INT 209/03 October 2003

REPORT ON THE SEMI-LAGRANGIAN CODE FOR ITG STUDIES

M. Brunetti, O. Sauter, J. Vaclavik and L. Villard

Contents

Introduction

This report summarises the work performed during summer 2003 at the Centre de Recherches en Physique des Plasmas (EPFL) on the development of a semi-Lagrangian code. This code is part of a project with the long term aim of studying gyrokinetic turbulence and anomalous transport in fusion devices.

Today's supercomputers allow us to consider the semi-Lagrangian method (which computes the distribution function on a fixed grid by tracing back in time the characteristics) as a good alternative to the Particle-in-Cell (PIC) approach (which is commonly used for gyrokinetic studies). The main advantages of the semi-Lagrangian method with respect to PIC codes are that (i) it is not affected by statistical noise and (ii) it gives informations of the particles distribution function in all phase space coordinates on a structured grid.

The present version of the code (developed in collaboration with CEA-Cadarache) globally models the collisionless electrostatic ion drift-kinetic equations in a straight cylinder configuration. The nonlinear evolution of ions temperature gradients (ITG) driven instabilities is studied assuming a uniform magnetic field, $\vec{B} = B\vec{e}_z$, adiabatic electrons and neglecting finite Larmor radius effects. In these hypotheses, the ion distribution function is a four-dimensional (4D) function, $f(r, \theta, z, v_{\|})$, and its time evolution is governed by the 4D Vlasov equation non-linearly coupled with the quasi-neutrality (3D) equation.

A previous version of the code was based on the parabolic approximation of particles trajectories and a Newton-Raphson algorithm was used to solve the corresponding implicit scheme. In order to improve the conservation of the constants of motion, the parabolic approximation has been substituted by the direct resolution of the differential equation of the trajectories based on the Bulirsch-Stoer algorithm. This new semi-Lagrangian scheme constitutes the starting point of this work and in this report we will describe the improvements that we have introduced in order to obtain a code able to properly study the saturation phase of the ITG instability.

In Chapter 1, we will analyse the results obtained with the new semi-Lagrangian scheme, and we will discuss the open problems to be solved: (i) the appearance of negative values in the ion distribution function during the saturation phase, and (ii) the necessity of increasing phase space resolution. In Chapter 2, we will compare different numerical techniques used to address the problem of negative values in f. In Chapter 3, we will present the first results obtained with increased resolution using a non-equidistant mesh in r and $v_{\|}$ directions.

Chapter 1

The new semi-Lagrangian scheme

The time evolution of the ions distribution function $f(r,\theta,z,v_{\|})$ is governed by the 4D Vlasov equation

$$\frac{\partial f}{\partial t} + \vec{v}_{GC} \cdot \vec{\nabla}_{\perp} f + v_{\|}\frac{\partial f}{\partial z} + \dot{v}_{\|}\frac{\partial f}{\partial v_{\|}} = 0 \tag{1.1}$$

where $\vec{\nabla}_{\perp} = (\partial/\partial r, (1/r)\partial/\partial\theta)$, $\vec{v}_{GC} = \vec{E} \times \vec{B}/B^2$, $\vec{B} = B\vec{e}_z$ and $\vec{E} = -\vec{\nabla}\Phi$. Eq. (1.1) is nonlinearly coupled to the quasi-neutrality equation:

$$-\vec{\nabla}_{\perp} \cdot \left[\frac{n_0(r)}{B\Omega}\vec{\nabla}_{\perp}\Phi\right] + \frac{en_0(r)}{T_e(r)}(\Phi - <\Phi>) = n_i(r,\theta,z,t) - n_0(r) \tag{1.2}$$

where $\Omega = q_i B/(m_i)$ is the ion cyclotron frequency (q_i and m_i are the ion charge and mass), n_i is the ion density, n_0 is the initial density profile, T_e is the electron temperature and $< . >$ gives the averaged along the magnetic field lines. Eq. (1.2) is Fourier transformed in θ and z, and it is solved by a standard finite element procedure where the electrostatic potential is discretized along r as a sum over a finite element (cubic spline) basis.

Eq. (1.1) is solved using a time-splitting algorithm which allows us to reduce the 4D equation (1.1) to a sequence of 2D and 1D equations:

$$\frac{\partial f}{\partial t} + \vec{v}_{GC} \cdot \vec{\nabla}_{\perp} f = 0 \tag{1.3}$$

$$\frac{\partial f}{\partial t} + v_{\parallel} \frac{\partial f}{\partial z} = 0 \tag{1.4}$$

$$\frac{\partial f}{\partial t} + \dot{v}_{\parallel} \frac{\partial f}{\partial v_{\parallel}} = 0 \tag{1.5}$$

A previous version of the code was based on the parabolic approximation of Eq. (1.3) and a Newton-Raphson algorithm was used to solve the corresponding implicit scheme together with a predictor-corrector method (see Refs. [1, 2] and Section 1.2). In order to improve the conservation of the constants of motion[1], the parabolic approximation has been substituted by the direct resolution of Eq. (1.3) based on the Bulirsch-Stoer algorithm.

