper, we will use a more and more popular method for solving the Maxwell equations : the Discontinuous Galerkin (DG) finite element approach. This method is presented in many works. We refer for instance to [50, 17, 22, 63, 24].

The Vlasov equation is a rather simple transport equation, but set in a six-dimensional (x, v) space-velocity phase space. This leads to very heavy computations. The most popular method for solving the Vlasov equation is thus the Particle-In-Cell (PIC) method of [14],

because this is one of the less expansive. It consists in distributing random particles with random velocities in the computational domain. The particles are then pushed by the electromagnetic field. They are also deposited on the Maxwell finite element mesh in order to generate a current in the right hand side of the Maxwell equations.

The PIC method is very easy to implement. However it is subject to numerical noise. It leads also to other issues : smoothing, charge conservation errors, energy conservation errors. We will also see in this paper that the PIC method is rather difficult to parallelize.

Therefore, while they are certainly more memory consuming, Eulerian approaches, which solve the Vlasov equation directly on a phase-space grid, are more and more investigated.

In this paper, we first review some aspects of the DG and PIC methods, which are very important for obtaining a robust and precise approximation. We then describe a parallelization of the full Vlasov-Maxwell coupling on recent Graphic Processing Units (GPU) with the OpenCL framework.

We will see that the DG method is very well adapted to parallelization. The PIC method is more difficult to efficiently parallelize. Therefore, we will present a recent approach, the reduction method, which allows approximating the Vlasov equation also by a DG solver.

7.1 Vlasov-Maxwell equations

7.1.1 Maxwell equations

In this paper, we consider the two-dimensional Maxwell equations in Transverse Magnetic (TM) mode. The unknowns depend on the space variable $x = (x_1, x_2) \in \mathbb{R}^2$ and the time $t \geq 0$. They are the electric field

$$E = (E_1, E_2, 0)^T,$$

and the magnetic field

$$H = (0, 0, H_3)^T.$$

The current

$$j = (j_1, j_2, 0)$$

is supposed to be given.

We then write the unknowns and the source term in a vector form

$$\mathbf{w} = (E_1, E_2, H_3)^T, \quad S = (-j_1, -j_2, 0)^T.$$

In this way, the Maxwell equations read as a linear first order hyperbolic system

$$\partial_t \mathbf{w} + A^i \partial_i \mathbf{w} = S, \tag{7.1.1}$$

where we use the Einstein convention (sum on repeated indices)

$$A^i \partial_i = A^1 \partial_1 + A^2 \partial_2,$$

and the notation

$$\partial_i = \frac{\partial}{\partial x_i}.$$

In the first order differential system (7.1.1) the matrices A^i are given by

$$A^1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}$$

(the equations are written under a dimensionless form where the speed of light $c = 1$). Let now $n = (n_1, n_2)^T \in \mathbb{R}^2$. We can also define the flux of the Maxwell equations

$$f(\mathbf{w}, n) = A^i n_i \mathbf{w}.$$

If we define

$$n^1 = (1, 0)^T, \quad n^2 = (0, 1)^T,$$

the Maxwell equations can also be written under the more general conservative form

$$\partial_t \mathbf{w} + \partial_i f(\mathbf{w}, n^i) = S. \tag{7.1.2}$$

7.1.2 Vlasov equation

We consider now the motion of N particles of mass m and charge q in the electromagnetic field. The particles are labeled by an index k, $1 \leq k \leq N$. The position of particle k at time t is $x^k(t)$ and its velocity is

$$\dot{x}^k(t) = \frac{d}{dt} x^k(t).$$

For $x = (x_1, x_2) \in \mathbb{R}^2$ and $v = (v_1, v_2) \in \mathbb{R}^2$, the distribution function of particles is defined by

$$f(x, v, t) = \sum_k \omega_k \delta(x - x^k(t)) \delta(v - \dot{x}^k(t)),$$

where δ denotes the Dirac measure on $\mathbb{R}^2$ and ω_k is the weight of particle k. The electric current generated by the particles motion is given by

$$j(x, t) = \int_v f(x, v, t) v dv = \sum_k \omega_k \delta(x - x^k(t)) \dot{x}^k(t). \tag{7.1.3}$$

The particle acceleration is given by the relativistic equation of motion

$$\dot{x} = v, \quad \ddot{x} = \mu \frac{q}{m} (E + v \wedge H), \tag{7.1.4}$$

with

$$E = (E_1, E_2, 0)^T, \quad H = (0, 0, H_3)^T,$$

$$\mu = \mu(v) = (1 - v \cdot v)^{1/2} (Id - v \otimes v). \tag{7.1.5}$$

where $Id - v \otimes v$ denotes the projection to the orthogonal space to v. We can also write the acceleration with the notation

$$\ddot{x} = \mu(\dot{x}) a(x, \dot{x}, t), \quad a(x, v, t) = \frac{q}{m} (E_1(x, t) + v_2 H_3(x, t), E_2(x, t) - v_1 H_3(x, t)).$$

It is then possible to prove that, in the weak sense, the distribution function satisfies the relativistic Vlasov equation written under a conservative form

$$\partial_t f + \nabla_x \cdot (fv) + (\mu(v)\, a) \cdot \nabla_v f = 0. \tag{7.1.6}$$

When the particle velocity is small compared to the speed of light $c = 1$, we can use the Galilean approximation

$$\mu(v) \simeq 1. \tag{7.1.7}$$

In addition, we introduce the admissible velocity disk

$$D = \left\{ v \in \mathbb{R}^2, v \cdot v < 1 \right\}.$$

Let us also observe that on the boundary of the velocity disk, we have

$$v \in \partial D \Rightarrow \mu(v) = 0.$$

This is a nice property. Indeed, the Vlasov equation (7.1.6) is a transport equation written in the (x, v) velocity phase space. The phase space transport velocity is

$$V = (v, \mu a) \in \mathbb{R}^4$$

The component of V in the velocity direction v is the acceleration μa, which vanishes on ∂D. Thus we will have no boundary condition to apply on ∂D, or more precisely, if at the initial time $f = 0$ on ∂D, then this property is maintained at all times.

7.1.3 Divergence cleaning

The charge $\rho(x, t)$ is defined by

$$\rho(x, t) = \int_v f(x, v, t) dv.$$

Integrating the Vlasov equation with respect to v, we obtain the charge conservation

$$\partial_t \rho + \nabla_x \cdot j = 0. \tag{7.1.8}$$

If at the initial time $t = 0$ the Gauss law is satisfied

$$\nabla_x \cdot E(x, t = 0) = \rho(x, t = 0),$$

then, using the charge conservation (7.1.8) and the Maxwell equations (7.1.1) we deduce the Gauss law at all times

$$\nabla_x \cdot E = \rho. \tag{7.1.9}$$

The Gauss law is thus a consequence of the Vlasov-Maxwell equations. However, depending on the numerical scheme, it might not be well satisfied by the numerical approximation. A practical tool for improving the numerical accuracy is to use a divergence cleaning technique of [75]. The divergence cleaning consists in considering an additional artificial unknown $\phi(x, t)$ in the Maxwell equations, which satisfies

$$\partial_t \phi + \chi \nabla_x \cdot E = \chi \rho.$$

The constant parameter $\chi > 0$ represents the speed at which the divergence errors are propagated to the boundaries of the computational domain. In addition, the time derivative of the electric field $\partial_t E$ is replaced by $\partial_t E + \chi \nabla_x \phi$ in the Maxwell equation. We thus obtain a new vector of unknowns

$$\mathbf{w} = (E_1, E_2, H_3, \phi)^T.$$

The divergence cleaning model still reads as an hyperbolic system (7.1.1) but the matrices are now

$$A^1 = \begin{pmatrix} 0 & 0 & 0 & \chi \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ \chi & 0 & 0 & 0 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & \chi \\ -1 & 0 & 0 & 0 \\ 0 & \chi & 0 & 0 \end{pmatrix},$$

and the source term becomes

$$S = (-j_1, -j_2, 0, \chi\rho)^T. \tag{7.1.10}$$

An important feature of this extended Maxwell system is that we recover exactly the Maxwell equations when ϕ is constant. Thus, with adequate boundary conditions we introduce only a numerical stabilization of the divergence errors without additional numerical errors, even for small values of χ. For more details on the divergence cleaning system, we refer to [75, 25, 24, 44].

7.1.4 Boundary conditions

The Vlasov-Maxwell equations (7.1.2), (7.1.6) need to be supplemented by conditions at the boundary of the computational domain $\Omega \times D$.

Maxwell boundary conditions

Stable boundary conditions for the classic Maxwell equations have been extensively studied. These conditions are properly generalized to the divergence cleaning model in a few papers : [44, 25]. We recall here briefly the general theory.

We consider local boundary conditions in the form

$$M(n)(\mathbf{w} - \mathbf{w}^{\text{inc}}) = 0, \tag{7.1.11}$$

where $n = (n_1, n_2, 0)$ is the normal outward vector on $\partial\Omega$ and W^{inc} a given incident boundary electromagnetic field (which can be zero). The boundary condition has to be chosen in such way that the Maxwell operator in space x is maximal positive. Stability conditions are given by the Lax-Philips theory ([66, 79, 17, 50, 25, 44]) which requires

- $\frac{1}{2} A^i n_i + M(n) \geq 0$.
- $\dim \ker A^i n_i^- = \dim \ker M(n)$.

It is possible to prove that the following boundary conditions (and associated M matrices) satisfy the Lax-Philips stability conditions

- Generalized "metal" :

$$n \times E = 0, \quad \phi = \lambda E \cdot n, \quad \lambda \geq 0 \tag{7.1.12}$$

- Generalized "Silver-muller", $M = -A^i n_i^-$:

$$\begin{aligned} H_3 - n_1 E_2 + n_2 E_1 &= H_3^{\text{inc}} - n_1 E_2^{\text{inc}} + n_2 E_1^{\text{inc}}, \\ E \cdot n - \phi &= E^{\text{inc}} \cdot n. \end{aligned} \tag{7.1.13}$$

The generalized metal condition is compatible with the original Maxwell system only when $\lambda = 0$ (because we need to have $\phi = 0$). However, a small $\lambda > 0$ can be interesting for numerical reasons, because it introduces a slight energy damping.

We would like to emphasize that the respect of the Lax-Philips stability condition is absolutely crucial for obtaining stable and precise numerical results, especially when the DG solver is coupled to a PIC solver.

Vlasov boundary conditions

As seen in Section 7.1.2, no boundary condition is required on the boundary $\Omega \times \partial D$ of the velocity domain. We thus consider the case $x \in \partial\Omega$ and $v \in D$. The outward normal vector on $\partial\Omega$ is still noted $n(x)$.

Inflow condition It is natural to impose the value of the distribution function f only at inflow ([58]). Introducing the notations

$$\alpha^+ = \max(\alpha, 0), \quad \alpha^- = \min(\alpha, 0),$$

the inflow condition can be written

$$(v \cdot n(x))^- f(x, v, t) = (v \cdot n)^- f_0(x, v, t),$$

in such a way that when $v \cdot n > 0$, no condition is imposed on f.

Child-Langmuir condition A more subtle boundary condition is the Child-Langmuir current condition. This condition is useful at a particles emitting boundary because it allows creating just the quantity of charges that cancels the normal component of the electric field. For the moment we do not know how to express in a rigorous mathematical way the Child-Langmuir condition. We just describe the practical computation.

The electrons are emitted at the cathode if the normal electric field is strong enough, until it cancels (Child-Langmuir law).

More precisely, we use the following algorithm for a discretization cell L that touches the cathode boundary Γ_C :

- if $E \cdot n < 0$ on $\partial L \cap \Gamma_C$ then compute

$$\delta_L = \rho_L - \int_{\partial L \setminus \Gamma_C} E \cdot n$$

  where $\rho_L = \sum_{x^k \in L} \omega_k$ (charge in the cell L).

- if $\delta_L < 0$, create n_e random particles in the cell L with weights δ_L / n_e.

7.2 Discontinuous Galerkin method

7.2.1 Weak upwind DG formulation

The Discontinuous Galerkin (in short DG) approximation is a more and more popular method for approximating hyperbolic systems of conservation laws (see, among many others, [17, 24, 25, 22, 63, 68]).

We consider a mesh of the domain Ω. In each cell L the fields $W(x,t)$ are approximated by second order polynomial in x. We denote by $P_2(\mathbb{R}^2)$ a linear space of second order polym.Mesh(c12,c13,c0,c15, c16,c17,c4,c18, c19,c20,c9,c8) ; nomial in $x = (x_1, x_2)$. In practice, we use $P_2(\mathbb{R}^2) = \text{span}\{1, x_1, x_2, x_1 \cdot x_2, x_1^2, x_2^2, x_1^2 \cdot x_2, x_1 \cdot x_2^2\}$ because with this choice, we have $\dim P_2(\mathbb{R}^2) = 8$ which is well suited to GPU optimizations. Then

$$\mathbf{w}_L(x,t) = \sum_j w_{L,j}(t)\psi_{L,j}(x), \quad \{\psi_{L,j}\} \text{ basis of } P_2(\mathbb{R}^2)^4. \tag{7.2.1}$$

The DG upwind weak formulation ([68, 58, 50]) consists in finding the basis components $w_{L,j}(t)$ in each cell L such that for all test function $\psi_L \in P_2(\mathbb{R}^2)^3$

$$\begin{aligned}\int_L \partial_t \mathbf{w}_L \cdot \psi_L - \int_L \mathbf{w}_L \cdot A^i \partial_i \psi_L + \int_{\partial L \cap \Omega} (A^i n_i^+ \mathbf{w}_L + A^i n_i^- \mathbf{w}_R) \cdot \psi_L \\ + \int_{\partial L \cap \Omega} (M + A^i n_i)\mathbf{w}_L \cdot \psi_L = \int_L S \cdot \psi_L + \int_{\partial L \cap \Omega} M \mathbf{w}^{\text{inc}} \cdot \psi_L\end{aligned} \tag{7.2.2}$$

where we denote by n the normal vector on ∂L oriented from the cell L to the neighboring cells R and

$$x^+ = \max(0, x), \quad x^- = \min(0, x).$$

The DG formulation is a generalization of the finite volume method. It relies on the standard upwind numerical flux for linear hyperbolic systems

$$f(\mathbf{w}_L, \mathbf{w}_R, n) = A^i n_i^+ \mathbf{w}_L + A^i n_i^- \mathbf{w}_R.$$

Finally, (7.2.2) is equivalent to a system of ordinary differential equations for the $w_{L,j}(t)$.

We do not give all the details of the implementations, but for our application, the main lines are :

- The cells L are quadratic curved "quadrilaterals".
- The components of the basis functions $\psi_{L,j}$ are orthonormal polynomials on the cell L when the cell is a parallelogram : we use a modal basis defined directly in the physical space. We do not rely on a reference element. Thanks to this choice we will not have to invert a geometric transformation for computing the fields at the particles.
- We use a high order numerical integration (16 Gauss-Legendre quadrature points in the cells and 4 points on each edge).

For more details we refer to [25].

7.2.2 GPU parallelization

The DG method can be parallelized efficiently on Graphic Processing Unit (GPU) ([63]). GPUs are recent computing devices that have proven to be very efficient for performing computations on data that can be regularly organized into the GPU memory.

GPUs are not as easy to program as classic processors. CUDA is a well known environment for programming NVIDIA GPUs. OpenCL is another framework for programming various multicore devices, including GPUs or CPUs of several vendors. OpenCL means "Open Computing Language". It includes a library of C functions, called from the host, in order to drive the GPU and a C-like language for writing the programs that will be executed on the processors of the multicore accelerator. The specification is managed by the Khronos Group [61].

OpenCL proposes a rather general abstraction that works well for various multicore SIMD hardware. Very schematically, an OpenCL device possesses a few gigabytes of global memory and is made of a few tens of Computing Units (CU). Each CU contains a few processors called Processing Elements (PE), and a small cache memory of a few kilobytes. The same program, called a kernel, can be executed on all the Processing Elements at the same time. The PEs have a very fast access to the cache memory of their CU. The PEs have also an efficient access to the GPU global memory if they read or write to adjoining memory locations. For non-regular computations, a classic strategy is thus first to fetch a tile of data into the CU cache, then perform the computations with fast access to the cache. When the computations are finished, they are copied back, in a regular way to the global memory. A special behavior of OpenCL devices makes the GPU programming rather complicated : if two processors try to write at the same memory location at the same time, only one will succeed... This has to be kept in mind, for instance in the flux collecting algorithm or when computing the current created by the particles in the same cell L.

We have written an OpenCL implementation of the previous DG formulation. Our programming strategy is described in details in [25]. We just recall here the principal points :

- Initialization : we compute and invert the local mass matrices on the CPU. We send (all) the data to the GPU.

- First pass of each time step : we associate to each Gauss point of each edge one processor. We compute the flux at the Gauss points and store it into global memory.

- Second pass : we associate to each basis function of each element a processor. We compute the time derivative of the $w_{L,j}$ using the DG weak formulation, the previously computed fluxes and the stored inverted mass matrices. The separation into two passes with parallelism redistribution allows avoiding concurrent writing operations.

- Time integration : we use a simple second order Runge-Kutta scheme.

According to some benchmark that we have performed, we observed a spectacular efficiency of the GPU implementation. Compared to a single core implementation on a traditional CPU we have observed that the GPU implementation is 50-100 times faster ([25]).

7.2.3 GPU implementation

We give some details on the GPU implementation for one time step. More informations are given in [25].

Particles motion

- Emission : we use the algorithm described in the last paragraph of Section 7.1.4. The random positions are given by independent van der Corput sequences.
- Particle acceleration : at each time step we associate one processor to each particle. We move the particle and find its new cell location. Our algorithm works on an unstructured grid, but for efficiency, we assume that during one time step the particle cannot cross more than one cell layer. This condition imposes a CFL condition that is not constraining compared to the DG solver CFL condition.

Current

This is the most subtle part of the GPU algorithm because we have to avoid concurrent memory write operations. This difficulty has been already addressed by several authors (see, for instance [9]). The most efficient solutions generally rely on a particles sorting pass at each time iteration.

- We thus first sort the list of particles according to their cell numbers. For this sorting, we use a GPU-optimized radix sort algorithm of [51]. Then, it is easy to know how many particles are in each cell.
- We can also sort the cells list according to the number of particles inside the cells (optional).
- We associate to each cell a processor. Then for each cell it is possible to loop on its particles in order to compute their contributions to the current

$$\int_L S \cdot \psi_L = \sum_{x^k \in L} \omega_k (-\dot{x}_1^k(t), -\dot{x}_2^k(t), 0, \chi)^T \cdot \psi_L(x^k(t)).$$

- Thanks to the second optional sorting, neighboring processors treat approximately the same number of particles and in this way they do not wait too much for each other. In some computations, this can increase the efficiency.

7.3 GPU numerical experiments

GPU programming is complex and time consuming. We thus expect at least high speedups of the implementation. For measuring the efficiency, we have compared in [25] a sequential and a GPU implementation of the DG Maxwell solver. In this case, without particles, we have observed speedups of the order of $50 - 100$. When we couple the Vlasov and the PIC solver we have still good speedups of the order $5 - 10$, but we clearly loose one order of magnitude in the computational time. This can be explained by several reasons :

- Particle sorting is time consuming and require non-coalescing memory accesses.
- The particles are sorted with an indirection array. The current computation thus also requires random memory accesses.
- Particles sorting and current evaluation take approximately the same time. While the particle solver is called only every ten iterations of the DG solver, the DG solver represents only 15% of the computation time. Recall that we do not use particles smoothing and that we need high values of χ for performing a good divergence cleaning (typically, we take $\chi \simeq 10$). Finally, this is not so expensive, because the DG solver is much cheaper than the PIC solver.

We now present rapidly several numerical experiments.

7.3.1 Child-Langmuir current

In our first example we try to compute numerically a stationary solution that can be expressed analytically. We will see that a high value of χ is necessary. If χ is too small, the scheme does not correct the divergence errors efficiently. Maybe that a smoothing of the particles would permit to diminish χ.

We consider a planar diode $\Omega = [0, L_x] \times [0, L_y]$ with a cathode $C = \{x = 0\} \times [0, L_y]$ and an exit boundary $A = \{x = L_x\} \times [0, L_y]$. We consider a metal boundary condition (7.4.1) at the anode $x = 0$ and we apply the exact solution with an inhomogeneous Silver-muler condition at $x = L_x$. At this point, the "incident" field is defined by

$$(E_1, E_2, H_3, \psi)^{\text{inc}} = (-1, 0, 0, 0)^T . \tag{7.3.1}$$

We apply periodic conditions at $y = 0$ and $y = L_y$.

We represent E_x on Figure 7.1 at times $t = 1$ and $t = 5$. χ is taken equal to 5 and we move the particles every 25 time steps in order to let the divergence correction act. Smaller values of χ give inaccurate results ([24]). We see the emission (there is no particle at $t = 0$) and the motion of electrons from the cathode to the anode. We can also remark that on the cathode $E_x = E \cdot n \approx 0$, which is the searched Child-Langmuir condition.

For such a planar diode in Cartesian geometry, the Child-Langmuir current J_{CL} on the anode for a given potential drop V_0 between the cathode and the anode verifies

$$J_{CL}(L_x) = \frac{4}{9{L_x}^2}\sqrt{\frac{2 \mid q \mid}{m}} V_0^{\frac{3}{2}}, \tag{7.3.2}$$

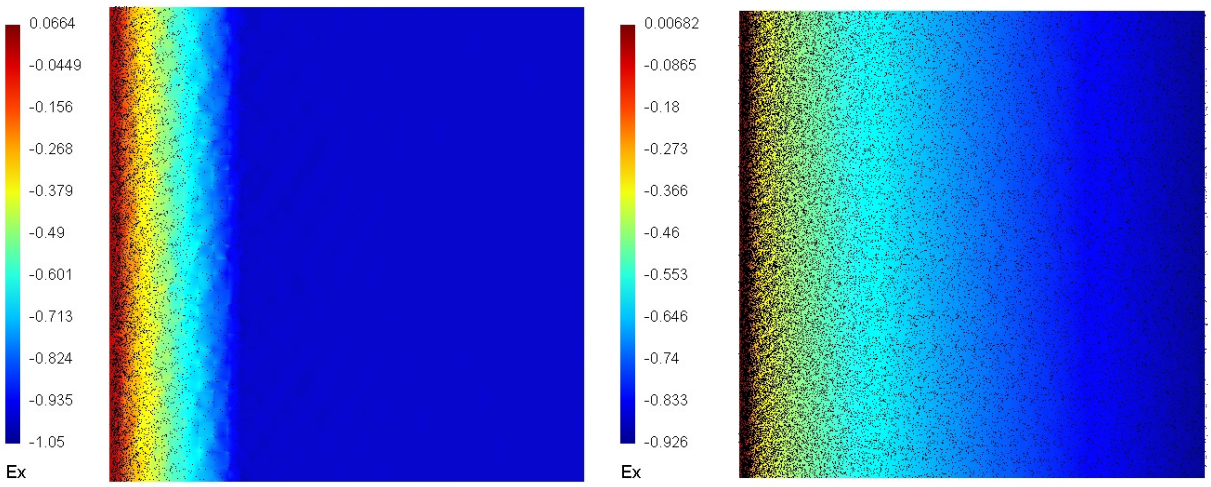
where

$$J_{CL} = \int_A j \cdot n \tag{7.3.3}$$

and

$$V_0 = V(x = L_x, y_0) - V(x = 0, y_0) = \int_0^{L_x} \partial_x V(x, y_0)\, dx = -\int_0^{L_x} E_x(x, y_0)\, dx, \tag{7.3.4}$$

$y_0 \in [0, L_y]$ and V denoting the scalar potential such that $E = -\nabla_x V$. The computed potential drop is given as a function of time on Figure 7.2 (left). It tends to the value denoted by V_0. The corresponding theoretical current J_{CL} is also given on Figure 7.2 (right) and compared to the computed one.



$t = 1$ $t = 5$

FIGURE 7.1 : Planar diode case : E_x at times $t = 1$ with 8918 particles (left), and $t = 5$ with 44133 particles (right), 1024 elements.

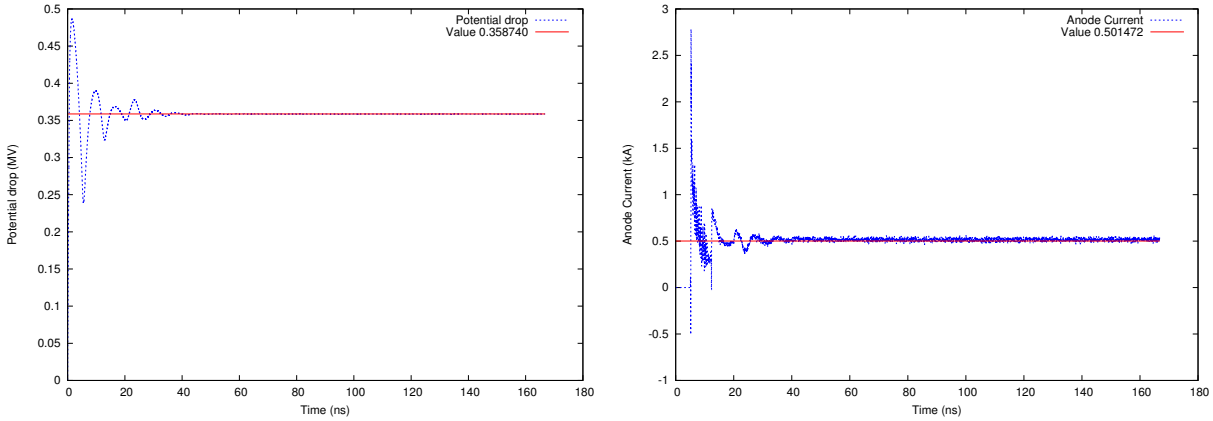


FIGURE 7.2 : Child-Langmuir law : potential drop (left) and Child-Langmuir current (right), computed values (blue) compared to theoretical ones (red).

7.3.2 X-Ray generator

With our code, we have also been able to simulate an electrons emitting diode. This device is used for producing x-rays, when the electrons hit the anode. The axisymmetric geometry of the diode and the mesh of the computational domain are represented on Figure 7.3. At time $t = 0$, an electromagnetic wave is entering at the left of the computational domain. At this boundary Γ_s we apply an inhomogeneous Silver-Muller boundary condition. At the cathode and the anode, we apply metal boundary conditions. The electrons are emitted at the cathode. The rotational symmetry implies additional geometric terms in the Maxwell equations and in the particles weights (see [25]). In addition there is no boundary condition to apply on the rotation axis, because the numerical flux in the Galerkin Discontinuous method simply cancels (it is multiplied by the distance to the axis).

We represent the radial component of the electric field and the particles at time $t = 0.22$

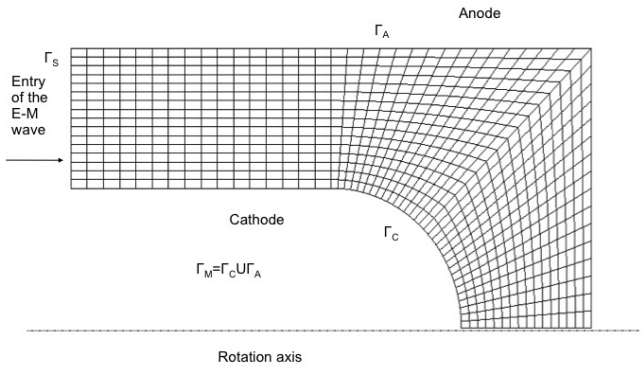


FIGURE 7.3 : Diode geometry

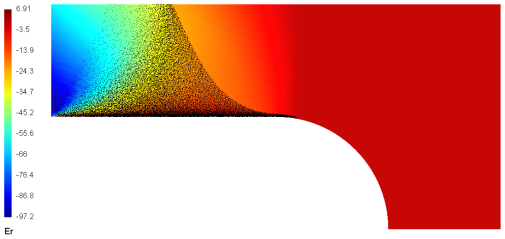


FIGURE 7.4 : Electron emission at dimensionless time $t = 0.22$. The length of the computational domain $L = 0.4$ and the dimensionless speed of light $c = 1$.

on Figure 7.4.

This numerical simulation has been awarded a prize at the AMD OpenCL competition in 2011. See

`http://developer.amd.com/events/amd-opencl-coding-competition-2/`

7.4 Reduced modeling

The conclusion of our Vlasov-Maxwell PIC-DG experience is that it is finally possible to achieve interesting accelerations on GPU. However, the implementation is complex. While the PIC sequential implementation is straightforward, its parallelization requires particles sorting, random memory accesses, *etc.* And in the end most of the GPU time is spent in the PIC algorithm while the DG solver is very efficient.

We would thus like to present another approach where we use a unified Eulerian DG solver for the Maxwell and the Vlasov equations. When coding our DG solver, we have tried to be as generic as possible. For instance the physical model is explicit only in a few parts of the code : in the numerical flux, the source terms and the boundary terms. But generally, the whole DG algorithm is not aware of the underlying physical model. We will show how to rewrite the Vlasov equation in order to obtain an augmented hyperbolic system, written only in (x, t) that can thus be solved by the generic DG solver. The resulting model is called

the reduced Vlasov-Maxwell model.

7.4.1 Velocity expansion

The Vlasov equation is a transport equation written in the (x, v) space. The objective of reduced modeling is to rewrite the Vlasov equation in order to obtain a hyperbolic system of conservation laws, but set only in the x space. In this way it is possible to reuse a generic DG solver. For this purpose, we consider a finite number of continuous basis functions depending on the velocity

$$v \in D \mapsto \varphi_i(v), \quad i = 1 \cdots P.$$

We suppose that

$$v \in \partial D \Rightarrow \varphi_i(v) = 0. \tag{7.4.1}$$

We expand the distribution function in this basis

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

We insert this representation into the Vlasov equation (7.1.6), multiply by φ_i and integrate on the velocity domain D. We also integrate by parts the acceleration term, and using (7.4.1), we obtain ([53])

$$\int_v \varphi_i \varphi_j \partial_t f^j + \int_v \varphi_i \varphi_j v^k \partial_k f^j - \int_v \mu a \cdot \nabla_v \varphi_i \varphi_j f^j = 0.$$

We can then define the following $P \times P$ matrices

$$M_{i,j} = \int_v \varphi_i \varphi_j, \quad A^k_{i,j} = \int_v \varphi_i \varphi_j v^k, \quad B_{i,j} = -\int_v \mu a \cdot \nabla_v \varphi_i \varphi_j, \tag{7.4.2}$$

and the Vlasov equation can be rewritten in the reduced form

$$M\partial_t \mathbf{w} + A^k \partial_k \mathbf{w} + B\mathbf{w} = 0, \tag{7.4.3}$$

or also

$$\partial_t \mathbf{w} + M^{-1} A^k \partial_k \mathbf{w} + M^{-1} B\mathbf{w} = 0, \tag{7.4.4}$$

where

$$\mathbf{w} = (f^1, \cdots, f^P)^T. \tag{7.4.5}$$

The form (7.4.4) is called the reduced Vlasov equation. It is a first order hyperbolic system of conservation laws ([53]) that can be solved by a standard DG solver. However, for practical reasons it is important to provide an efficient choice of basis functions φ_i. A good choice ensures a small number of basis functions P and that the matrices M, A^k and B are sparse. We detail now such a basis and also an adequate choice of numerical quadrature that will lead to diagonal matrices M and A^k.

7.4.2 Finite element basis with nodal integration

We consider a nodal finite element interpolation in the velocity space with curved "quadrilaterals". In addition, the nodal points will coincide with Gauss-Lobatto quadrature points. In this way we obtain several interesting properties of the basis functions. Let us give now more details.

We choose first a degree $d \geq 1$ of polynomial approximation. We can associate to this degree $d+1$ Gauss-Lobatto points on the interval $[0,1]$, $\xi_1 = 0 < \xi_2 < \cdots < \xi_{d+1} = 1$. We consider also integration weights $\omega_1 \cdots \omega_d$. The Gauss-Lobatto integration rule

$$\int_0^1 Q(\xi)d\xi \simeq \sum_{i=1}^{d+1} \omega_i Q(\xi_i),$$

is then exact if Q is a polynomial of degree $\leq 2d-1$. We also consider the Lagrange polynomials L_i associated to the Gauss-Lobatto subdivision. The polynomial L_i, $i = 1 \cdots d+1$ is of degree d and satisfies

$$L_i(\xi_j) = \delta_{ij},$$

where δ_{ij} is the Kronecker delta.

We construct now a mesh of the velocity disk D. The mesh is made of nodes V_k, $k = 1 \cdots P_D$, $P_D > P$. Each node V_k for $k = 1 \cdots P$ is associated to a basis function φ_k. We also suppose that the nodes $V_{P+1} \cdots V_{P_D}$ are on the boundary ∂D in such a way that they are not associated to basis functions φ_k. The nodes are associated to elements Λ_k, $k = 1 \cdots K$. Each element is a curved "quadrilateral" and owns $(d+1)^2$ nodes. An example of such a mesh with degree $d = 3$, $K = 80$ elements, $P_D = 745$ nodes and $P = 697$ interior nodes is given on Figure 7.5 .

As it is traditional in the finite element method, we consider a local and a global numbering of the nodes of element Λ_k. The local node l, $l = 1 \cdots (d+1)^2$ of element Λ_k is also noted

$$V_{k,l} = V_{\kappa(k,l)},$$

where $\kappa(k,l)$ is the $K \times (d+1)^2$ connectivity array of the finite element mesh. Each element Λ_k is obtained from a geometrical transformation τ_k that maps the square $\hat{\Lambda} =]0,1[\times]0,1[$ on element Λ_k. The geometrical transformation is defined as follows. First, we consider $(d+1)^2$ reference nodes

$$\hat{V}_l = (\xi_i, \xi_j), \quad \text{with } l = (i-1)(d+1) + j, \quad 1 \leq i, j \leq d+1.$$

The reference nodes are the two-dimensional Gauss-Lobatto points of the reference element. Reference node $\hat{V}_l$ is also associated to an integration weight

$$\hat{\omega}_l = \omega_i \omega_j \text{ with } l = (i-1)(d+1) + j, \quad 1 \leq i, j \leq d+1. \tag{7.4.6}$$

To each reference node, we associate a reference basis function, which is a tensor product of Lagrange polynomials

$$\hat{\varphi}_l(\xi, \eta) = L_i(\xi) L_j(\eta), \quad \text{with } l = (i-1)(d+1) + j, \quad 1 \leq i, j \leq d+1.$$

We can check that

$$\hat{\varphi}_l(\hat{V}_m) = \delta_{lm}.$$

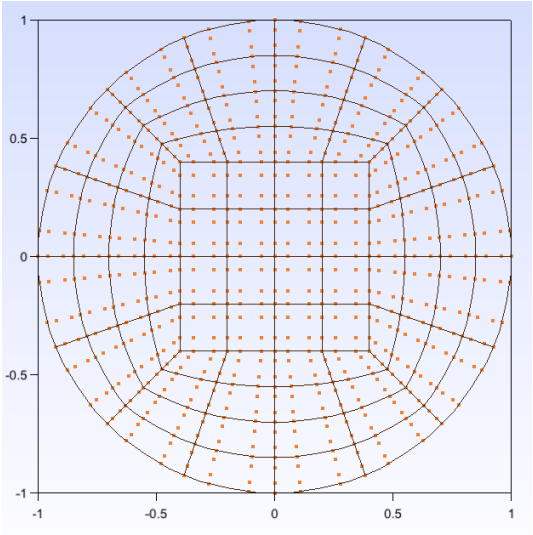


FIGURE 7.5 : A mesh of the velocity disk D with degree $d = 3$, $K = 80$ elements (in black), $P_D = 745$ Gauss-Lobatto nodes (red points) and $P = 697$ interior nodes. The distribution function cancels on the boundary and is thus computed only at the interior velocity points. Such a mesh leads to very heavy computations because we have to solve in space and time a hyperbolic system with $P = 697$ (Vlasov) $+3$ (Maxwell) $= 700$ components.

Then, the geometric transformation is defined by

$$\tau_k(\xi, \eta) = \sum_{l=1}^{(d+1)^2} \hat{\varphi}_l(\xi, \eta) V_{k,l},$$

in such a way that

$$\tau_k(\hat{V}_l) = V_{k,l}.$$

We also suppose that the node V_i are defined in such a way that τ_k is invertible and also a direct transformation

$$\det \tau_k^{'} > 0.$$

We have now all the pieces to construct the basis functions. Let $i = 1 \cdots P$ and $v \in D$, then necessarily, $v \in \Lambda_k$ for some $k = 1 \cdots K$[1]. We then have two possibilities :

1. Node V_i is not a node of element Λ_k then

$$\varphi_i(v) = 0. \tag{7.4.7}$$

2. Node V_i is a node of element Λ_k. It means that $i = \kappa(k, l)$ for some $l = 1 \cdots (d+1)^2$. Then

$$\varphi_i(v) = \hat{\varphi}_l(\hat{v}), \text{ with } v = \tau_k(\hat{v}). \tag{7.4.8}$$

In this case, we can also compute the gradient of the basis function

$$\nabla_v \varphi_i(v) = (\tau_k'(\hat{v})^T)^{-1} \nabla_{\hat{v}} \hat{\varphi}_l(\hat{v}), \quad v = \tau_k(\hat{v}). \tag{7.4.9}$$

[1]We assume that the elements Λ_k are disjoint open sets and that $\overline{\cup \Lambda_k} = \overline{D}$. Of course, this cannot be exactly true because τ_k is a polynomial transformation. We neglect in our presentation the error made in the approximation of D.