In Section 1.1, we will discuss basic properties of Hamiltonian systems, in Section 1.2 the semi-Lagrangian scheme based on the predictor-corrector method, in Section 1.3 the order of the new semi-Lagrangian scheme and in Section 1.4 we will present the numerical results obtained using these schemes.

1.1 Hamiltonian systems

The Vlasov equation (1.1) is Hamiltonian and can be put in the form

$$\frac{df}{dt} \equiv \frac{\partial f}{\partial t} + \{H, f\} = 0 \tag{1.6}$$

where $\{,\}$ are the Poisson brackets and H is the Hamiltonian. This equation expresses the fact that the distribution function $f(x, v, t)$ is a Lagrangian invariant, i.e., it is constant along any particle trajectory (Liouville's theorem). As a consequence, the integral over the entire phase space of the distribution function is a constant, as well as the integral of any arbitrary (smooth)

[1]In our code, we follow the time evolution of the following constants of motion: the number of ions $\int f \; dR \; dv_{\parallel}$, the entropy $\int f \; \ln f \; dR \; dv_{\parallel}$, the L2-norm $\int f^2 \; dR \; dv_{\parallel}$ and the total (perturbed) energy of the system $E_t = E_k + E_f$, where the kinetic energy is $E_k = (1/2) \int (f - f_M) \; v_{\parallel}^2 \; dR \; dv_{\parallel}$ (f_M being the equilibrium Maxwellian distribution such that $\int f_M \; dv_{\parallel} = n_0$) and the field energy is $E_f = (1/2) \int (n_i - n_0) \; \Phi \; dR$.

function of f, $C(f)$, since

$$\frac{dC(f)}{dt} \equiv \frac{\partial C(f)}{\partial t} + \{H, C(f)\} = 0 \tag{1.7}$$

Thus, the evolution of the distribution function f, described by the Vlasov equation, is constrained by an infinite number of constants of motion. This implies, for example, that the volume of phase space regions is conserved, and that two phase space regions initially disconnected cannot connect at later times (the number of saddle points is conserved). Since dissipative phenomena are not allowed, an important characteristic of Hamiltonian systems is reversibility, $t \rightarrow -t$.

From a numerical point of view, it is non-trivial to simulate such Hamiltonian systems without introducing spurious dissipative effects, as we will see in the following. Anyway, we will try to take advantage of the fact that, in the absence of collisions, there is an infinite number of conserved quantities, and to construct a Vlasov code in such a way as to preserve a good control of these constants of motion.

1.2 Predictor-corrector method

In this section, we describe the method used in the 4D code to solve the equations of motion in the r-θ plane before the introduction of the Bulirsch-Stoer algorithm.

The equations of motion in r-θ are solved in cartesian coordinates and are given by:

$$\frac{d\vec{X}}{dt} = \vec{v}_{GC}(\vec{X}(t), z(t), t) \tag{1.8}$$

where $\vec{X} = (x, y)$ and the components of the guiding-center velocity are

$$v_{GCx} = E_y/B \tag{1.9}$$
$$v_{GCy} = -E_x/B \tag{1.10}$$

The solution of Eq. (1.8) is obtained to second order accuracy by

$$\vec{X}(t+dt) - \vec{X}(t) = dt\ \vec{v}_{GC}(\vec{X}(t+dt/2), z(t+dt/2), t+dt/2) \tag{1.11}$$

If we suppose that there is a displacement $\vec{d}$ such that $\vec{X}(t) = \vec{X}(t+dt) - \vec{d}$ and $\vec{X}(t+dt/2) = \vec{X}(t+dt) - \vec{d}/2$, Eq. (1.11) becomes

$$\vec{d} = dt\ \vec{v}_{GC}(\vec{X}(t+dt) - \vec{d}/2, z(t+dt/2), t+dt/2) \tag{1.12}$$

which can be solved using the Newton-Raphson algorithm. Thus, we need the electric field components E_x and E_y at time $t + dt/2$ in order to have a second order accurate scheme. These components can be obtained using

- a predictor-corrector method: the predictor gives the particle positions $\vec{X}(t + dt/2)$ (with first-order accuracy in time), from which we can calculate the new distribution function and (from the solution of the quasi-neutrality equation) the electric field at time $t+dt/2$. Finally, the corrector uses the (approximated) field calculated with the predictor to solve Eq. (1.11)
- a leap-frog method (see Ref. [3])

The predictor-corrector method and the time-splitting scheme described in Appendix A give a solution of Eq.(1.1) which is second-order accurate in time (apart from the estimate of $\vec{E}$ obtained with the predictor).

1.3 Bulirsch-Stoer method and its implementation in the semi-Lagrangian code

The Bulirsch-Stoer method (see Ref. [4]) is an extrapolation method for solving initial value problems for ordinary differential equations (ODEs). Consider the first-order differential equation

$$\frac{dy(x)}{dx} = g(x, y) \tag{1.13}$$

where the function g is known. Suppose to know y at the initial time x. The key idea of the Bulirsch-Stoer method is the following: in order to find y at time $x + H$, where H is the time-step, H is divided in different sequences of finer and finer substeps $h = H/n$. The computed results of $y(x + H)$, corresponding to the different sequences, are then extrapolated to $h \to 0$. In the Bulirsch-Stoer method, the integrations are done by the **modified midpoint method**, and the extrapolation technique is **rational function or polynomial extrapolation**.