From the previous definitions, we obtain basis functions that are continuous on D. In addition, they satisfy the interpolation property

$$\varphi_i(V_j) = \delta_{ij}, \quad 1 \leq i, j \leq P.$$

This interpolation property ensures that the components of w in (7.4.5) are simply approximations of the distribution function at the Gauss-Lobatto points V_i

$$f^i(x,t) \simeq f(x, V_i, t).$$

We can thus also use the convention that on the boundary nodes

$$f^i(x,t) = 0 \text{ if } P+1 \leq i \leq P_D.$$

For computing the matrices in (7.4.2) we use the Gauss-Lobatto integration rule. For a given function h defined on D the integral is first split into elementary integrals

$$\int_D h(v)dv = \sum_{k=1}^{K} \int_{\Lambda_k} h(v)dv,$$

and then (using definition (7.4.6))

$$\begin{aligned} \int_{\Lambda_k} h(v)dv &= \int_{\hat{\Lambda}} h(\tau_k(\xi,\eta)) \det \tau_k'(\xi,\eta) \\ &\simeq \sum_{l=1}^{(d+1)^2} \hat{\omega}_l \det \tau_k'(\hat{V}_l) h(\tau_k(\hat{V}_l)) \\ &= \sum_{l=1}^{(d+1)^2} \omega_{k,l} h(V_{k,l}). \end{aligned} \tag{7.4.10}$$

Using the quadrature rule (7.4.10) and formula (7.4.9) for computing the gradient of the basis function we can practically compute the matrices in (7.4.2). Our choice of integration points does not ensure exact integration for$M$, A^k and B. However, in the sequel, we use the same notation for the exact and approximate matrices. With our choice of quadrature points we obtain that M and A^k are diagonal matrices. More precisely, we have

$$M_{ii} = \sum_{i=\kappa(k,l)} \omega_{k,l}, \tag{7.4.11}$$

and

$$A^k_{ii} = M_{ii} V^k_i, \quad V_i = (V^1_i, V^2_i).$$

These computations show that the components of the vector w satisfy a set of coupled transport equations

$$\partial_t f^i + V_i \cdot \nabla_x f^i + \Sigma^i(\mathbf{w}) = 0, \quad i = 1 \cdots P. \tag{7.4.12}$$

In the vector form, the coupling source term is given by

$$\Sigma(\mathbf{w}) = M^{-1} B \mathbf{w}.$$

More precisely, after expanding the computations, the coupling source term becomes

$$\Sigma^i(\mathbf{w}) = \frac{-1}{M_{ii}} \sum_{\overline{\Lambda_k} \ni V_i} \sum_{l=1}^{(d+1)^2} \omega_{k,l} f^{\kappa(k,l)} \mu(V_{k,l}) a(\cdot, V_{k,l}, \cdot) \cdot \nabla_v \varphi_i(V_{k,l}), \tag{7.4.13}$$

where we recall that the gradient of the basis function φ_i is given by (7.4.9).

Remark : In practice, we can compute M_{ii} and Σ^i efficiently by a classic finite element assembly procedure : we loop on the elements, compute the elementary contributions and distribute them into the global (M_{ii}) or (Σ^i) vectors.

7.4.3 Unified expression of the reduced Vlasov-Maxwell model

We are now in a position to write in a unified way the Maxwell and reduced Vlasov system. We extend the original vector W of (7.1.1) in the following way

$$\mathbf{w} = (E_1, E_2, H_3, \phi, f^1 \cdots f^P)^T.$$

We then obtain a hyperbolic system in the form (7.1.1) where the new matrices A^k are block diagonal. The blocks are constructed with the matrices A^k of Section 7.1.3 and Section 7.4.2. The source term of the new system is also assembled from the source terms (7.1.10) and (7.5) of Section 7.1.3 and Section 7.4.2

$$S(\mathbf{w}) = (j_1, j_2, 0, \chi\rho, \Sigma^T)^T. \tag{7.4.14}$$

For computing the current and the charge here, we again use the Gauss-Lobatto quadrature (7.4.10) in the velocity space

$$\rho = \sum_{k,l} \omega_{k,l} f^{\kappa(k,l)}, \quad j = \sum_{k,l} \omega_{k,l} f^{\kappa(k,l)} V_{k,l}. \tag{7.4.15}$$

In this formalism, it is very easy to adapt a generic Discontinuous Galerkin solver for handling the reduced Vlasov-Maxwell model. We just have to modify the numerical flux and source functions. We have seen in (7.4.12) that the reduced Vlasov equation is a set of coupled transport equations with velocities V_i. We use a standard upwind numerical flux for the transport equations. The source term (7.4.13) is computed with the assembly procedure and superimposed with the Maxwell source term from (7.4.15) and (7.1.10).

Remark 7.4.1. *We will use a Runge-Kutta scheme of order* 2 *or order* 4 *in time. Note a CFL condition should be satisfied to ensure the stability of the scheme.*

7.4.4 Preliminary numerical results

In this section, we present preliminary Vlasov-Maxwell numerical results obtained with the reduced approach. We have not yet implemented the Child-Langmuir boundary condition. Therefore we only present a very simple and academic test case. We consider a cloud of charged particles in the center of a square domain Ω. Because the particles repel each other, the cloud will expand with time. We plot the evolution of the total charge in the domain at

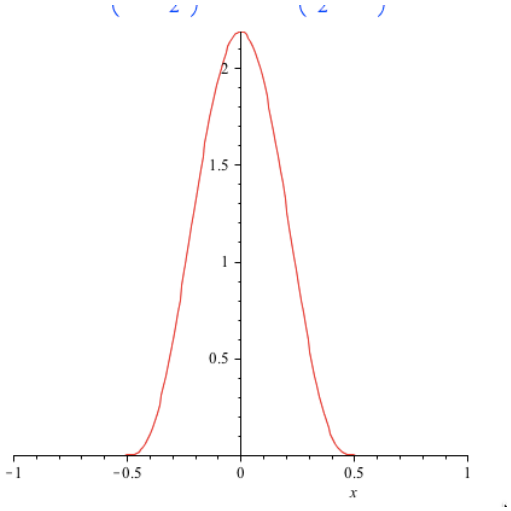


FIGURE 7.6 : The function q.

different times. We also represent the distribution function in the velocity space at a given point $(x_1, x_2) = (0.37, 0.5)$. The initial distribution function is

$$f(x_1, x_2, v_1, v_2, 0) = -q(x_1)q(x_2)q(v_1)q(v_2),$$

where the function q, represented on Figure 7.6, has its support in $[-1/2, 1/2]$ and satisfies $\int_{r=-\infty}^{\infty} q(r)dr = 1$.

We suppose that the initial electromagnetic field vanishes. This initial condition is not physical, because the Gauss law is not satisfied : $\nabla \cdot E \neq \rho$. Therefore, we take a divergence correction parameter $\chi = 4$. On the boundary of Ω, we apply homogeneous Silver-Müller conditions. Finally, we take $\mu = 1$: our computation is non-relativistic. The simulation time is short enough so that the exact distribution function vanishes on the boundary of the velocity disk, even at the final time $T = 1$.

We use a mesh of Ω with $8 \times 8 = 64$ cells. The velocity mesh is of order $d = 2$. It contains 305 Gauss-Lobatto nodes.

We plot the charge evolution on Figure 7.7.

We also plot the distribution function at point $(x_1, x_2) = (0.37, 0.5)$ at different times on Figure 7.8.

Finally, we plot the x_1-component of the electric field at time $t = 1$ on Figure 7.9.

Conclusion

In this work, we have presented two numerical schemes for approximating the Vlasov-Maxwell system. The first scheme is a coupling between an upwind DG solver for the Maxwell equations with a PIC solver for the Vlasov equation. We have reviewed some practical aspects of a robust and precise implementation of the whole procedure : high order polynomials, upwind flux, stable boundary conditions, divergence cleaning. We have also implemented the algorithm on GPU, which requires a sorting of the particles list at each time step. We obtained interesting speedups, but we also observe that the PIC method is the most expensive part of the computation. Therefore we propose another fully Eulerian approach. Thanks to

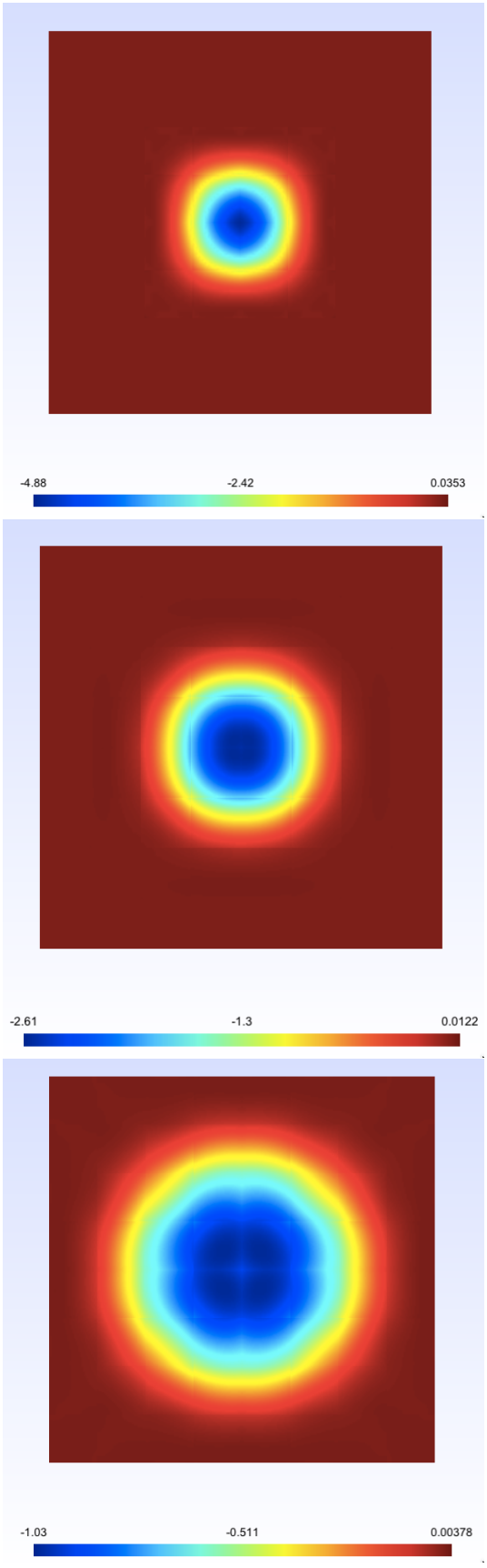


FIGURE 7.7 : evolution of the charge in the computational domain, $t = 0$ (top), $t = 0.5$ (middle), $t = 1.0$ (bottom).

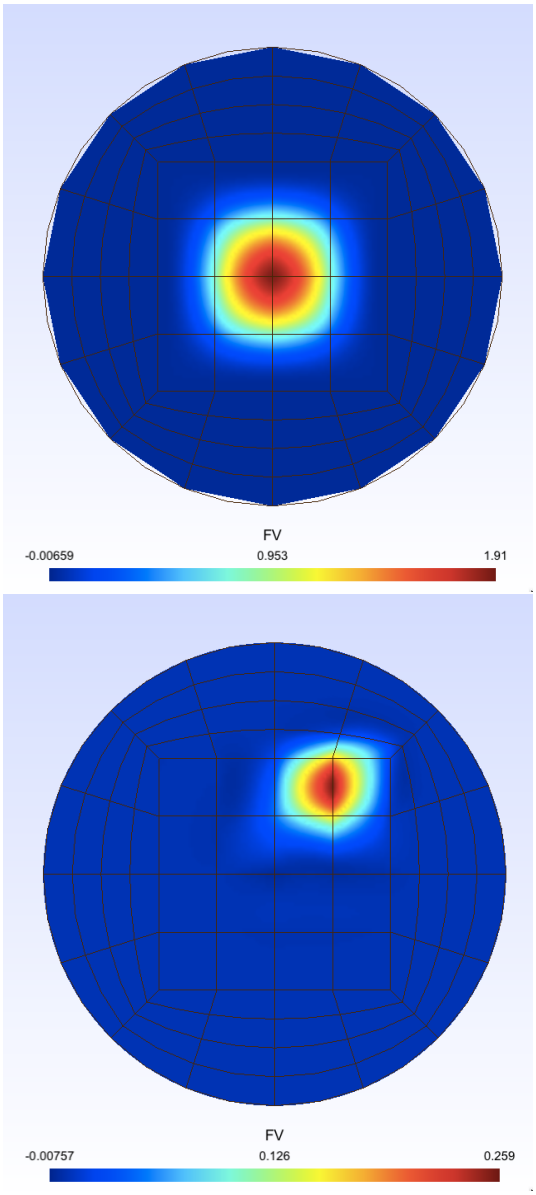


FIGURE 7.8 : Evolution of the distribution function $f(0.37, 0.5, v_1, v_2, t)$ in the computational domain, $t = 0$ (top), $t = 0.5$ (bottom). . The apparent polygonal shape of the mesh boundary is due to a post-processor bug in the first picture.

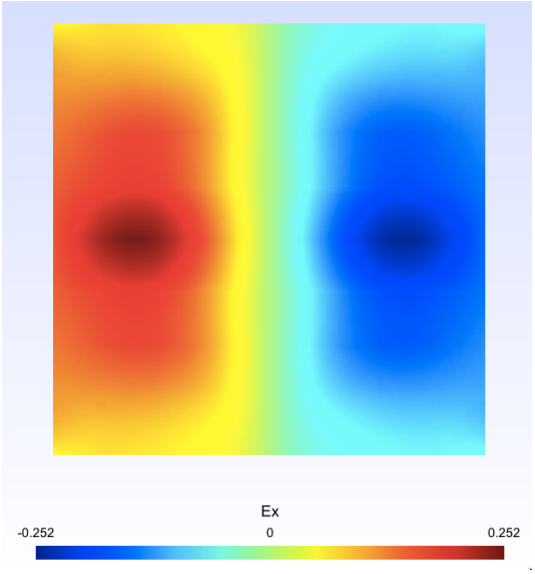


FIGURE 7.9 : x_1 component of the electric field at time $t = 1.0$.

a decomposition of the distribution function on velocity basis functions, we obtain a reduced Vlasov model, which appears to be a hyperbolic system of conservation laws written only in the (x, t) space. We can thus adapt very easily our DG solver to the reduced model. We presented preliminary numerical results. Our next step will be to implement more physical boundary conditions and test the reduced approach on emitting diode test cases.

Annexe 7.A Rate of convergence in velocity

We consider the initial function

$$f(x_1, x_2, v_1, v_2, 0) = q(v_1)q(v_2),$$

where q is defined in section 7.4.4 and the electric and magnetic fields are taken uniform and constant in time : $E_1 = 0, E_2 = 1$, and $H_3 = 0$.

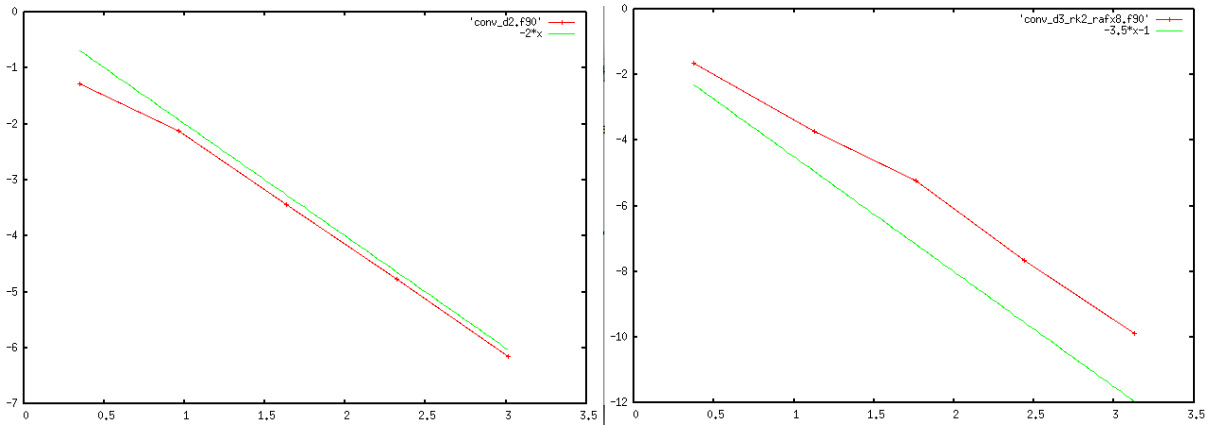
In the relativistic case, the exact solution is given by (thanks to the computation on Maple)

$$f(x_1, x_2, v_1, v_2, t) = \frac{1}{\mu} q(v_1)q(\bar{v}_2(t, v_1)),$$

with $\bar{v}_2(t, v_1)$ is given by

$$\bar{v}_2(t, v_1) = -\frac{-t\sqrt{1 - v_1^2 - v_2^2} + t\sqrt{1 - v_1^2 - v_2^2}v_1^2 - v_2}{t^2v_1^4 - 2t^2v_1^2 + t^2v_1^2v_2^2 + 1 - t^2v_2^2 + t^2 + 2t\sqrt{1 - v_1^2 - v_2^2}v_2}. \tag{7.A.1}$$

In Fig. 7.A, we plot the convergence analysis for the non-relativistic transport equation (resp. relativistic transport equation) when varying the number of basis function in velocity. In both case, we observe that the scheme is numerically of order 2 for d = 2 and about order 3.5 for d = 3, where d is the polynomial degree of the finite element basis in velocity. This is consistent with the theoretical order of convergence (which is of order d, see section (2.5)).



d = 2 d = 3

FIGURE 7.10 : Transport test case in the v_2 direction (relativistic case) : L^2 error between exact and numerical solution of the distribution function with respect to Δv_2 in log scale. Parameters : $T = 0.1, \kappa = 0.1$

Chapitre 8

Drift-kinetic reduced equation and discontinuous Galerkin scheme

Introduction

In the Tokamak devices, the plasma is confined in the toroidal chamber thanks to a large magnetic field. The particles rapidly rotate around the magnetic field lines. In order to avoid to consider the 6D Vlasov equation, gyro-kinetic models have been developed to take benefit from the fast rotation of the particles and consider models with only 4D variables (+ 1D adiabatic variable). For more details, we refer to [43, 32].