The modified midpoint method advances $y(x)$ from a point x to a point

$x + H$ by a sequence of n substeps each of size $h = H/n$ and its formulas are

$$\begin{aligned}
z_0 &= y(x) \\
z_1 &= z_0 + hg(x, z_0) \\
z_{m+1} &= z_{m-1} + 2hg(x + mh, z_m), \qquad \forall m = 1, 2, \ldots, n-1 \\
y(x+H) &\sim y_n = \frac{1}{2}[z_n + z_{n-1} + hg(x + H, z_n)] \qquad (1.14)
\end{aligned}$$

The modified midpoint method is basically a 'centred difference' (second-order) method except at the first and last points.

In order to implement the previous formulas (1.14) into the semi-Lagrangian code, some modifications are necessary. In our code, the particle positions are known at time t_{n+1} (i.e., the grid points) and one wants to find the corresponding positions at the previous time t_n. Thus, the first simple modification is to write (1.14) as:

$$\begin{aligned}
z_0 &= y(x+H) \\
z_1 &= z_0 - hg(x + H, z_0) \\
z_{m+1} &= z_{m-1} - 2hg(x + H - mh, z_m), \qquad \forall m = 1, 2, \ldots, n-1 \\
y(x) &\sim y_n = \frac{1}{2}[z_n + z_{n-1} - hg(x, z_n)] \qquad (1.15)
\end{aligned}$$

Moreover, since in our case g is known only at time t_n, we assume $g(x + H - mh, z_m) = g(x, z_m)$, for all $m = 1, 2, \ldots, n-1$, so that the sequence (1.15) reduces to:

$$\begin{aligned}
z_0 &= y(x+H) \\
z_1 &= z_0 - hg(x, z_0) \\
z_{m+1} &= z_{m-1} - 2hg(x, z_m), \qquad \forall m = 1, 2, \ldots, n-1 \\
y(x) &\sim y_n = \frac{1}{2}[z_n + z_{n-1} - hg(x, z_n)] \qquad (1.16)
\end{aligned}$$

where z_m are calculated explicitly by interpolating (with cubic splines) the terms $g(x, z_m)$. All the steps in the sequence (1.16) are now of first order in time, and thus the global scheme is of first order, as confirmed by Ref. [3]).

Despite the fact that the scheme (1.16) is of first order in time, the global splitting scheme provides a solution of Eq. (1.1) which is more accurate than the one obtained using the predictor-corrector method together with a Newton-Raphson algorithm. Why?

The reason is that in our problem the function $g(x,y)$ in Eqs. (1.16) corresponds to the electric field components, $E_r(r,\theta,z,t)$ and $E_\theta(r,\theta,z,t)$. Thus, one has to provide the first derivatives of the potential field Φ. Since Φ is calculated with cubic splines, its first derivatives are obtained using quadratic splines.

On the other hand, for the Newton-Raphson algorithm it is necessary to provide the second derivatives of Φ, which are obtained in our code using linear splines. Thus, even if the Newton-Raphson algorithm allows one to obtain a second-order scheme in time, it is less accurate in space.

1.4 Numerical results with $N = 64$ in each direction

In this section, we describe numerical results obtained with the following parameters (RUN1):

- phase space resolution: $N_r = N_\theta = N_z = N_{v_\parallel} = 64$
- Geometrical configuration: $r_a/\rho_{i0} = 14.5$, where $\rho_{i0} = m_i v_{thi0}/(qB)$ and $v_{thi0} = \sqrt{k_B T_{i0}/m_i}$ (T_{i0} being the ion temperature at the peak, $T_{i0} = T_i(0.5r_a) = T_e$); cylinder length: $L_z/\rho_{i0} = 1508$
- density and ion temperature profiles such that $L_{n_0} = 1/\nabla \ln n_0 = 1.25r_a$, $L_{T_i} = 1/\nabla \ln T_i = 0.25r_a$ at the peak
- time step: $dt\Omega^{-1} = 0.2$ (in the following, time is normalised to the inverse of the ion cyclotron frequency Ω); since the convergence studies of the code with the time step have not been performed yet, we set dt to a very small value so that $dt \ll |\omega_{3m}|^{-1} \simeq 80$, where ω_{3m} is the frequency of the most linearly unstable ITG mode (in our case, the mode with $n = 3$; see, for example, Ref. [2] for the frequency values)[2].
- cutoff velocity: $v_{\parallel\ max}/v_{thi0} = 4.15$
- The plasma is initialised by exciting a superposition of ITG modes (m,n) with random phases α_{mn} and amplitudes $0 \leq \epsilon_{mn} \leq 0.001$,

[2]Note also that $dt = 0.2 \ll |\omega_{32m}|^{-1} \simeq 8$, where $n = 32$ is the larger mode studied in simulations with $N_z = 64$.

$f(r,\theta,z,t=0) = f_M[1 + h(v)g(r)\sum_{mn}\epsilon_{mn}\cos(2\pi nz/L_z + m\theta + \alpha_{mn})]$, where $1 \leq m \leq 12$, $1 \leq n \leq 4$, and where h and g are Gaussians

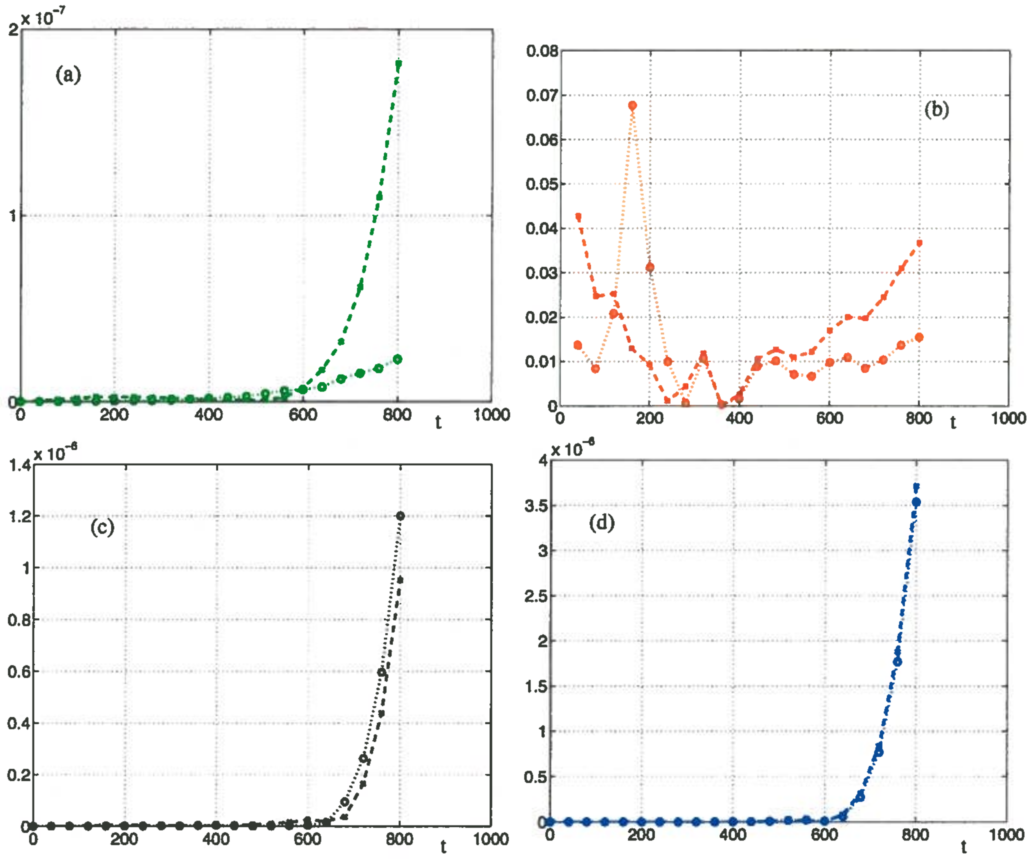

Figure 1.1: Time evolution of the relative error in the number of ions (a), the total energy (b), the entropy (c) and the L2-norm (d) for RUN1 (parameters given in the main text): Bulisch-Stoer method corresponds to dotted lines and predictor-corrector to dashed lines.

The parameters of RUN1 are the same as those of the runs presented in Ref. [3] except for the cutoff velocity, which in Ref. [3] is set to $v_{\parallel\ max}/v_{thi0} = 7.33$.

We compare the results obtained with the Bulirsch-Stoer method given by Eqs. (1.16) (dotted line in Fig. 1.1) and the predictor-corrector method described in Section 1.2 (dashed line). Fig. 1.1 shows the time evolution of the relative error in the number of ions (a), the perturbed energy (b), the