As a first step before tackling the Tokamak geometry, we consider a periodic cylinder without curvature and a magnetic field parallel to the cylinder axis. The associated plasma models are the so-called 4D drift-kinetic model. If the plasma is homogeneous in the magnetic field direction, the drift-kinetic model reduces to the 2D guiding center model. As regards the distribution function dynamics, the guiding center model is a conservative transport equation in the cylinder section (in (r, θ) variables) and thus is an hyperbolic system. The drift kinetic equation combines this 2D transport (in the cylinder section) with a 2D Vlasov equation in the (z, v_z) variables where z denotes the cylinder axis variable and v_z the associated kinetic velocity. Thanks to the reduction method (similar to section 2), the drift-kinetic model can be also written as an hyperbolic system.

To solve these hyperbolic systems, we here consider the discontinuous Galerkin (DG) method. Like the semi-Lagrangian method (see section 6), it is a way to reach high-order accuracy. Furthermore, it has the advantage to easily handle general geometries (contrary to the basic semi-Lagrangian method) [19, 73, 96, 12, 58].

This method is implemented in `schnaps` library ("Solveur Conservatif Hyperbolique Non-linéaire Appliqué aux PlasmaS") ([2]), developed in the Tonus team (Inria Nancy -Grand Est). Working within this library allows us to accelerate the code with parallelization tools, such as OpenCL[1] and StarPU[2]. Thanks to the high-order accuracy and the parallelization, we then can have reasonable computation time and good efficiency while taking small time step so as to satisfy the CFL stability conditions.

In section 1, we present the 4D drift-kinetic model and the 2D guiding center model.

[1] `https://www.khronos.org/opencl/`
[2] `http://starpu.gforge.inria.fr`

The electric potential satisfies the quasineutralityity equation. In the second section, we introduce the hyperbolic system obtained after the reduction of the drift-kinetic model. We then present the discontinuous Galerkin method to solve the reduced system and the Finite Element solver to solve the quasineutrality equation. In section 3, we briefly describe the `schnaps` library and discuss the geometry errors. Section 4 is devoted to the Diocotron instability test case for the guiding center model. We report the computation of the growth rate (already given in [29]). In the last section, we discuss and measure the parallelization of the code with OpenCL. We compare the computation time between a sequential code and a parallelized code, and between the parallelization executed by a GPU or by a multi-core CPU, since OpenCL can handle both configurations.

8.1 Drift-kinetic, guiding center and quasineutralityity models

8.1.1 Drift-kinetic model

Let us consider the dimensionless Vlasov equation (1.2.1) as written in chapter 1

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + (\mathbf{E} + \mathbf{v} \times \mathbf{B}_{\text{app}}) \cdot \nabla_{\mathbf{v}} f = 0.$$

where $\mathbf{B}_{\text{app}}$ is an applied magnetic field. The computation domain is defined by $\mathbf{x} \in \Omega_v \subset \mathbb{R}^3$ and $\mathbf{v} \in \Omega_v \subset \mathbb{R}^3$. The applied magnetic field is supposed to be uniform : if e_3 denotes the unit vector in that direction, we have $\mathbf{B}_{\text{app}} = b\,\mathbf{e}_3$ with $b \in \mathbb{R}$. We consider the case of a large magnetic field : $b = 1/\varepsilon$ with $0 < \varepsilon \ll 1$. In large time, the particles exhibit a drift at velocity $\varepsilon \mathbf{E}_\perp^\perp$ in the orthogonal plane to $\mathbf{B}_{\text{app}}$, where $\mathbf{E}_\perp = (E_{x_1}, E_{x_2})^T$ denotes the components of the electric field in the orthogonal plane to $\mathbf{B}_{\text{app}}$ and $\mathbf{E}_\perp^\perp = (-E_{x_2}, E_{x_1})^T$ the perpendicular vector. In order to obtain a macroscopic drift, we consider small scale in velocity and space in the orthogonal direction (it is equivalent to consider large time scale in the orthogonal direction) :

$$\begin{aligned} \widetilde{\mathbf{x}}_\perp &= \frac{1}{\varepsilon}\mathbf{x}_\perp, & \widetilde{x}_{||} &= x_{||}, \\ \mathbf{u}_\perp &= \frac{1}{\varepsilon}(\mathbf{v}_\perp - \varepsilon\, \mathbf{E}_\perp^\perp), & u_{||} &= v_{||}. \end{aligned}$$

where $\mathbf{x}_\perp = (x_1, x_2)^T$, $\mathbf{v}_\perp = (v_1, v_2)^T$ and $x_{||} = x_3$, $v_{||} = v_3$. We then consider the new unknown function :

$$\begin{aligned} \tilde{f}(\widetilde{\mathbf{x}}_\perp, \widetilde{x}_{||}, u_{||}, \mathbf{u}_\perp, t) &= f(\mathbf{x}_\perp, x_{||}, v_{||}, \mathbf{v}_\perp, t), \\ \tilde{E}(\widetilde{\mathbf{x}}_\perp, \widetilde{x}_{||}, t) &= E(\mathbf{x}_\perp, x_{||}, t). \end{aligned}$$

We then have

$$\begin{aligned}
\nabla_{v_{||}} f &= \nabla_{u_{||}} \tilde{f}, \qquad \nabla_{\mathbf{v}_\perp} f = \frac{1}{\varepsilon} \nabla_{\mathbf{u}_\perp} \tilde{f} \\
\nabla_{\mathbf{x}} f &= \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \tilde{f} \\ \nabla_{\widetilde{x}_{||}} \tilde{f} \end{bmatrix} - \nabla_{\mathbf{x}} \mathbf{E}_\perp^{\perp T} \nabla_{u_\perp} \tilde{f} \\
&= \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \tilde{f} \\ \nabla_{\widetilde{x}_{||}} \tilde{f} \end{bmatrix} - \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \mathbf{E}_\perp^{\perp T} & \nabla_{\widetilde{x}_{||}} \mathbf{E}_\perp^{\perp T} \\ 0 & 0 \end{bmatrix}^T \begin{bmatrix} \nabla_{\mathbf{u}_\perp} \tilde{f} \\ \nabla_{u_{||}} \tilde{f} \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \tilde{f} \\ \nabla_{\widetilde{x}_{||}} \tilde{f} \end{bmatrix} - \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \mathbf{E}_\perp^{\perp} & 0 \\ \nabla_{\widetilde{x}_{||}} \mathbf{E}_\perp^{\perp} & 0 \end{bmatrix} \begin{bmatrix} \nabla_{\mathbf{u}_\perp} \tilde{f} \\ \nabla_{u_{||}} \tilde{f} \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \tilde{f} - \frac{1}{\varepsilon} \nabla_{\widetilde{\mathbf{x}}_\perp} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} \tilde{f} \\ \nabla_{\widetilde{x}_{||}} \tilde{f} - \nabla_{\widetilde{x}_{||}} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} \tilde{f} \end{bmatrix}
\end{aligned}$$

where the partial derivative with respect to $x_{||}$ and $v_{||}$ are denoted by $\nabla_{x_{||}}$ and $\nabla_{v_{||}}$ as the perpendicular gradient to make the notations uniform. Therefore, the Vlasov equation writes, after dropping the tildes :

$$\begin{aligned}
\partial_t f + u_{||} \cdot \Big[\nabla_{x_{||}} f - \nabla_{x_{||}} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \Big] + \Big[\varepsilon \, \mathbf{u}_\perp + \varepsilon \, \mathbf{E}_\perp \perp \Big] \cdot \frac{1}{\varepsilon} \Big[\nabla_{\mathbf{x}_\perp} f - \nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \Big] \\
+ E_{||} \cdot \nabla_{u_{||}} f + \Big(\mathbf{E}_\perp + \frac{1}{\varepsilon} \left(\varepsilon \, \mathbf{u}_\perp + \varepsilon \, \mathbf{E}_\perp^{\perp} \right)^\perp \Big) \cdot \frac{1}{\varepsilon} \nabla_{\mathbf{u}_\perp} f = 0.
\end{aligned}$$

where the scalar product in the perpendicular direction is the 2D scalar product and the scalar product in the parallel direction is just the multiplication. Since $\mathbf{E}_\perp^{\perp \perp} = -\mathbf{E}_\perp$, we have :

$$\begin{aligned}
\partial_t f + u_{||} \cdot \Big[\nabla_{x_{||}} f - \nabla_{x_{||}} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \Big] + \Big[\mathbf{u}_\perp + \mathbf{E}_\perp^{\perp} \Big] \cdot \Big[\nabla_{\mathbf{x}_\perp} f - \nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \Big] \\
+ E_{||} \cdot \nabla_{u_{||}} f + \frac{1}{\varepsilon} \, \mathbf{u}_\perp^{\perp} \cdot \nabla_{\mathbf{u}_\perp} f = 0.
\end{aligned}$$

We then integrate in the $\mathbf{u}_\perp$ direction. The following quantities vanishes :

$$\begin{aligned}
&\int u_{||} \cdot \Big[\nabla_{x_{||}} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \Big] \, d\mathbf{u}_\perp = 0, \\
&\int \mathbf{E}_\perp^{\perp} \cdot \nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \, d\mathbf{u}_\perp = 0, \\
&\int \mathbf{u}_\perp^{\perp} \cdot \nabla_{\mathbf{u}_\perp} f \, d\mathbf{u}_\perp = - \int (\nabla_{\mathbf{u}_\perp} \cdot \mathbf{u}_\perp^{\perp}) \, f d\mathbf{u}_\perp = 0,
\end{aligned}$$

and

$$\begin{aligned}
\int \mathbf{u}_\perp \cdot \nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp} \nabla_{\mathbf{u}_\perp} f \, d\mathbf{u}_\perp &= \int (\nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp T} \mathbf{u}_\perp) \cdot \nabla_{\mathbf{u}_\perp} f \, d\mathbf{u}_\perp \\
&= - \int \nabla_{\mathbf{u}_\perp} \cdot \Big[(\nabla_{\mathbf{x}_\perp} \mathbf{E}_\perp^{\perp})^T \mathbf{u}_\perp \Big] f \, d\mathbf{u}_\perp \\
&= 0,
\end{aligned}$$

since $\partial_i(\partial_j E^{\perp}_{\perp\, i} u_j) = \partial_j E^{\perp}_{\perp\, j} = (\nabla_{\mathbf{x}_\perp} \cdot \mathbf{E}^{\perp}_{\perp}) = \pm(\nabla_{\mathbf{x}_\perp} \times \mathbf{E}_\perp) = 0$. we end up with an equation on the averaged distribution function :

$$\partial_t \bar{f} + u_{||} \cdot \nabla_{x_{||}} \bar{f} + \mathbf{E}^{\perp}_{\perp} \cdot \nabla_{\mathbf{x}_\perp} \bar{f} + E_{||} \cdot \nabla_{u_{||}} \bar{f} = 0.$$

where

$$\bar{f}(x, u_{||}, t) = \int f(x, u_{||}, \mathbf{u}_\perp, t)\, d\mathbf{u}_\perp.$$

Or in a simpler form

$$\partial_t f + \mathbf{E}^{\perp}_{\perp} \cdot \nabla_{\mathbf{x}_\perp} f + v_{||} \partial_{x_{||}} f + E_{||} \partial_{v_{||}} f = 0. \tag{8.1.1}$$

Remark 8.1.1. *In our study, we consider that the spatial domain is the cylinder :* $\Omega_x = D \times [0, L]$*, where* D *is the disc in* $\mathbb{R}^2$ *of radius* R *centered at the origin.*

8.1.2 Quasineutrality equation

The electric field in equation (8.1.1) is computed by

$$\mathbf{E} = -\nabla \Phi, \tag{8.1.2}$$

in which Φ is the electric potential. It is solution to the quasineutrality Poisson equation

$$-\nabla_{\mathbf{x}_\perp} \cdot \left(\frac{\rho_0}{B} \nabla_{\mathbf{x}_\perp} \Phi \right) + \frac{\rho_0}{T_e} (\Phi - \bar{\Phi}) = \rho - \rho_0, \tag{8.1.3}$$

where $\rho_0(x_\perp)$ and $T_e(x_\perp)$ are density and electron temperature profiles in the transverse disc and $\bar{\Phi}$ is the potential averaged along the magnetic field line

$$\bar{\Phi}(\mathbf{x}_\perp, t) = \frac{1}{L} \int_0^L \Phi(\mathbf{x}_\perp, x_{||}, t)\, dx_{||}.$$

This model was considered in [43, 72]. In order to solve the quasineutralityity equation (8.1.3), we first take its average in the $x_{||}-$direction. Since Φ is periodic in the $x_{||}-$direction(in particular $\Phi(\mathbf{x}_\perp, 0) = \Phi(\mathbf{x}_\perp, L_{x_{||}})$), we obtain

$$-\nabla_\perp \cdot \left(\frac{\rho_0(\mathbf{x}_\perp)}{B} \nabla_\perp \bar{\Phi} \right) = \bar{\rho} - \rho_0, \tag{8.1.4}$$

where $\bar{\rho}$ is the average mean charge in the $x_{||}-$direction

$$\bar{\rho}(\mathbf{x}_\perp, t) = \frac{1}{L} \int_0^L \rho(\mathbf{x}_\perp, x_{||}, t)\, dx_{||}.$$

Substracting the two equations (8.1.3) and (8.1.4), we obtain

$$-\nabla_{\mathbf{x}_\perp} \cdot \left(\frac{\rho_0(\mathbf{x}_\perp)}{B} \nabla_{\mathbf{x}_\perp} (\Phi - \bar{\Phi}) \right) + \frac{\rho_0(\mathbf{x}_\perp)}{T_e(\mathbf{x}_\perp)} (\Phi - \bar{\Phi}) = \rho - \bar{\rho}, \tag{8.1.5}$$

with the average of electric charge $\bar{\rho}$. Note that, once the density is known, this quasineutral Poisson equation is only set on the transverse disc. Numerically, the equation can be solved slice by slice in the $x_{||}-$direction.

Remark 8.1.2. *In practice, instead of solving* (8.1.4)*, we consider that the average potential vanishes :* $\bar{\Phi} = 0$*.*

The purpose of this chapter is to solve the drift kinetic equation (8.1.1) in which the electric potential Φ is solution of (8.1.5).

8.1.3 Guiding-center model

In the case when all quantities are homogeneous in the direction of the magnetic field i.e. f does not depend on x_3, we obtain the two dimensional equation in the disc

$$\partial_t f + \nabla_{\mathbf{x}_\perp} \Phi^\perp \cdot \nabla_{\mathbf{x}_\perp} f = 0.$$

After integrating in velocity, we obtain the guiding center model

$$\partial_t \rho + \nabla_{\mathbf{x}_\perp} \Phi^\perp \cdot \nabla_{\mathbf{x}_\perp} \rho = 0. \tag{8.1.6}$$

This equation is coupled with the Poisson equation

$$-\Delta_{\mathbf{x}^\perp} \Phi = \rho. \tag{8.1.7}$$

Remark 8.1.3. *This system is equivalent to the 2D incompressible Euler equation in the vorticity formulation where ρ stands for the vorticity and $\mathbf{E}^\perp = \nabla_{\mathbf{x}^\perp} \Phi^\perp$ for the fluid velocity.*

Remark 8.1.4. *We note that the drift-kinetic model* (8.1.6)-(8.1.7) *is the combination of the guiding center model in the transverse disc and a one-dimensional Vlasov transport equation in the $(x_{||}, v_{||})$ direction.*

8.2 Numerical method

The proposed numerical method to solve (8.1.1)-(8.1.5) consists in applying the same methodology as introduced in chapter 2 : first discretize the drift-kinetic equation using the finite element method in velocity and then use a transport solver to solve the hyperbolic system. Here we consider the discontinuous Galerkin (DG) solver with Gauss-Lobbato points for accuracy and efficiency reasons (see Section 2). The quasineutrality equation (or the Poisson equation) are solved with a Lagrange finite-element solver (FE), valid for general elliptic equations. It requires a mapping from the DG nodes to the FE nodes.

8.2.1 Finite element method in velocity

Let assume that the velocity domain is bounded and given by $[-V_{\max}, V_{\max}]$. We consider the d + 1 Gauss-Lobatto interpolation points in the reference interval $[0, 1]$ and M elements in the domain $[-V_{\max}, V_{\max}]$. Considering the associated Lagrange FE basis $(\varphi_j)_{j=1,\dots,N_v}$ and using the following decomposition

$$f(\mathbf{x}, v, t) = \sum_{j=1}^{N_v} w_j(\mathbf{x}, t) \varphi_j(v). \tag{8.2.1}$$

The weak formulation (in velocity) of the drift-kinetic equation writes

$$\mathcal{M} \partial_t \mathbf{w} + \mathbf{E}_\perp \cdot \mathcal{M} \nabla_\perp \mathbf{w} + A\, \partial_{x_{||}} \mathbf{w} + B(E_{x_{||}})\, \mathbf{w} = 0. \tag{8.2.2}$$

where $\mathbf{w}(\mathbf{x},t) = (w_j(\mathbf{x},t))_j \in \mathbb{R}^{N_v}$ is the vector of unknown components (in the FEM basis) and matrices $\mathcal{M}, A, B$ of dimension $N_v \times N_v$ are defined by

$$\mathcal{M} = \Big(\int_v \varphi_i \varphi_j \Big), \quad A = \Big(\int_v v \varphi_i \varphi_j \Big),$$
$$B(E_{x_{||}}) = E_{x_{||}} \left(\beta \frac{E^+_{x_{||}}}{E_{x_{||}}} \varphi_j(-V)\varphi_i(-V) - \beta \frac{E^-_{x_{||}}}{E_{x_{||}}} \varphi_j(V)\varphi_i(V) - \int_v \varphi_i \varphi_j' \right).$$

These are the same matrices as in Chapter 2. We recall that when choosing the Gauss-Lobatto quadrature points the matrices $\mathcal{M}$ and A are diagonal. More precisely, we have

$$\mathcal{M}_{ii} = \sum_{i=\kappa(k,l)} \omega_{k,l} = \sum_{i=\kappa(k,l)} \omega_l \det \tau_k'(N_l), \tag{8.2.3}$$

with τ_k is the linear transformation which maps $[0,1]$ onto the element Q_k

$$\tau_k(\hat{v}) = -V_{\max} + (k-1)\Delta v + \Delta v\, \hat{v}.$$

Since $\tau_k' = \Delta v$ for all k, we then have $\mathcal{M}_{ii} = \Delta_v \sum_{i=\kappa(k,l)} \omega_l$ and consequently

$$A_{ii} = \mathcal{M}_{ii}\, N_i.$$

The definition of all the involved quantities ($N_l, \kappa, etc.$) can be found in Section 2.