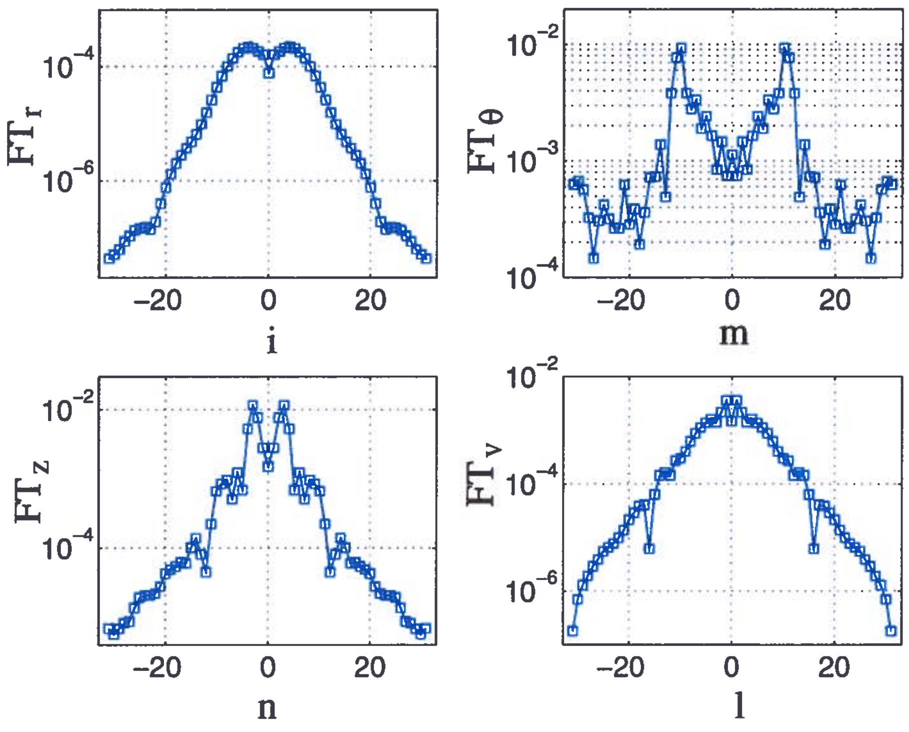

Figure 1.2: Spectrum in r (top left), θ (top right), z (bottom left) and $v_{||}$ (bottom right) at time $t = 720$.

entropy (c) and the L2-norm (d). Even if the new scheme gives a better energy and number of ions conservation with respect to the scheme based on the Newton-Raphson algorithm (cf. frames (a) and (b) in Fig. 1.1) open problems still remain.

A quantity which is very useful in order to understand if the chosen resolution is sufficient is the Fourier spectrum of the perturbed distribution function, $\delta f = f - f_M$. In Fig. 1.2, the Fourier transforms at time $t = 720$ of δf with respect to r, θ, z and $v_{||}$ are shown (these results refer to the Bulirsch-Stoer method). This figure shows that small scales come into play very early in the simulation (at time $t \simeq 700$, which corresponds to the end of the linear phase[3], see Fig 3.5, and even before using the predictor-corrector method), and they first appear in θ direction. Indeed, the θ spectrum at

[3]This explains why Fig. 2.2 in Ref. [3] is difficult to understand: a study of the convergence in dt is performed until $t \simeq 4000$ while spatial resolution starts to give numerical problems at time $t \simeq 700$.

time $t = 720$ is flat near the boundaries $m = \pm 32$ and the mode amplitudes are of the order of 10^{-4}.

Moreover, the distribution function becomes negative at time $t \simeq 1100$ (note that the same occurs also using the Newton-Raphson algorithm).

In summary, even if the Bulirsch-Stoer method gives better results with respect to the predictor-corrector method (at given resolution and time-step), there are two main problems which must be solved in order to properly study the non-linear phase of ITG instability: (i) small scales which appear in the distribution function must be correctly resolved and (ii) the positivity of f must be preserved. In the next chapter, we will see how these two problems are connected.

Chapter 2

Solving the negative values problem

This chapter is organised as follows: in the first part, we will describe two different schemes which preserve the positivity of the distribution function: the Positive and Flux Conservative (PFC) method (in Section 2.1, see Ref. [5]) and a method based on the direct interpolation of the logarithm of the distribution function (in Section 2.2, idea by L. Villard). In the last section, we will present numerical results in 2D phase space.

2.1 Positive Flux Conservative scheme

We want to solve the following equation:

$$\partial_t f + \partial_x (u(x,t)\ f) = 0 \tag{2.1}$$

We introduce a finite set of mesh points $(x_{i+1/2})$ of the computational domain (x_{min}, x_{max}), we denote the grid size as $\Delta x = x_{i+1/2} - x_{i-1/2}$ and the cell as $C_i = [x_{i-1/2}, x_{i+1/2}]$. Assuming that the values of the distribution function are known at time $t^n = n\Delta t$, one can find the new values at time t^{n+1} by integrating the distribution function over each cell. Denoting by $X(s) = X(s; t, x)$ the characteristic curve at time s such that $X(t) = x$ and $dX/ds = u(X(s), s)$ gives

$$\int_{x_{i-1/2}}^{x_{i+1/2}} f(x, t^{n+1})\ dx = \int_{X(t^n; t^{n+1}, x_{i-1/2})}^{X(t^n; t^{n+1}, x_{i+1/2})} f(x, t^n)\ J(t^n, t^{n+1}, x)\ dx \tag{2.2}$$

where $J(t^n, t^{n+1}, x) = \partial_x X(t^n; t^{n+1}, x)$ is the Jacobian. If the condition $J = 1$ is satisfied[1], one can define the flux as

$$\Phi_{i+1/2}(t^n) = \int_{X(t^n;t^{n+1},x_{i+1/2})}^{x_{i+1/2}} f(x, t^n)\, dx \tag{2.4}$$

to obtain the **conservative scheme**

$$\int_{x_{i-1/2}}^{x_{i+1/2}} f(x, t^{n+1})\, dx = \Phi_{i-1/2}(t^n) + \int_{x_{i-1/2}}^{x_{i+1/2}} f(x, t^n)\, dx - \Phi_{i+1/2}(t^n) \tag{2.5}$$

Introducing the distribution function averaged over a cell

$$F_i(t^n) = \int_{x_{i-1/2}}^{x_{i+1/2}} f(x, t^n)\, dx \tag{2.6}$$

Eq. (2.5) reads

$$F_i(t^{n+1}) = F_i(t^n) + (\Phi_{i-1/2}(t^n) - \Phi_{i+1/2}(t^n))/\Delta x \tag{2.7}$$

In the following, the time variable t^n only acts as a parameter and will be dropped. We want now a **suitable approximation of f over a cell in order to calculate (approximatively) the fluxes** (2.4). One can take the following expression (see Ref. [5]):

$$\begin{aligned}\mathcal{F}(x) &= F_i + \frac{\epsilon_i^+}{6\Delta x^2}[2(x - x_i)(x - x_{i-3/2}) \\ &+ (x - x_{i-1/2})(x - x_{i+1/2})](F_{i+1} - F_i) \\ &- \frac{\epsilon_i^-}{6\Delta x^2}[2(x - x_i)(x - x_{i+3/2}) \\ &+ (x - x_{i-1/2})(x - x_{i+1/2})](F_i - F_{i-1})\end{aligned} \tag{2.8}$$

[1]In the case of the 4D code, this condition is satisfied, for example, for the z and $v_{||}$ advections (cf. Eqs. (1.4)-(1.5)). For the r-θ advection, in order to have $J = 1$ one must consider (instead of Eq. (2.1)) a 2D equation

$$\partial_t f + \nabla \cdot (v_{GC}\ f) = 0 \tag{2.3}$$

and one must generalise Eqs. (2.7)-(2.8)-(2.11)-(2.13), which define the PFC method, to two dimensions.

where ϵ_i^+ and ϵ_i^- are the **slope correctors, introduced to guarantee the positivity of** f and defined as:

$$\epsilon_i^+ = \begin{cases} \min(1; 2F_i/(F_{i+1} - F_i)) & \text{if } F_{i+1} > F_i \\ \min(1; -2(f_\infty - F_i)/(F_{i+1} - F_i)) & \text{if } F_{i+1} < F_i \end{cases}$$

and

$$\epsilon_i^- = \begin{cases} \min(1; 2(f_\infty - F_i)/(F_i - F_{i-1})) & \text{if } F_i > F_{i-1} \\ \min(1; -2F_i/(F_i - F_{i-1})) & \text{if } F_i < F_{i-1} \end{cases}$$

with $f_\infty = \max(f)$. Eq. (2.8) is chosen such that:

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \mathcal{F}_i \, dx = F_i \, \Delta x \tag{2.9}$$

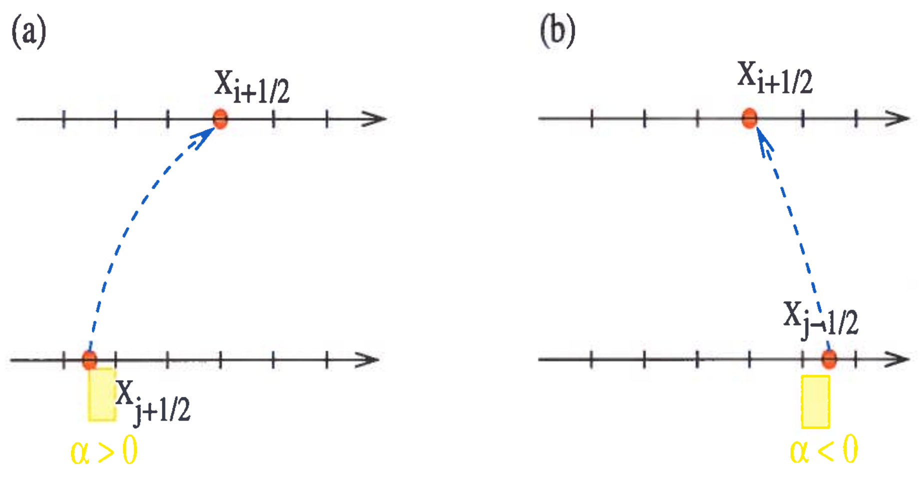

Figure 2.1: (a) Shift in x direction for the case $u > 0$ (a) and $u < 0$ (b).

Now, consider the case of positive propagating velocity $u > 0$. Find the cell C_j such that $X(t^n; t^{n+1}, x_i) \in C_j = [x_{j-1/2}, x_{j+1/2}]$ and define (see Fig. 2.1a)

$$\alpha = x_{j+1/2} - X(t^n; t^{n+1}, x_{i+1/2}), \qquad \text{with} : 0 \leq \alpha \leq \Delta \text{x} \tag{2.10}$$

Thus, Eq. (2.4) and Eq. (2.8) give

$$\begin{aligned} \Phi_{i+1/2} &= \Delta x \sum_{k=j+1}^{i} F_k + \alpha[F_j + \frac{\epsilon_i^+}{6}\left(1 - \frac{\alpha}{\Delta x}\right)\left(2 - \frac{\alpha}{\Delta x}\right)(F_{j+1} - F_j) \\ &+ \frac{\epsilon_i^-}{6}\left(1 - \frac{\alpha^2}{\Delta x}\right)(F_j - F_{j-1})] \end{aligned} \tag{2.11}$$