8.2.2 Discontinuous Galerkin methods in space

We describe the discontinuous Galerkin methods for the transport equation. For more detail about the DG method, we refer, for instance, to [73, 96, 12, 58]). We consider a multi-patch mapped Cartesian mesh of the computational domain, denoted $\mathcal{D} = \{\mathcal{D}_i\}$, which is a finite collection of disjoint mapped hexahedron such that

$$\overline{\Omega} = \cup_{\{\mathcal{D}_i \in \mathcal{D}\}} \overline{D}.$$

For an abitrary $\mathcal{D}_i \in \mathcal{D}$, $\mathcal{D}$ is called the mesh element, the diameter of $\mathcal{D}$, denote $h_{mathcalD} := \max_{x,y \in D} ||x-y||$. The meshsize of mesh $\mathcal{D}$ is then

$$h := \max_{\mathcal{D}_i \in \mathcal{D}} \{h_{\mathcal{D}}\}.$$

We denote $\mathcal{D}_h$ for the mesh $\mathcal{D}$ with a meshsize h.

The second step is then to contruct a discrete functional space. Here we consider the $\mathbb{Q}_{\mathbf{d_x}}$ space, where $\mathbf{d_x} = (d_{x_1}, d_{x_1}, d_{x_3}) \in \mathbb{N}^3$. In general $3D$ case, $\mathbf{d_x} = (d_{x_1}, d_{x_1}, d_{x_3}) \in \mathbb{N}^3$ but for the sake of simplicity, we consider $d_x = d_{x_1} = d_{x_1} = d_{x_3}$.

Now, let us consider the same strategy to the DG method for approximating the drif-kinetic equation like described in section 7.2.1. In an mesh element L, we define the approximated field by

$$\mathbf{w}^L(x,t) = \sum_k \mathbf{w}^{L,k}(t)\psi_{L,k}(x), \quad \{\psi_{L,k}\} \text{ basis of } P_2(\mathbb{R}^2)^{N_{v_{||}}}. \tag{8.2.4}$$

So, the weak upwind DG formulation for the drif-kinetic equation (8.2.2) is that for all test function ψ_L, we have

$$\int_L \partial_t \mathbf{w}^L \cdot \psi_L - \int_L E_\perp \mathbf{w}^L \cdot \nabla_\perp \psi_L + \int_{\partial L \cap \Omega_{\mathbf{x}}} E_\perp \big(n^+ \mathbf{w}_L + n^- \mathbf{w}_R\big) \cdot \psi_L + \int_L \mathbf{w}^L \cdot (\mathcal{M}^{-1} A) \partial_3 \psi_L + \int_{\partial L \cap \Omega_{\mathbf{x}}} ((\mathcal{M}^{-1} A) n_3^+ \mathbf{w}_L + (\mathcal{M}^{-1} A) n_3^- \mathbf{w}_R) \cdot \psi_L + \int_L (\mathcal{M}^{-1} B(E_{X_3})) \mathbf{w}^L \cdot \psi_L = 0 \tag{8.2.5}$$

where n the normal vector on ∂L oriented from the cell L to the neighboring cells R and

$$x^+ = \max(0, x), \quad x^- = \min(0, x).$$

Remark 8.2.1. *For the time discretization, we use a RK4 scheme.*

Remark 8.2.2. *The convergence rate of DG is $d_x + 1/2$ (the proof is given, for instance, in [94]) while it is of order d_x for FEM.*

Remark 8.2.3. *At given approximation degree $d_x \in \mathbb{N}$, the implementation of DG methods are more expensive than the FEM. It is due to the storage requirements of DG methods. Let us illustrate this in the two dimensional case with a Cartesian grid. We further assume than the approximation degree, d_x, is the same in each direction and that we have M_x elements in each dimension. So that, the total number of degrees of freedom equals $(d_x M_x + 1)^2$ for the FEM, while the total number of freedoms for the DG methods it equals $(d_x + 1)^2 M_x^2$. So, with the DG methods, we have $(M_x - 1)(M_x + 1 + 2M_x d_x)$ more degrees of freedom. In figure 8.2.3, we have the case $d_x = M_x = 2$, so we have 25 nodes with FEM and 36 points for the DG methods.*

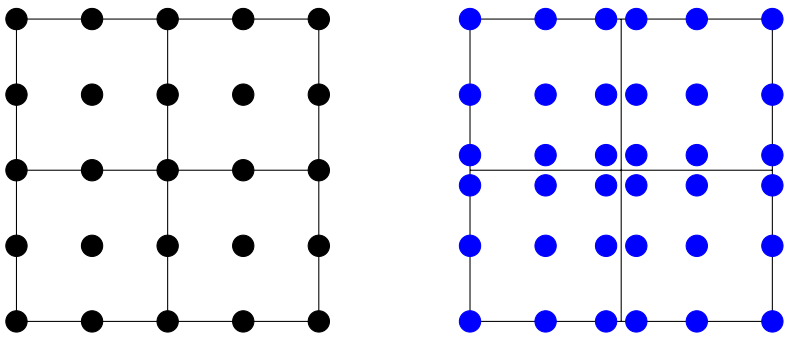


FIGURE 8.1 : DOF in $2D$ square mesh. Left : FEM, right : DG methods.

8.2.3 Finite Element Method for the (quasineutral) Poisson equation

The quasineutral equation (8.1.3) is solved using a standard Lagrange Finite Element Method. We use the same Gauss-Lobatto nodes as for the DG discretization of the drift-kinetic equation to ensure compatibility. To make the coupling between the two equations (for the

electric potential and the density), it is necessary to convert the continuous points into discontinuous points after solving the quasineutral equation. In practice, the electric potential is simply duplicated on all the discontinuous Gauss-Lobatto points before computing the electric field. In addition, the finite element assembly procedure handles in a natural way the discontinuous charge density source term in the Poisson equation.

In practice, since we solve the quasineutrality equation slice by slice , we need to compute the number of slices, the number of finite element points in each slice. In order to efficiently make the relation between the DG points and the FEM points, we build a function that converts the coordinates of each Gauss points into a set of three integers (one for each direction). We then sort in lexicographical order and we compare the discontinuous points two by two (closed DG points are associated into so called FatNodes). Then, we count the finite elements and compute the total number of finite element points. At the end of this procedure, we obtain the array of FEM points and their relation with the discontinuous Galerkin points. By gathering the finite element points in the plane $x_3 = 0$, we have the number of finite element points in each slice. Then we obtain the number of slices by dividing the total number of finite elements points by the number of points in each slice. We also count the number of finite element points at the boundary in order to update the mass matrix involved in the resolution of the quasineutrality equation. The homogeneous Dirichlet boundary condition is imposed by a penalization method. We add big values in the corresponding diagonal terms of the mass matrix.

8.3 Implementation : `schnaps` code

`Schnaps` is a code for solving hyperbolic systems with the discontinuous Galerkin method (see [2]). As mentioned in the previous section, a finite element elliptic solver is also implemented. It is written in C programming language and was designed to be executed on GPU. As in chapters 6 and 7, we use the OpenCL language to parallelize the code : the code can thus be executed on both a GPU or a multi-core CPU. For additional details, see section 6.3.3. More recent versions of the code involve parallelization using the starPU library (for more detail, see [4]), that enables to manage the tasks on heterogeneous hardware. Future development would be to use this new functionality.

8.4 Mesh generation

In this section, we describe how we construct the computational domain approximating the cylinder geometry. For this purpose, we consider that the whole physical domain is divided into macrocells. The macrocells are built by mapping a reference cube onto H20 hexahedrons. H20 finite elements allow to representing curved geometry with second order accuracy. The control points of a H20 hexahedron are the 8 nodes of the hexahedron and the 12 nodes a the middle of the edges (for more detail, see [10, 2]). Therefore, each macrocell is described by a quadratic transformation τ which maps the cubic reference element onto the curved hexahedron. Let us detail the construction of the geometry in the 2D slices. We use 5 macrocells as presented in figure (8.4). We then construct sub-meshes in each macro mesh with N_{raf} quadrilaterals (in each direction) and the transformation τ maps the

reference square into each sub-mesh. The transformation is defined by the formulation (7.2.1) in chapter 7.

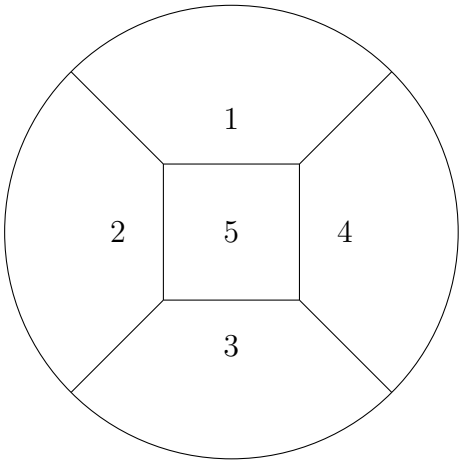


FIGURE 8.2 : Macro-mesh of a simple circle

Remark 8.4.1. *Note that we actually approximate the disc by a domain with piecewise quadratic boundaries. With such a construction, we create a geometry error of order 2. In order to decrease the geometry error, we have to increase the number of macrocells.*

Among the five macro meshes (see Fig. 8.4), the 5^{th} macro mesh is a square, so in this macro-mesh, we obtain less geometry error than in others macro meshes. It is an interesting problem to construct macro meshes suitable for each case test.

We illustrate the geometry error on a simple test case for the Poisson equation. In polar coordinates, the Poisson equation writes

$$-\frac{1}{r}\frac{\partial}{\partial r}\big(r\frac{\partial \Phi}{\partial r}\big) - \frac{1}{r^2}\frac{\partial^2 \Phi}{\partial^2 \theta} = \rho - \rho_0.$$

We consider Dirichlet boundary conditions at $r = R$, reads $\Phi(t, R, \theta) = 0$, and Neumann boundary condition at $r = 0$, namely $\partial_r \Phi(t, 0, \theta) = 0$. For a uniform density equal to 1 in the whole domain, the solution to the Poisson equation is given by : $\Phi(r, \theta) = -(r^2 - R^2)/4$. In Fig.8.3, we display the error of the exact solution with the numerical solution at fixed number of discretization points (120 Gauss points meshes in the radial direction and approximately 320 Gauss points in the angular direction) : the L^∞ error is of order 7.4×10^{-4} when using 5 macro-cells with 16 sub-cells (per direction) while it is of order 2.5×10^{-4} when using 4×5 macro-cells witt only 4 sub-cells (per direction). This is only due to the geometry error. We note also that the errors appear at the boundary.

8.5 Diocotron instability test case

To validate our code, we consider the diocotron instability test-case : this is a Rayleigh-Taylor type instability for the guiding-center model (see section 8.1.3). This test case was numerically studied in [72, 43]. The instability rates are presented in [29]. In this section, we reproduce their derivation and we perform numerical experiments. We observe the importance of the geometry approximation for capturing the right instability rates.

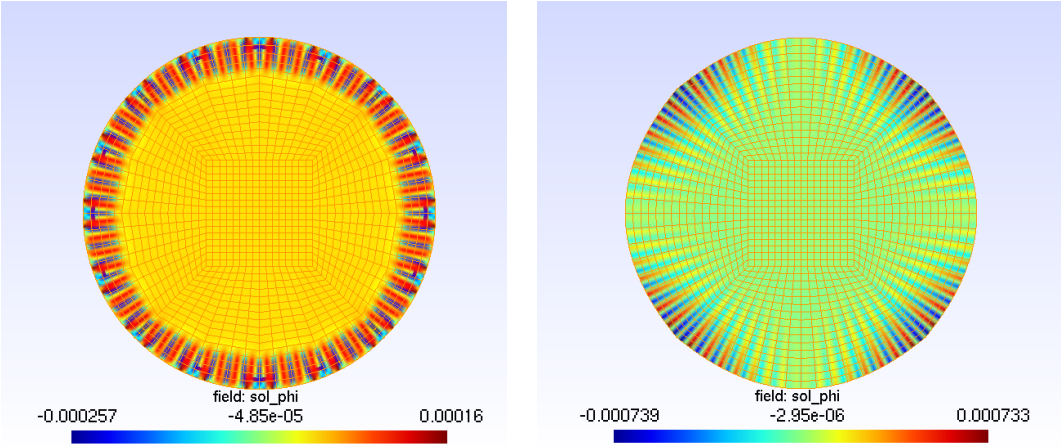


FIGURE 8.3 : Difference between the exact solution and the approximate solution of the potential. Parameters : Left : 4×5 macro-cells with $d_x = 2$ and $M_x = 4$ elements in each direction. Right : 5 macro-cells with $d_x = 2$ and $M_x = 16$ elements in each direction.

8.5.1 Linearization of the guiding-center model

In polar coordinates $(r, \theta) \in \Omega = [0, R] \times [0, 2\pi]$ the guiding-center equation (8.1.6) and the Poisson equation (8.1.7) become (see [72]) :

$$\partial_t \rho + \frac{1}{r}\big(\partial_r \Phi\, \partial_\theta \rho - \partial_\theta \Phi\, \partial_r \rho\big) = 0,$$
$$-\frac{1}{r}\partial_r(r\partial_r \Phi) - \frac{1}{r^2}\partial_{\theta^2}\Phi = \rho$$

where $\rho = \rho(t, r, \theta)$ the density and $\Phi = \Phi(t, r, \theta)$ is the potential.

The linearized equation around a radial solution $(\rho_0(r), \Phi_0(r))$ writes :

$$\partial_t \tilde{\rho} + \frac{1}{r}\big(\partial_r \Phi_0\, \partial_\theta \tilde{\rho} - \partial_\theta \tilde{\Phi}\, \partial_r \rho_0\big) = 0,$$
$$-\frac{1}{r}\partial_r(r\partial_r \tilde{\Phi}) - \frac{1}{r^2}\partial_{\theta^2}\tilde{\Phi} = \tilde{\rho}.$$

To simplify the notation, we remove the tilde over the perturbed quantities. Considering perturbations like :

$$\rho(r, \theta, t) = \rho_\ell(r) e^{i\ell\theta} e^{-i\omega t},$$
$$\Phi(r, \theta, t) = \Phi_\ell(r) e^{i\ell\theta} e^{-i\omega t},$$

we obtain the dispersion relations :

$$-i\omega\rho_\ell + \frac{i\ell}{r}\big(\partial_r \Phi_0\, \rho_\ell - \Phi_\ell\, \partial_r \rho_0\big) = 0,$$
$$-\frac{1}{r}\partial_r(r\partial_r \Phi_\ell) + \frac{\ell^2}{r^2}\Phi_\ell = \rho_\ell$$

hence the closed equation :

$$-\frac{1}{r}\partial_r(r\partial_r \Phi_\ell) + \frac{\ell^2}{r^2}\Phi_\ell = \frac{\partial_r \rho_0}{-\omega + \ell\partial_r \Phi_0 / r}\frac{\ell \Phi_\ell}{r} \tag{8.5.1}$$

8.5.2 Diocotron instability

Let us consider the following initial condition

$$\rho_0(r,\theta) = \begin{cases} 0,\ 0 \le r < a, \\ 1,\ a \le r \le b, \\ 0,\ b < r \le R. \end{cases} \tag{8.5.2}$$

If we consider the Poisson equation with Dirichlet boundary condition at $r = R : \Phi_0(t, R, \theta) = 0$, and Neumann boundary condition at $r = 0$, namely $\partial_r \Phi_0(t, 0, \theta) = 0$, we obtain the solution

$$\Phi_0(r) = \Big(k(a) - k(b) + w(b)\Big)\, 1_{r\in[0,a]} + \Big(k(r) - k(b) + w(b)\Big)\, 1_{r\in[a,b]} + w(r)\, 1_{r\in[b,R]},$$

with two functions k, w defined by :

$$k(r) = a^2/2 \log(r) - r^2/4,$$
$$w(r) = bk'(b) \log(r/R).$$

Démonstration. On $[0, a]$, we have $\partial_r(r\partial_r\Phi_0)/r = 0$, and because of the Neumann boundary condition at $r = 0$, we have $r\partial_r\Phi_0(r) = 0$. It infers that $\Phi_0(r) = c_1$ where c_1 is a constant to be determined.

On $[a, b]$, we have $-\partial_r(r\partial_r\Phi_0) = r$. By continuity of $\partial_r\Phi_0$, we have : $\partial_r\Phi_0(a) = 0$ and therefore : $r\partial_r\Phi_0(r) = -(r^2 - a^2)/2$. We then obtain : $\Phi_0(r) = k(r) + c_2$ with $k(r) = a^2/2 \log(r) - r^2/4$.

On $[b, R]$, we have $-\partial_r(r\partial_r\Phi_0) = 0$. By continuity of $\partial_r\Phi_0$, we have : $\partial_r\Phi_0(b) = k'(b)$. We thus obtain : $r\partial_r\Phi_0(r) = bk'(b)$. We then have : $\Phi_0(R) - \Phi_0(r) = bk'(b) \log(R/r)$. Using the Dirichlet boundary condition at $r = R$, we obtain : $\Phi_0(r) = w(r) = bk'(b) \log(r/R)$.

The constants c_1 and c_2 are determined to ensure the continuity of Φ_0. □

Proposition 8.5.1 (Davidson, chap. 6 [29])**.** *The solution of equation* (8.5.1) *is given by :*

$$\Phi_\ell(r) = \sum_{i=1}^{3} (A_i r^\ell + B_i r^{-\ell}) 1_{I_i}$$
$$a^\ell A_1 = (a^\ell A_2 + a^{-\ell} B_2) g_{A1}, \quad a^{-\ell} B_1 = (a^\ell A_2 + a^{-\ell} B_2) g_{B1}, \tag{8.5.3}$$
$$b^\ell A_3 = (b^\ell A_2 + b^{-\ell} B_2) g_{A3}, \quad b^{-\ell} B_3 = (b^\ell A_2 + b^{-\ell} B_2) g_{B3}, \tag{8.5.4}$$
$$a^\ell A_2 + a^{-\ell} B_2 = 2\big(1 - (a\omega - c_a)\big)\, a^{-\ell} B_2, \tag{8.5.5}$$

and the growth rates of instability ω satisfying the dispersion equation :

$$\alpha_2 \omega^2 + \alpha_1 \omega + \alpha_0 = 0, \tag{8.5.6}$$
$$\alpha_2 = -2 \frac{b^\ell}{a^\ell} f_{b,-}, \tag{8.5.7}$$
$$\alpha_1 = \frac{a^\ell}{b^\ell}\Big(f_{b,+}\Big) - \frac{b^\ell}{a^\ell}\Big(f_{b,-}(1 - 2c_a) + 2(1 - c_b f_{b,-})\Big), \tag{8.5.8}$$
$$\alpha_0 = \frac{a^\ell}{b^\ell}(1 - c_b f_{b,+}) - \frac{b^\ell}{a^\ell}(1 - 2c_a)(1 - c_b f_{b,-}). \tag{8.5.9}$$

or after simplification :

$$\alpha_2 = 1 \tag{8.5.10}$$
$$\alpha_1 = \frac{\ell}{2}(1-(a/b)^2) + \frac{1}{2}\Big((b/R)^{2\ell} - (a/R)^{2\ell}\Big) \tag{8.5.11}$$
$$\alpha_0 = -\frac{1}{4}\big(1-\frac{a^{2\ell}}{b^{2\ell}}\big)\big(1-\frac{b^{2\ell}}{R^{2\ell}}\big) + \frac{\ell}{4}\big(1-(\frac{a}{b})^2\big)\big(1-(\frac{a}{R})^{2\ell}\big) \tag{8.5.12}$$
$$\tag{8.5.13}$$

The parameters are given by :

$$\begin{aligned}
&c_a = \ell\partial_r\Phi_0(a), \quad c_b = \ell\partial_r\Phi_0(b),\\
&g_{A1} = 1, \quad g_{B1} = 0, \quad g_{A3} = h(R/b), \quad g_{B3} = h(b/R),\\
&f_{b,\pm} = g_{A3} - g_{B3} \pm 1 = f(R/b) \pm 1,
\end{aligned}$$

where $h(x) = 1/(1-x^{2\ell})$, $f(x) = (1+x^{2\ell})/(1-x^{2\ell})$.