In the case of negative propagating velocity $u < 0$, one obtains (see Fig. 2.1b)

$$\alpha = x_{j-1/2} - X(t^n; t^{n+1}, x_{i+1/2}), \qquad \text{with}: \; -\Delta \mathrm{x} \leq \alpha \leq 0 \tag{2.12}$$

and

$$\begin{aligned}\Phi_{i+1/2} &= -\Delta x \sum_{k=i+1}^{j-1} F_k + \alpha[f_j - \frac{\epsilon_i^+}{6}\left(1 - \frac{\alpha^2}{\Delta x}\right)(F_{j+1} - F_j) \\ &- \frac{\epsilon_i^-}{6}\left(2 + \frac{\alpha}{\Delta x}\right)\left(1 + \frac{\alpha}{\Delta x}\right)(F_j - F_{j-1})]\end{aligned} \tag{2.13}$$

In summary, the Positive Flux Conservative (PFC) scheme is based on the following three main ingredients: **(i)** a conservative scheme (Eq. (2.7)), **(ii)** an approximation of the distribution function over a cell (Eq. (2.8)) and **(iii)** slope correctors to guarantee the positivity of f.

2.2 Logarithmic interpolation

In order to solve the problem of negative values in f one can use another scheme based on a simple idea (by L. Villard).

Instead of calculating the cubic spline coefficients of f, for example in 1D:

$$f(x) = \sum_\alpha c_\alpha \Lambda_\alpha(x) \tag{2.14}$$

one calculates first the cubic spline coefficients of the logarithm of f:

$$\ln f(x) = \sum_\alpha \tilde{c}_\alpha \Lambda_\alpha(x) \tag{2.15}$$

and uses $\tilde{c}_\alpha$ (instead of c_α) to interpolate the distribution function at the particle position x^* and calculate $\ln f(x^*)$. Inverting the logarithm now gives the desired value of $f(x^*)$ which, by construction, is positive.

2.3 Numerical results in 2D phase space

In order to test the methods presented in the previous sections, we use a semi-Lagrangian code in 2D phase space, which solves the Vlasov equation

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} + E(t,x)\frac{\partial f}{\partial v} = 0 \tag{2.16}$$

where $f(x,v,t)$ is the electron distribution function and $E(x,t)$ is the electric field. Eq. (2.16) is coupled to the Poisson equation[2]

$$\frac{\partial E(x,t)}{\partial x} = \int_{-\infty}^{\infty} f(x,v,t)\, dv - 1 \tag{2.17}$$

Ions are taken to be motionless and provide a uniform, neutralizing background. Furthermore, periodic boundary conditions are assumed in x, L being the box length. Oscillations are excited by initializing a single Fourier mode $k = 2\pi/L$:

$$f(x,v,0) = f_M(v)(1 + \alpha\cos(kx)) \tag{2.18}$$

where $f_M(v) = (2\pi)^{-1/2}\exp(-v^2/2)$ is the equilibrium Maxwellian. The Vlasov equation (2.16) is solved using the time-splitting scheme described in Ref. [6], which is second-order in time.

We use this 2D code because it is much faster than the 4D code and because it describes a physical problem which is well known. From the results of this simplified code we can learn a lot on how the semi-Lagrangian scheme works. We report results for a case with parameters $\alpha = 0.05$, $k = 0.4$ and $dt = 0.1$, which are the same as those used in Ref. [7] except for the cutoff velocity, which is set to $v_{max} = 5$.

Four different schemes will be compared:

- CS: scheme which uses cubic spline interpolation
- CSLI: scheme which uses cubic spline and logarithmic interpolation
- FC: flux conservative scheme
- PFC: positive flux conservative scheme (which includes the slope correctors to guarantee the positivity of f)

We consider a case with mesh grid $(N_x = 128, N_v = 512)$. First we note that negative values appear with both the FC (at time $t \simeq 10$) and the CS method (at time $t \simeq 60$), while the positivity of f is preserved with both PFC and CSLI.

[2]In normalized units: time is normalized to the inverse electron plasma frequency ω_{pe}^{-1}, space is normalized to the Debye length λ_D, and velocity to the electron thermal speed $V_{T_e} = \lambda_D\omega_{pe}$.

We now study the time evolution of the Fourier spectrum of the perturbed distribution function, $\delta f(x, v, t) = f(x, v, t) - f_M(v)$. This quantity, as said in Section 1.4, is very useful because it tells us if the chosen resolution is sufficient. In Fig. 2.2, the Fourier transform of δf with respect to v, $FT_v(\delta f(L/2, v, t = 60)$, is shown: the frame on the left refers to the case obtained using CSLI, while the frame on the right corresponds to the case PFC.