Démonstration. In each interval $I_1 = [0,a]$, $I_2 = [a,b]$ and $I_3 = [b,R]$, equation (8.5.1) reduces to :

$$-r^2\partial_r^2\Phi_\ell - r\partial_r\Phi_\ell + \ell^2\Phi_\ell = 0,$$

Hence the solution takes the form :

$$\Phi_\ell(r) = \sum_{i=1}^{3}(A_i r^\ell + B_i r^{-\ell})1_{I_i}.$$

with A_i and B_i are constant.

Neumann boundary condition at $r = 0$ implies :

$$A_1 r^{\ell-1} - B_1 r^{-\ell-1} = 0,$$

and then $B_1 = 0$. Continuity of Φ_ℓ at $r = a$ implies :

$$A_1 a^\ell = A_2 a^\ell + B_2 a^{-\ell}$$

Hence :

$$a^\ell A_1 = (a^\ell A_2 + a^{-\ell}B_2)g_{A1}, \quad a^{-\ell}B_1 = (a^\ell A_2 + a^{-\ell}B_2)g_{B1}$$

with $g_{A1} = 1$ and $g_{B1} = 0$. We thus obtain (8.5.3).

Dirichlet boundary condition at $r = R$ implies :

$$A_3 R^\ell + B_3 R^{-\ell} = 0,$$

hence $B_3 = -A_3 R^{2\ell}$. Continuity at $r = b$ implies :

$$A_3 b^\ell (1 - R^{2\ell}b^{-2\ell}) = A_2 b^\ell + B_2 b^{-\ell}.$$

Hence :

$$b^\ell A_3 = (b^\ell A_2 + b^{-\ell}B_2)g_{A3}, \quad b^{-\ell}B_3 = (b^\ell A_2 + b^{-\ell}B_2)g_{B3}$$

with $g_{A3} = h(R/b)$ and $g_{B3} = h(b/R)$ with $h(x) = 1/(1-x^{2\ell})$. We thus obtain (8.5.4).

We finally have to determine A_2 and B_2 such that equation (8.5.1) is also valid at $r = a$ and $r = b$. The jump conditions writes :

$$\frac{\ell\Phi_\ell}{\omega - \ell\partial_r\Phi_0/r}[\rho_0] = [r\partial_r\Phi_\ell].$$

The jump condition at $r = a$ writes :

$$\begin{aligned}
\frac{\ell}{\omega - \ell\partial_r\Phi_0(a)/a}(a^\ell A_2 + a^{-\ell}B_2) &= [r\partial_r\Phi_\ell(r)]_a \\
&= -\ell\left[a^\ell(A_1 - A_2) - a^{-\ell}(B_1 - B_2)\right] \\
&= -\ell\left[(a^\ell A_2 + a^{-\ell}B_2)g_{A1} - a^\ell A_2 - (a^\ell A_2 + a^{-\ell}B_2)g_{B1} + a^{-\ell}B_2\right] \\
&= -\ell\left[a^\ell A_2(g_{A1} - g_{B1} - 1) + a^{-\ell}B_2(g_{A1} - g_{B1} + 1)\right] \\
&= -\ell\left[a^\ell A_2 f_{a,-} + a^{-\ell}B_2 f_{a,+}\right],
\end{aligned}$$

with $f_{a,\pm} = (g_{A1} - g_{B1} \pm 1)$. We have $f_{a,+} = 2, f_{a,-} = 0$: we thus obtain (8.5.5). However for keeping the symmetries of the expression, we do not simplify it further for the moment. We thus obtain :

$$a^\ell\Big(1 + (\omega - c_a)f_{a,-}\Big)A_2 + a^{-\ell}\Big(1 + (\omega - c_a)f_{a,+}\Big)B_2 = 0,$$

with $c_a = \ell\partial_r\Phi_0(a)/a$.

The jump condition at $r = b$ writes :

$$\begin{aligned}
\frac{-\ell}{\omega - \ell\partial_r\Phi_0(b)/b}(b^\ell A_2 + b^{-\ell}B_2) &= [r\partial_r\Phi_\ell(r)]_b \\
&= \ell\left[b^\ell(A_3 - A_2) - b^{-\ell}(B_3 - B_2)\right] \\
&= \ell\left[(b^\ell A_2 + b^{-\ell}B_2)g_{A3} - b^\ell A_2 - (b^\ell A_2 + b^{-\ell}B_2)g_{B3} + b^{-\ell}B_2\right] \\
&= \ell\left[b^\ell A_2(g_{A3} - g_{B3} - 1) + b^{-\ell}B_2(g_{A3} - g_{B3} + 1)\right] \\
&= \ell\left[b^\ell A_2 f_{b,-} + b^{-\ell}B_2 f_{b,+}\right],
\end{aligned}$$

with $f_{b,\pm} = g_{A3} - g_{B3} \pm 1$. This equation writes :

$$b^\ell\Big(1 + (\omega - c_b)f_{b,-}\Big)A_2 + b^{-\ell}\Big(1 + (\omega - c_b)f_{b,+}\Big)B_2 = 0,$$

with $c_b = \ell\partial_r\Phi_0(b)/b$.

We have two linear equations with two unknowns A_2 and B_2 and ω is such that this system has solutions, i.e. the determinant of the linear system vanishes :

$$\frac{a^\ell}{b^\ell}\Big((1-c_a f_{a,-})+\omega f_{a,-}\Big)\Big((1-c_b f_{b,+})+\omega f_{b,+}\Big) - \frac{b^\ell}{a^\ell}\Big((1-c_a f_{a,+})+\omega f_{a,+}\Big)\Big((1-c_b f_{b,-})+\omega f_{b,-}\Big) = 0.$$

This is a second-degree polynomial equation in ω :

$$\alpha_2\omega^2 + \alpha_1\omega + \alpha_0 = 0,$$

with

$$\alpha_2 = \Big(\frac{a^\ell}{b^\ell} f_{a,-} f_{b,+} - \frac{b^\ell}{a^\ell} f_{a,+} f_{b,-}\Big),$$
$$\alpha_1 = \frac{a^\ell}{b^\ell}\Big(f_{a,-}(1 - c_b f_{b,+}) + f_{b,+}(1 - c_a f_{a,-})\Big) - \frac{b^\ell}{a^\ell}\Big(f_{b,-}(1 - c_a f_{a,+}) + f_{a,+}(1 - c_b f_{b,-})\Big),$$
$$\alpha_0 = \frac{a^\ell}{b^\ell}(1 - c_a f_{a,-})(1 - c_b f_{b,+}) - \frac{b^\ell}{a^\ell}(1 - c_a f_{a,+})(1 - c_b f_{b,-}).$$

Since $f_{a,-} = 0$ and $f_{a,+} = 2$, we recover (8.5.6)-(8.5.9). □

If ω is a solution of equation 8.5.6, then the case $\mathrm{Im}(\omega) > 0$ corresponds to instability. Consider $\Delta = \alpha_1^2 - 4\alpha_0$. With $\Delta > 0$ we obtain real solutions of equation (8.5.6), and we have the stability. But when $\Delta < 0$ or $\alpha_1^2 < 4\alpha_0$, then the solutions of equation (8.5.6)are complex. So the condition for stability is $\alpha_1^2 - 4\alpha_0 \geq 0$ or

$$\Big(\frac{\ell}{2}(1 - (a/b)^2) + \frac{1}{2}\Big((b/R)^{2\ell} - (a/R)^{2\ell}\Big)\Big)^2 + \big(1 - \frac{a^{2\ell}}{b^{2\ell}}\big)(1 - \frac{b^{2\ell}}{R^{2\ell}}) - \ell\big(1 - (\frac{a}{b})^2\big)\big(1 - (\frac{a}{R})^{2\ell}\big) \geq 0 \tag{8.5.14}$$

So, with each value of mode ℓ, we have a difference domain of stable or unstable. Figure 8.5.2 illustrate the domain of stable or unstable problem which depends on two variable $x := a/R$ and $y := b/R$. Note that $0 < a < b < R$, so $0 < x < y < 1$. In this figure, we display zone of stable-unstable with mode $\ell = 2, 4, 6, 7$. Instability is when the value of (x, y) below the curve and above the line $y = x$, otherwise, we have the stability. For exemple, with the mode $\ell = 2$ if we take the couple $(x, y) = (0.1, 0.8)$, we have then the stability, but with $(0.1, 0.3)$, the problem is unstable.

Remark 8.5.2.
- *With the mode $\ell = 1$, we always have $\alpha_1^2 - 4\alpha_0 \geq 0$ which means that the system is stable with the mode 1.*
- *In case of instability, when the thickness of diocotron $b - a$ decreases, it increases the instability mode.*
- *We observe that when a tends to 0 or b tends to R, the instability growth rate is reduced.*

8.5.3 Numerical results

We consider the initial density as follows

$$\rho_0(r, \theta) = \begin{cases} 0, \ 0 \leq r < a, \\ 1 + \varepsilon \cos(\ell\theta), \ a \leq r \leq b, \\ 0, \ b < r \leq R. \end{cases} \tag{8.5.15}$$

where $\varepsilon > 0$ a small parameter of perturbation and ℓ the mode. This is perturbation of the stationary state (8.5.2). We have tested the code with mode $1, 2, \ldots, 7$

In Fig.8.5, we compute the growth rate for the diocotron instability test case. We obtain the good rate. Indeed, the numerical results show that for the mode $\ell = 3$, the growth rate

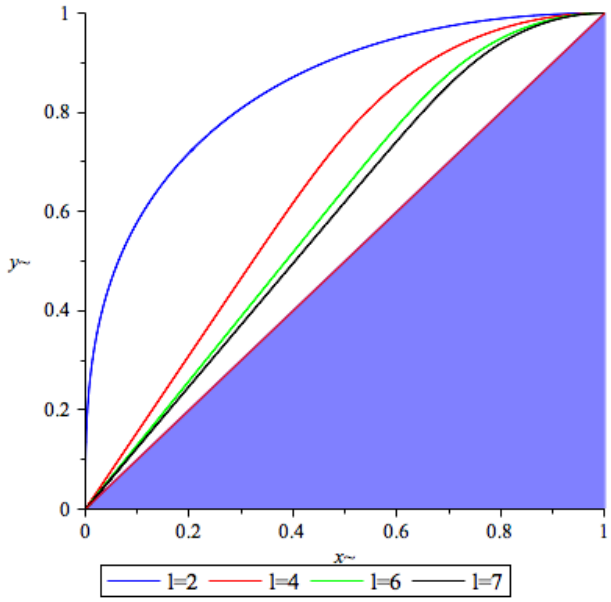


FIGURE 8.4 : Stability-instability curve in $(a/R, b/R)$ plane for mode numbers $\ell = 2, 4, 6, 7$. Instability is obtained for values $(a/R, b/R)$ between the curve and the straight line $y = x$.

$\omega = 0.118$, the growth rate of instability with the mode $\ell = 4$ is 0.14665 and for the mode 7, the rate is 0.1734, which is consistent with the growth rate obtained by the analytic solutions from equation 8.5.6. It is computed by a small Maple code in annex D. Notice that for each case, we have chosen different parameters for a, b because the growth rate also depends on a, b. For example, in the case $a = 5.75, b = 6.6$ we have a small rate for the mode 3 (< 0.03). In the case $a = 4.5, b = 5.5$ the mode 7 produces a rate of 0.094, which is not visible.

When verifying numerically the growth rate, we remark that our code better captures mode $\ell = 4$ (see Fig.8.6). It is may be due to the fact that we use a mesh with a square macro mesh at the center and four neighbors macro meshes around (see Fig.8.4).

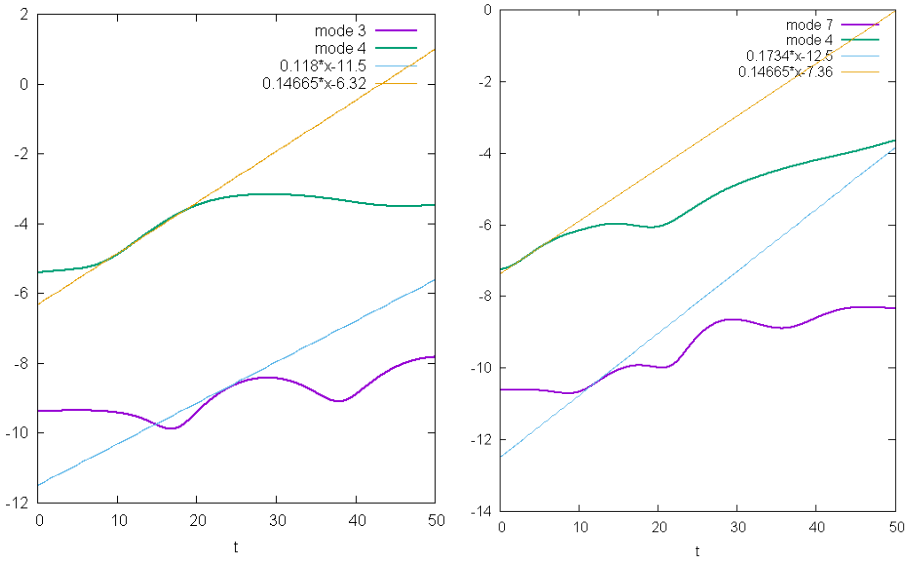
Remark 8.5.3. *In practice, in order to obtain the correct growth rate, we have to refine in geometry mesh. This fact admits to decrease the geometry error and then admit that the geometry error not affect so much in the numerical solution.*

Now, let us take the parameters : $a = 4, b = 5, \varepsilon = 10^{-6}$ for the diocotron test case. In Fig. 8.7, we present the charge density ρ at to time $t = 100$ with two different exciting modes : $\ell = 3$ and $\ell = 6$.

8.6 Parallelization

An advantage of the schnaps library is that it provides parallelization the code thanks to the OpenCL library. OpenCL allows to write parallel algorithms that work both on GPU or multicore CPU (see Chapter 6).

Let us evaluate the performance of the parallelization on the transport in space test-case.



Mode 3 and 4 Mode 4 and 7

FIGURE 8.5 : Growth rate instability of diocotron, capture with the initial function $(1 + \varepsilon\cos(\theta))$. Parameters : Left $\mathbf{d_x} = 2, M_{\mathbf{x}} = 16, a = 4.5, b = 5.5, R = 10, \varepsilon = 10^{-3}$. Right $\mathbf{d_x} = 2, M_{\mathbf{x}} = 16, a = 5.75, b = 6.6, R = 10, \varepsilon = 10^{-3}$ (and with 1 sub-meshes in each macro-mesh.)

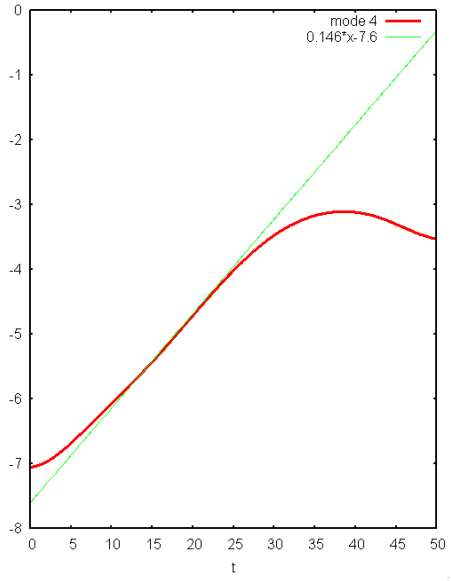
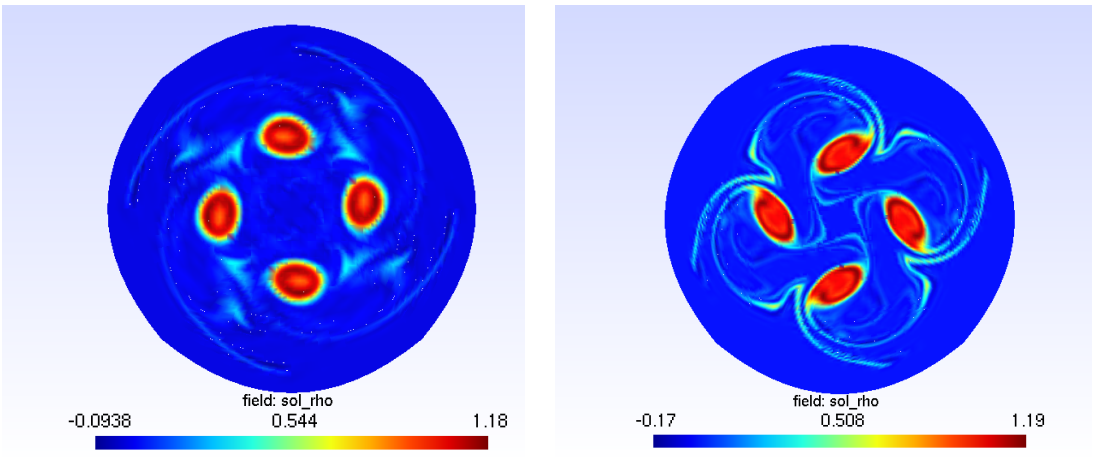


FIGURE 8.6 : Growth rate instability of diocotron, 4^{th} mode capture with the initial function $(1 + \varepsilon\cos(4\theta))$. Parameters : d $= 2, M_{\mathbf{x}} = 8, a = 4.5, b = 5.5, R = 10, \varepsilon = 10^{-3}$ and with 4 sub-meshes in each macro-mesh.



Mode 6 Mode 3

FIGURE 8.7 : The charge density of diocotron instability test case at time $t = 100$. Parameters : Left $\mathbf{d_x} = 3, M_\mathbf{x} = 16$, right : $\mathbf{d_x} = 3, M_\mathbf{x} = 30$.

We consider the following equation

$$\partial_t f + v_{||}\partial_{x_{||}} f = 0,$$

on the cylinder. We consider the following size of problem : the cylinder is composed of 5 macro-cells. In each spatial direction, we consider $M_\mathbf{x} = 20$ elements and the degree of the DG scheme d = 3. We thus have $((3+1)\times 20)^3 = 512\,000$ Gauss points in each macro-cells. In the one-dimensional velocity direction, we consider 10 cells and the degree of the finite element basis is taken equal to $d = 3$. At each Gauss point in space, we thus have : $31 = (10 \times 3 + 1)$ kinetic variables. The total number of discretization points is

$$N = 512\,000 \times 31 \times 5 = 79\,360\,000.$$

We execute the program on the Irma Atlas cluster hosted at the IRMA laboratory (Institut de Recherche Mathématique Avancée, UMR 7501, Strasbourg). We compare the computational time of the sequential code (without parallelization) with the one of the OpenCL parallelized code. We run 20 iterations of the code and we provide the results in table 8.1. We observed that, even using only one core (Intel Xeon E5-2680 v3, 24 cores hyperthreaded, 2.50GHz), the OpenCL parallelized code is 1.5 times faster than the sequential code. This is due to the optimized access to the memory. We next compare it the computational time on one GPU (NVIDIA K80 : 2496×2 cores, 560 MHz, 24 GB). We note that the execution on GPU is about 53.5 times faster than one core CPU. Further simulation should be carried out for simulations including the resolution of the quasineutrality equation.

We then study the scalability of the OpenCL code on a multi-CPU. We run 20 iterations of the code (with the same parameters as previously) on a node of Irma Atlas cluster composed of 4 processor units (AMD Opteron 6386 SE, 2,8 GHz, 16 cores) : we thus have 64 cores and 512 GB of RAM. In table 8.2, we compute the computational time and the speedup. The speed-up is not as optimal than the one of the MPI code developed in chapters 5 and 6. Future development would be to use the StarPU library to combine multi-CPU simulations [4] and GPU acceleration.

	sequential code 1 core	OpenCL code 1 core	OpenCL code 1GPU
20 iterations	3532	2401.3	39.2
1 iterations	168.19	114.34	1.8
relative speed-up		1.47	63.5

TABLE 8.1 : Computational time and relative speed-up for the 2D `schnaps` code on GPU and CPU. Computed on the Irma Atlas cluster. Parameters : 20 iterations, $M_x = 20$ $\mathrm{d}_x = 3$, $N_v = 31$ ($\mathrm{d}_v = 3$, $M_v = 20$).

Nb of processor units (CPU)	1	2	4	8	16	32	64
computation time (second)	2401.3	1396.125	801.39	504.6	276.89	193.9	142.48
speed-up		1.72	2.997	4.759	8.67	12.38	16.85

TABLE 8.2 : Computational time and speedup for the 2D `schnaps` code on CPU. Computed on the Irma Atlas cluster. Parameters : 20 iterations, $M_x = 20$ $d_x = 3$, $N_v = 31$ ($d_v = 3$, $M_v = 20$).

8.7 Conclusion

In this chapter, we have presented a discontinuous Galerkin scheme for the drift-kinetic model based on the reduction strategy. We have implemented this scheme in `schnaps` and we also develop a finite element solver for the (quasineutral) Poisson equation. We first validate the code for the guiding center model : we recover the right growth rate for the diocotron test-case if the geometry domain is well approximated by the computational domain. We also test the parallelization of the transport term. We next plan to test the full drift-kinetic code using the ion turbulence test-case, as presented in [72]. For this test case, the initial distribution function is given by

$$f(t=0,r,\theta,z,v) = f_{eq}(r,v)\left[1+\varepsilon\exp\left(-\frac{(r-r_p)^2}{\delta r}\right)\cos\left(\frac{2\pi n}{L}z+m\theta\right)\right],$$

with $n, m \in \mathbb{N}$, $L, r_p > 0$, $\varepsilon \ll 1$ and where $f_{eq}(r,v)$ denotes the equilibrium distribution :

$$f_{\mathrm{eq}}(r,v) = \frac{n_0(r)\exp\left(-\frac{v^2}{2T_i(r)}\right)}{\left(2\pi T_i(r)\right)^{1/2}}.$$

and where the profils $\{T_i, T_e, n_0\}$ are defined by

$$\mathcal{P}(r) = \mathcal{C}_{\mathcal{P}}\exp\left(-\kappa_{\mathcal{P}}\,\delta r_{\mathcal{P}}\,\tanh\left(\frac{r-r_p}{\delta r_{\mathcal{P}}}\right)\right), \quad \mathcal{P}\in\{T_i,T_e,n_0\}.$$

with $\kappa_{\mathcal{P}}, \delta r_{\mathcal{P}}, C_{\mathcal{P}} > 0$, are given parameters. Finally, we would like to use time implicit solver for removing the CFL stability condition.

Conclusion générale & Perspectives

Conclusion générale

Dans ce manuscrit, nous avons présenté un modèle réduit pour l'équation de Vlasov (présenté au chapitre 2), obtenu en appliquant une méthode d'éléments finis en vitesse. Nous avons ensuite proposé et étudié différentes méthodes numériques pour résoudre ce modèle.

Une base d'éléments finis continus de type Lagrange (tensoriel) est utilisée pour semi-discrétiser l'équation de Vlasov en vitesse. Les degrés de liberté, correspondant aux valeurs nodales de la fonction distribution, satisfont un système dans les variables d'espace et de temps, appelé équation de Vlasov réduite. Nous avons démontré quelques propriétés pour l'équation de Vlasov réduite : l'hyperbolicité, la stabilité en norme L^2 ainsi que la conservation de la charge, de l'énergie totale et de la quantité de mouvement. Nous avons démontré que l'ordre de convergence de la méthode des éléments finis en vitesse est d où d est le degré de base en vitesse.

Nous avons ensuite, dans les chapitres 3 et 4, utilisé une méthode de volumes finis en espace et la méthode de Runge-Kutta (RK) en temps pour le système Vlasov réduit 1D (chap. 3) et également pour le modèle de Vlasov-Fourier en vitesse (chap. 4). Nous démontrons que le flux centré (d'ordre 2 en espace) ne peut pas être utilisé avec les schémas RK1 ou RK2 car les schémas sont alors instables. Pour le flux visqueux (d'ordre 1 espace), les schémas RK3 et RK4 sont stables sous condition CFL tandis que les schémas RK1 et RK2 nécessitent en plus une diffusion numérique artificielle suffisamment grande. Nous avons ensuite vérifié numériquement l'ordre de convergence en vitesse et espaces des schémas. Nous avons validé également nos codes sur les cas-tests classiques en physique des plasmas (l'amortissement Landau et l'instabilité double faisceaux) et en les comparant au résultat de la méthode PIC. Notons que la méthode assure automatiquement la stabilité en norme L2 mais par contre ne garantit pas la positivité de la fonction de distribution.

Nous avons ensuite mis en oeuvre la méthode numérique pour l'équation de Vlasov $2D$ au sein de la bibliothèque `SELALIB`. Le code a été parallélisé avec MPI (par décomposition de domaine) pour réduire le temps de calcul. Nous avons considéré deux schémas numériques : un schéma volumes finis et un schéma semi-Lagrangien. Pour ce dernier, il est intéressant de prendre les points de Gauss-Lobatto (en vitesse) comme points de quadrature, car ainsi le système hyperbolique est diagonalisé (technique de "mass-lumping"). La méthode semi-Lagrangienne n'est pas contrainte par une condition de stabilité, et donc des grands pas de temps peuvent être utilisés. Nous avons enfin testé la scalabilité des codes sur le super-calculateur Curie (TGCC). Pour les deux schémas, l'accélération obtenue est raisonnable.

Dans la dernière partie, nous avons considéré un schéma de type Galerkin Discontinu pour résoudre le système de Vlasov-Maxwell relativiste et le modèle drift-kinetic, pour lesquels

nous avons également effectué une réduction par éléments finis en vitesse. Noter que pour le modèle Vlasov-Maxwell, un seul code est nécessaire pour résoudre les deux parties. Ce schéma permet de plus d'être à la fois d'être d'ordre élevé (en espace) et de pouvoir considérer des domaines courbes. Nous avons mis en oeuvre ce schéma dans un code parallélisé grâce au langage OpenCL : cela permet de faire à la fois du calcul multi-CPU ou sur GPU (carte graphique). En vue de résoudre le modèle drift-kinetic, nous avons introduit un solveur éléments finis pour résoudre l'équation (elliptique) de quasi-neutralité. Afin de valider notre code, nous avons considéré le cas test du Diocotron pour le modèle centre-guide : nous avons présenté le calcul du taux d'instabilité (ou de stabilité) et nous l'avons comparé avec le taux d'instabilité numérique.

Perspectives

1. Pour le modèle de type Vlasov-Fourier en vitesse, il serait intéressant de considérer la condition aux bords en vitesse proposée dans [37] et de la comparer avec la condition au bord que nous utilisons. Par ailleurs, nous envisageons également d'utiliser une condition de type couche absorbante (PML) pour gérer la sortie des hautes fréquences en vitesse. Enfin, nous pourrions développer la méthode en dimension 2.

2. Suite au travail effectué dans le chapitre 8, nous souhaitons appliquer la méthode pour résoudre le modèle drift-kinetic et ensuite résoudre le modèle gyro-cinétique (impliquant des gyro-moyennes).

3. Nous pouvons également continuer la comparaison entre les schémas semi-Lagrangien (développé dans `SELALIB`) avec les schémas Galerkin discontinue (développé dans `schnaps`) pour le modèle drift-kinetic. Ceci permettrait de comparer plus finement les propriétés des schémas à ordre d'approximation égal.

4. Nous souhaitons étudier le modèle de Vlasov avec un terme de collision comme présenté au chapitre 2. Cela permettrait de faire du couplage entre un modèle cinétique et un modèle fluide.

Annex

A Computation of zeros and weights for Gauss-Legendre and Gauss-Lobatto integration. Maple code

The following Maple code computes the zeros and the weights for the Gauss-Legendre and Gauss-Lobatto and the value of Lagrange polynomials and the derivetive of the Lagrange polynomial in each node.

```
> restart: Digits:=10:
> d := 2:
> with(CodeGeneration):
> G := n->expand(1/2^n/(n!)*diff((x^2-1)^n,seq(x,i=1..n))*sqrt((2*n+1)/2)):
> gauss := sort(map(Re,map(evalf,[fsolve(G(d+1),x)]))):
> writeto("./legendre.f90"): Fortran(gauss, resultname = "gauss", precision
= double): writeto(terminal):
> Poids := unapply(evalf(-2/(d+2)/diff(G(d+1),x)/G(d+2)*sqrt(d+1+1/2)*
sqrt(d+1+3/2)) ,x): weight := map(Poids, gauss):
> appendto("./legendre.f90"): Fortran(weight, resultname = "weight",
precision = double): writeto(terminal):
> N := evalf([seq(-1+2/d*(i-1), i = 1 .. d+1)]):
> L := k -> product((x-N[i])/(N[k]-N[i]), i = 1.. k-1) * product((x-N[i])/
(N[k] - N[i]), i = k+1..d+1):
> lag := map(evalf,matrix(d+1,d+1,(i,j) -> evalf(subs(x = gauss[j],L(i))))):
> appendto("./legendre.f90"): Fortran(lag, resultname="lag", precision
= double): writeto(terminal):
> dlag := map(evalf,matrix(d+1,d+1,(i,j) -> evalf(subs(x = gauss[j],
diff(L(i),x))))):
> appendto("./legendre.f90"): Fortran(dlag, resultname = "dlag", precision
 = double): writeto(terminal):
```

B Matrix assembly

The algorithm for assembling the matrices is the following :

1. We loop on the elements of the mesh, we loop on the local nodes of each element. Using the connectivity array, we construct the shape of the sparse matrices. More precisely,

if

$$j_1 = \text{connec}(k_1, i) \text{ and } j_2 = \text{connec}(k_2, i)$$

then the corresponding (j_1, j_2) elements in the matrices are $\neq 0$.

2. We loop again on the elements of the mesh. For each element Q_i we perform the following algorithm

```
do k1 = 1, d+1
    do k2 = 1, d+1
        j1 = connec(k1, i)
        j2 = connec(k2 , i)
        M(j1, j2) = M(j1, j2) + ∫_{v∈Q_i} φ_{j1} φ_{j2}
        A(j1, j2) = A(j1, j2) + ∫_{v∈Q_i} v φ_{j1} φ_{j2}
        B(j1, j2) = B(j1, j2) + ∫_{v∈Q_i} φ_{j1} φ'_{j2}
    enddo
enddo
```

Of course, in this algorithm, the matrices $\mathcal{M}, A$ and B are stored in a sparse way, for saving computer memory. The first and last terms of the diagonal of B have also to be corrected according to (3.1.10) and (3.1.11).

C Non-linear Vlasov reduction

For improving the dissipation, we consider now a source term in the Vlasov equation. It can represent collisions between particles. The Vlasov equation (7.1.6) becomes

$$\partial_t f + v\partial_x f + E\partial_v f = Q(f), \tag{C.1}$$

where $Q(f)$ is a source term, which we denote by the collision kernel. We also consider an entropy $S(f)$. The entropy is supposed to be a smooth strictly convex function of f. In practice, we will consider the following entropies

$$\text{(a) } S(f) = \frac{f^2}{2}, \quad \text{(b) } S(f) = f(\ln f - 1), \quad \text{(c) } S(f) = \frac{f^2}{2} - \varepsilon \ln f, \quad \varepsilon > 0.$$

The entropy variable is

$$g = S'(f).$$

Considering S^*, the Legendre transform of S

$$S^*(g) = \max_f (gf - S(f)),$$

we also have

$$f = S^{*'}(g).$$

In the previous sections, we expanded f on an interpolation basis. Now, we expand the entropy variable g in the Legendre basis. A first advantage is that if $S^{*'}$ is positive, the distribution function is automatically positive. It is the case with choice (b) and (c). With

choice (a) we recover the reduced Vlasov model of the previous sections because then $S'(f) = g = f$.

Suppose that

$$g(x,t,v) = \sum_{k=1}^{P} g_k(x,t)\varphi_k(v). \tag{C.2}$$

We denote by Π the orthogonal projection of g on a subspace $\mathcal{V}_0$ of the interpolation space $\mathcal{V} = \text{span}\{\varphi_i,\ 1 \leq i \leq P\}$. For a given constant $\lambda > 0$, we can then consider the collision kernel

$$Q(f) = \lambda\left(\Pi g - g\right). \tag{C.3}$$

If f has a compact support in v, we can prove the following results

- $\int_v Q(f)\varphi dv = 0, \quad \forall \varphi \in \mathcal{V}_0$. In particular, if $v \to v^m$ is in the space $\mathcal{V}_0$ then the $m-$moment of f is conserved in the sense that

$$\int_x \int_v f v^m dv dx = Cst.$$

- The total entropy $\Sigma = \int_v S(f)dv$ is decreasing

$$\partial_t \Sigma + \partial_x G(\Sigma) \leq 0, \quad \text{with } G(\Sigma) = \int_v vS(f).$$

The proof is an immediate adaptation of techniques presented in many works on non-linear hyperbolic systems and Boltzmann theory. We refer for instance to [12, 85, 78].

The collision kernel (C.3) can be used of course for introducing a physical phenomenon. But we can also use it as a numerical tool for damping numerical oscillations. We will investigate this kind of tools in forthcoming works.

D Computation of the growth rate instability for diocotron test case : Maple code

```
> restart:
> Digits := 30:
> a := 4.5: b := 5.5: R := 10: ell := 4:
> k := r -> (a^2/2)*log(r) - r^2/4:
> w := r -> b*D(k)(b)*log(r/R):
> f1 := r -> k(a)-k(b)+w(b):
> f2 := r -> k(r)-k(b)+w(b):
> f3 := r -> w(r):
> phi0 := r -> piecewise(0 < r  and r <= a, f1, a < r and r <= b,
 f2, b < r and r < R, f3):
> ca := ell*D(f2)(a)/a:
> cb := ell*D(f2)(b)/b:
> f := x -> (x^(2*ell)+1)/(1-x^(2*ell)):
```

```
> fbm := (f(R/b) - 1):
> fbp := (f(R/b) + 1):
> fam := 0 : fap:=2:
> alpha1 := evalf(a^(2*ell)/b^(2*ell)*fbp -  (fbm*(1-2*ca)+2*(1-cb*fbm))):
> alpha0 := evalf(a^(2*ell)/b^(2*ell)*(1-cb*fbp) - (1-2*ca)*(1-cb*fbm)):
> alpha2 := evalf(-2*fbm):
> A := evalf(solve(alpha2*x^2+alpha1*x+alpha0)):
```

Index des notations

Operators

Bibliographie

[1] Curie, supercomputer of genci, cea. `http://www-hpc.cea.fr/en/complexe/tgcc-curie.htm`.

[2] schnaps, solveur pour les lois de conservation hyperboliques non-linéaires appliqué aux plasmas. `http://schnaps.gforge.inria.fr`.

[3] Selalib, a semi-lagrangian library. `http://selalib.gforge.inria.fr/`.

[4] A unified runtime system for heterogeneous multicore architectures. `http://starpu.gforge.inria.fr`.

[5] P Abbott. Tricks of the trade : Legendre-gauss quadrature. *Mathematica Journal*, 9 :689–691, 2005.

[6] Bedros Afeyan, Fernando Casas, Nicolas Crouseilles, Adila Dodhy, Erwan Faou, Michel Mehrenberger, and Eric Sonnendrücker. Simulations of kinetic electrostatic electron nonlinear (keen) waves with variable velocity resolution grids and high-order time-splitting. *The European Physical Journal D*, 68(10) :1–21, 2014.

[7] Grégoire Allaire and Marc Schoenauer. *Conception optimale de structures*, volume 58. Springer, 2007.

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

[9] D. Aubert, M. Amini, and R. David. A Particle-Mesh Integrator for Galactic Dynamics Powered by GPGPUs. *Lecture Notes in Computer Science*, 5544 :874–883, 2009.

[10] A Avdis and SL Mouradian. A gmsh tutorial.

[11] Blanca Ayuso de Dios, C-W Shu, José Antonio Carrillo de la Plata, et al. Discontinuous galerkin methods for the one-dimensional vlasov-poisson system. 2009.

[12] Timothy Barth. On discontinuous galerkin approximations of boltzmann moment systems with levermore closure. *Computer methods in applied mechanics and engineering*, 195(25) :3311–3330, 2006.

[13] Prabhu Lal Bhatnagar, Eugene P Gross, and Max Krook. A model for collision processes in gases. i. small amplitude processes in charged and neutral one-component systems. *Physical review*, 94(3) :511, 1954.

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

[15] Daniel Bouche, G Bonnaud, and D Ramos. Comparison of numerical schemes for solving the advection equation. *Applied mathematics letters*, 16(2) :147–154, 2003.

[16] François Bouchut. *Nonlinear stability of finite Volume Methods for hyperbolic conservation laws : And Well-Balanced schemes for sources*. Springer Science & Business Media, 2004.

[17] F. Bourdel, P. A. Mazet, and P. Helluy. Resolution of the non-stationary or harmonic Maxwell equations by a discontinuous finite element method. Application to an E.M.I. (electromagnetic impulse) case. *Proceedings of the 10th international conference on computing methods in applied sciences and engineering*, pages 405–422, 1992.

[18] Chio-Zong Cheng and Georg Knorr. The integration of the vlasov equation in configuration space. *Journal of Computational Physics*, 22(3) :330–351, 1976.

[19] Yingda Cheng, Irene M Gamba, Fengyan Li, and Philip J Morrison. Discontinuous galerkin methods for the vlasov–maxwell equations. *SIAM Journal on Numerical Analysis*, 52(2) :1017–1049, 2014.

[20] Yingda Cheng, Irene M Gamba, and Philip J Morrison. Study of conservation and recurrence of runge–kutta discontinuous galerkin schemes for vlasov–poisson systems. *Journal of Scientific Computing*, 56(2) :319–349, 2013.

[21] Stephane Clain, Jorge Figueiredo, and Carolina Ribeiro. A finite volume scheme for the shallow-water system with the polynomial reconstruction. 2012.

[22] G. Cohen, X. Ferrieres, and S. Pernet. A spatial high-order hexahedral discontinuous Galerkin method to solve Maxwell's equations in time domain. *Journal of Computational Physics*, 217(2) :340–363, 2006.

[23] Gary Cohen. *Higher-order numerical methods for transient wave equations*. Springer Science & Business Media, 2013.

[24] A. Crestetto. *Optimisation de méthodes numériques pour la physique des plasmas. Application aux faisceaux de particules chargées*. PhD thesis, Université de Strasbourg, 2012.

[25] A. Crestetto and P. Helluy. Resolution of the Vlasov-Maxwell system by PI discontinuous Galerkin method on GPU with OpenCL. In *CEMRACS'11 : Multiscale coupling of complex models in scientific computing*, volume 38 of *ESAIM Proc.*, pages 257–274. EDP Sci., Les Ulis, 2012.

[26] A. Crestetto, P. Helluy, and J. Jung. Numerical resolution of conservation laws with OpenCL. *ESAIM : Proceedings*, 40 :51–62, 2013.

[27] Nicolas Crouseilles, Michel Mehrenberger, and Eric Sonnendrücker. Conservative semi-lagrangian schemes for vlasov equations. *Journal of Computational Physics*, 229(6) :1927–1953, 2010.

[28] Nicolas Crouseilles, Thomas Respaud, and Eric Sonnendrücker. A forward semi-lagrangian method for the numerical solution of the vlasov equation. *Computer Physics Communications*, 180(10) :1730–1745, 2009.

[29] R. C. Davidson. *Physics of non neutral plasmas*. 2013.

[30] Blanca Ayuso De Dios, José A Carrillo, and Chi-Wang Shu. Discontinuous galerkin methods for the multi-dimensional vlasov–poisson problem. *Mathematical Models and Methods in Applied Sciences*, 22(12) :1250042, 2012.

[31] Blanca Ayuso de Dios and Soheil Hajian. High order and energy preserving discontinuous galerkin methods for the vlasov-poisson system. *preprint*, 2012.

[32] Pierre Degond and Francis Filbet. On the asymptotic limit of the three dimensional vlasov-poisson system for large magnetic field : formal derivation. *arXiv preprint arXiv :1603.03666*, 2016.

[33] Bruno Després. Symmetrization of vlasov–poisson equations. *SIAM J. Math. Anal.*, 46(4) :2554–2580, 2014.

[34] Daniele Antonio Di Pietro and Alexandre Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69. Springer Science & Business Media, 2011.

[35] Sever Silvestru Dragomir. *Some Gronwall type inequalities and applications*. Nova Science Pub Incorporated, 2003.

[36] B. Dubroca and J.-L. Feugeas. Etude théorique et numérique d'une hiérarchie de modèles aux moments pour le transfert radiatif. *C.R. Acad. Sci. I-Math.*, 329 :915–920, 1999.

[37] Bengt Eliasson. Outflow boundary conditions for the fourier transformed one-dimensional vlasov–poisson system. *Journal of scientific computing*, 16(1) :1–28, 2001.

[38] Bengt Eliasson. Outflow boundary conditions for the fourier transformed two-dimensional vlasov equation. *Journal of Computational Physics*, 181(1) :98–125, 2002.

[39] Alexandre Ern. *Theory and practice of finite elements*, volume 159. Springer Science & Business Media, 2004.

[40] Alexandre Ern. *Aide-mémoire des éléments finis*. Dunod, 2005.

[41] Francis Filbet and Eric Sonnendrücker. Comparison of eulerian vlasov solvers. *Computer Physics Communications*, 150(3) :247–266, 2003.

[42] Francis Filbet, Eric Sonnendrücker, and Pierre Bertrand. Conservative numerical schemes for the vlasov equation. *Journal of Computational Physics*, 172 :166–187, 2001.

[43] Francis Filbet and Chang Yang. Mixed semi-lagrangian/finite difference methods for plasma simulations. *arXiv preprint arXiv :1409.8519*, 2014.

[44] B. Fornet, V. Mouysset, and Á. Rodríguez-Arós. Mathematical study of a hyperbolic regularization to ensure Gauss's law conservation in Maxwell-Vlasov applications. *Math. Models Methods Appl. Sci.*, 22(4) :1150020, 28, 2012.

[45] André Fortin and André Garon. Les éléments finis : de la théorie à la pratique. *Université Laval*, 2011.

[46] C Kristopher Garrett and Cory D Hauck. A comparison of moment closures for linear kinetic transport equations : The line source benchmark. *Transport Theory and Statistical Physics*, 42(6-7) :203–235, 2013.

[47] Alain Ghizzo, Pierre Bertrand, Magdi Shoucri, Eric Fijalkow, and Marc R Feix. An eulerian code for the study of the drift-kinetic vlasov equation. *Journal of Computational Physics*, 108(1) :105–121, 1993.

[48] Sebastien Guisset, Philippe Helluy, Michel Massaro, Laurent Navoret, Nhung Pham, and Malcolm Roberts. Lagrangian/eulerian solvers and simulations for vlasov-poisson. 2015.

[49] D.R. Hatch, D. del Castillo-Negrete, and P.W. Terry. Analysis and compression of six-dimensional gyrokinetic datasets using higher order singular value decomposition. *J. Comput. Phys.*, 231 :4234–4256, 2012.

[50] P. Helluy. *Résolution numérique des équations de Maxwell harmoniques par une méthode d'éléments finis discontinus. PhD, Sup'aéro, 1994.* PhD thesis, Sup'aéro, 1994.

[51] P. Helluy. A portable implementation of the radix sort algorithm in OpenCL. http ://hal.archives-ouvertes.fr/hal-00596730/PDF/ocl-radix-sort.pdf, 2011.

[52] P. Helluy, L. Navoret, N. Pham, and A. Crestetto. Reduced Vlasov-Maxwell simulations. http ://hal.archives-ouvertes.fr/hal-00957045/PDF/vlamax.pdf, March 2014.

[53] P. Helluy, N. Pham, and A. Crestetto. Space-only hyperbolic approximation of the Vlasov equation. *ESAIM : Proceedings*, 43 :17–36, 2013.

[54] P. Helluy and T. Strub. Multi-GPU numerical simulation of electromagnetic waves. DGA & AxesSim, October 2013.

[55] G. B. Jacobs and J. S. Hesthaven. Implicit-explicit time integration of a high-order particle-in-cell method with hyperbolic divergence cleaning. *Comput. Phys. Comm.*, 180(10) :1760–1767, 2009.

[56] Antony Jameson, Wolfgang Schmidt, Eli Turkel, et al. Numerical solutions of the euler equations by finite volume methods using runge-kutta time-stepping schemes. *AIAA paper*, 1259 :1981, 1981.

[57] Claes Johnson, Uno Nävert, and Juhani Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer methods in applied mechanics and engineering*, 45(1) :285–312, 1984.

[58] Claes Johnson and Juhani Pitkäranta. An analysis of the discontinuous galerkin method for a scalar hyperbolic equation. *Mathematics of computation*, 46(173) :1–26, 1986.

[59] Sébastien Jund. *Méthodes d'éléments finis d'ordre élevé pour la simulation numérique de la propagation d'ondes*. PhD thesis, Université Louis Pasteur-Strasbourg I, 2007.

[60] Ansgar Jüngel. Entropy dissipation methods for nonlinear partial fifferential equations. 2012.

[61] The Khronos Group Inc., 2013. OpenCL documentation. `http://www.khronos.org/`.

[62] A.J. Klimas and W.M. Farrell. A splitting algorithm for vlasov simulation with filamentation filtration. *J. Comput. Phys.*, 110 :150–163, 1994.

[63] A. Klöckner, T. Warburton, J. Bridge, and J. S. Hesthaven. Nodal discontinuous Galerkin methods on graphics processors. *J. Comput. Phys.*, 228(21) :7863–7882, 2009.

[64] LD Landau. The transport equation in the case of coulomb interactions. *Collected Papers of LD Landau, Pergamon press, Oxford*, pages 163–170, 1981.

[65] Guillaume Latu, Nicolas Crouseilles, Virginie Grandgirard, and Eric Sonnendrücker. Gyrokinetic semi-lagrangian parallel simulation using a hybrid openmp/mpi programming. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 356–364. Springer, 2007.

[66] P. D. Lax and R. S. Phillips. Local boundary conditions for dissipative symmetric linear differential operators. *Communications on Pure and Applied Mathematics*, 13(3) :427–455, 1960.

[67] S Le Bourdiec, F De Vuyst, and L Jacquet. Numerical solution of the vlasov–poisson system using generalized hermite functions. *Computer physics communications*, 175(8) :528–544, 2006.

[68] P. Lesaint and P.-A. Raviart. On a finite element method for solving the neutron transport equation. In *Mathematical aspects of finite elements in partial differential equations (Proc. Sympos., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1974)*, pages 89–123. Publication No. 33. Math. Res. Center, Univ. of Wisconsin-Madison, Academic Press, New York, 1974.

[69] Randall J LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge university press, 2002.

[70] Doron Levy and Eitan Tadmor. From semidiscrete to fully discrete : Stability of runge–kutta schemes by the energy method. *SIAM review*, 40(1) :40–73, 1998.

[71] Yuan-yuan Liu, Chi-wang Shu, and Meng-ping Zhang. On the positivity of linear weights in weno approximations. *Acta Mathematicae Applicatae Sinica, English Series*, 25(3) :503–538, 2009.

[72] Eric Madaule, Sever Adrian Hirstoaga, Michel Mehrenberger, and Jérôme Pétri. Semi-lagrangian simulations of the diocotron instability. 2013.

[73] Éric Madaule, Marco Restelli, and Eric Sonnendrücker. Energy conserving discontinuous galerkin spectral element method for the vlasov–poisson system. *Journal of Computational Physics*, 279 :261–288, 2014.

[74] Jessy Mallet, Stéphane Brull, and Bruno Dubroca. General moment system for plasma physics based on minimum entropy principle. *Kinet. Relat. Models*, 8(3) :533–558, 2015.

[75] C.-D. Munz, P. Omnes, R. Schneider, E. Sonnendrücker, and U. Voß. Divergence Correction Techniques for Maxwell Solvers Based on a Hyperbolic Model. *Journal of Computational Physics*, 161(2) :484–511, 2000.

[76] Laurent Navoret. *Méthodes asymptotico-numériques pour des problèmes issus de la physique des plasmas et de la modélisation des interactions sociales*. PhD thesis, Université Paul Sabatier-Toulouse III, 2010.

[77] Joseph T Parker and Paul J Dellar. Fourier–hermite spectral representation for the vlasov–poisson system in the weakly collisional limit. *Journal of Plasma Physics*, 81(02) :305810203, 2015.

[78] B Perthame. Boltzmann type schemes for gas dynamics and the entropy property. *SIAM Journal on Numerical Analysis*, 27(6) :1405–1421, 1990.

[79] V. M. Petkov and L. N. Stoyanov. *Geometry of reflecting rays and inverse spectral problems*. Pure and Applied Mathematics (New York). John Wiley & Sons Ltd., Chichester, 1992.

[80] Nhung Pham, Philippe Helluy, and Laurent Navoret. Hyperbolic approximation of the fourier transformed vlasov equation. *ESAIM : Proceedings and Surveys*, 45 :379–389, 2014.

[81] Martin Campos Pinto, Eric Sonnendrücker, Alex Friedman, David P Grote, and Steve M Lund. Noiseless vlasov–poisson simulations with linearly transformed particles. *Journal of Computational Physics*, 275 :236–256, 2014.

[82] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery. *Numerical recipes in C*, volume 2. Citeseer, 1996.

[83] Jing-Mei Qiu and Andrew Christlieb. A conservative high order semi-lagrangian weno method for the vlasov equation. *Journal of Computational Physics*, 229(4) :1130–1149, 2010.

[84] P-A Raviart and JM Thomas. Primal hybrid finite element methods for 2nd order elliptic equations. *Mathematics of computation*, 31(138) :391–413, 1977.

[85] Pierre-Arnaud Raviart. *Numerical approximation of hyperbolic systems of conservation laws*, volume 118. Springer Science & Business Media, 1996.

[86] James A Rossmanith and David C Seal. A positivity-preserving high-order semi-lagrangian discontinuous galerkin scheme for the vlasov–poisson equations. *Journal of Computational Physics*, 230(16) :6203–6232, 2011.

[87] Frédéric Rotella and Pierre Borne. *Théorie et pratique du calcul matriciel*, volume 6. Editions technip, 1995.

[88] J.W. Schumer and J.P. Holloway. Vlasov simulations using velocity-scaled hermite representations. *J.Comput.Phys.*, 144 :626–661, 1998.

[89] Eric Sonnedrücker. *Numerical methods for the Vlasov equations.*

[90] Eric Sonnedrücker. Numerical methods for hyperbolic systems. 2013.

[91] Eric Sonnendrücker, Jean Roche, Pierre Bertrand, and Alain Ghizzo. The semi-lagrangian method for the numerical resolution of vlasov equations. 1998.

[92] Christophe Steiner. *Résolution numérique de l'opérateur de gyromoyenne, schémas d'advection et couplage : applications à l'équation de Vlasov.* PhD thesis, Université de Strasbourg, 2014.

[93] Christophe Steiner, Michel Mehrenberger, and Daniel Bouche. A semi-lagrangian discontinuous galerkin convergence. Technical report, HAL preprint, hal, 2013.

[94] Thomas Strub. *Resolution of tridimensional instationary Maxwell's equations on massively multicore architecture.* Theses, Université de strasbourg, March 2015.

[95] Thomas Strub, Nathanaël Muot, and Philippe Helluy. Méthode galerkin discontinue appliquéea l'électromagnétisme en domaine temporel.

[96] Yoshifumi Suzuki. *Discontinuous Galerkin methods for extended hydrodynamics.* ProQuest, 2008.

[97] Eleuterio F Toro. *Riemann solvers and numerical methods for fluid dynamics : a practical introduction.* Springer Science & Business Media, 2013.

[98] David Tskhakaya and Ralf Schneider. Optimization of pic codes by improved memory management. *Journal of Computational Physics*, 225(1) :829–839, 2007.

[99] Eric W Weisstein. Legendre-gauss quadrature. *From MathWorld–A Wolfram Web Resource, http ://mathworld. wolfram. com/Legendre-GaussQuadrature. html, Accessed*, 30, 2008.

[100] Haiping Ye, Jianming Gao, and Yongsheng Ding. A generalized gronwall inequality and its application to a fractional differential equation. *Journal of Mathematical Analysis and Applications*, 328(2) :1075–1081, 2007.

[101] M Zerroukat, N Wood, and A Staniforth. The parabolic spline method (psm) for conservative transport problems. *International journal for numerical methods in fluids*, 51(11) :1297–1318, 2006.

Beaucoup de méthodes numériques ont été développées pour résoudre l'équation de Vlasov car obtenir des simulations numériques précises en un temps raisonnable pour cette équation est un véritable défi. Cette équation décrit en effet l'évolution de la fonction de distribution de particules (électrons/ions) qui dépend de 3 variables d'espace, 3 variables de vitesse et du temps. L'idée principale de cette thèse est de réécrire l'équation de Vlasov sous forme d'un système hyperbolique par semi-discrétisation en vitesse. Cette semi-discrétisation est effectuée par méthode d'éléments finis. Le modèle ainsi obtenu est appelé équation de Vlasov réduite. Nous proposons différentes méthodes numériques pour résoudre efficacement ce modèle.
Dans cette thèse, nous appliquons cette démarche à différents contextes. Nous commençons par l'étude du modèle Vlasov-Poisson en 1D (chapitre 3), puis nous adaptons la méthode au modèle de Vlasov-Poisson avec la transformation de Fourier en vitesse (chapitre 4). Ensuite, nous mettons en oeuvre la méthode pour le système Vlasov-Poisson 2D dans la bibliothèque `SELALIB` (chapitre 5 et 6). Finalement, nous nous intéressons aux modèles de Vlasov-Maxwell (chapitre 7 et aux modèles Drift-Kinetic (chapitre 8) développés dans le code `schnaps`.

INSTITUT DE RECHERCHE MATHÉMATIQUE AVANCÉE
UMR 7501
Université de Strasbourg et CNRS
7 Rue René Descartes
67 084 STRASBOURG CEDEX

Tél. 03 68 85 01 29
Fax 03 68 85 03 28

www-irma.u-strasbg.fr
irma@math.unistra.fr

IRMA (2016/009)
http ://tel.archives-ouvertes.fr/tel-01412750

ISSN 0755-3390