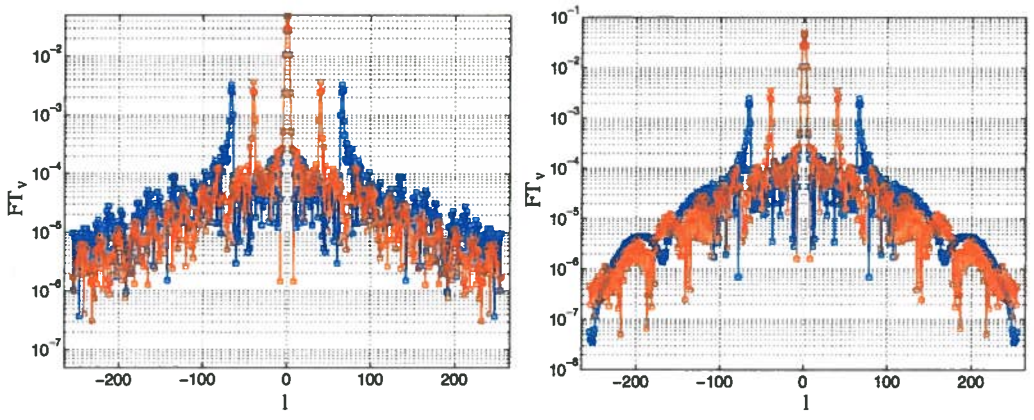

Figure 2.2: Spectrum in v for the case CSLI (left frame) and PFC (right frame) for times $t = 60$ (red) and $t = 100$ (blue).

Note that:

- In the case CSLI (and the same happens with CS), problems appear very early in the simulation and first in the v direction
- at $t \simeq 60$, the Fourier transform in v direction of δf shows that small scales become important and reach an amplitude of the order of 10^{-5} – 10^{-6} both in CS and in CSLI
- The results obtained with PFC show that small scales in v direction are one order of magnitude smaller that those with CSLI, i.e. of the order of 10^{-7} until the end of the simulation $t = 100$. Indeed, small details of the distribution function are eliminated by the algorithm itself which, as seen in Section 2.1, evolves the average of f over a cell.

Let us analyse now the time evolution of the constants of motion.
From Fig. 2.3, it can be easily seen that

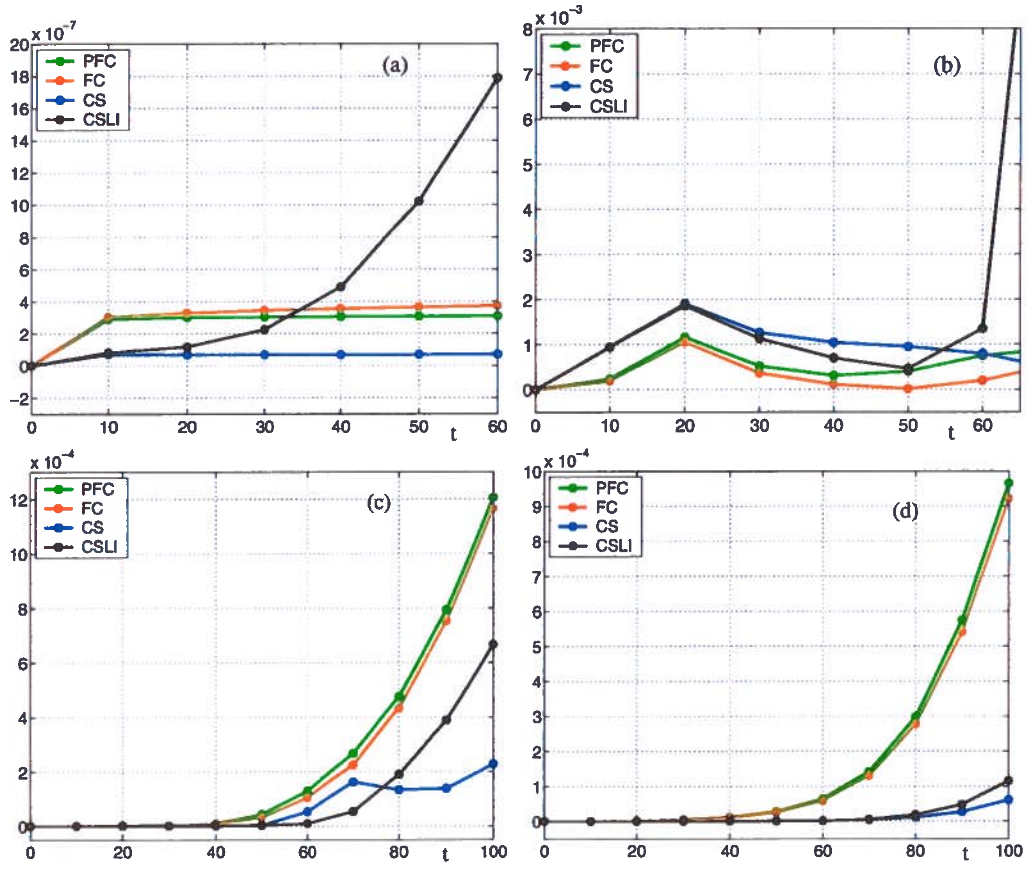

Figure 2.3: Time evolution of the relative error in the number of ions (a), the total energy (b), the entropy (c) and the L2-norm (d) for the different schemes.

- the PFC scheme gives very good results because the corresponding simulation is not affected by negative values (as in the case of FC or CS), the resolution is sufficient (cf. Fig. 2.2, right frame) and the constants of motion are conserved to a good accuracy. The force of the PFC method is to introduce **numerical diffusion** which smoothes the small scales and stabilizes the scheme. Indeed, the flux conservative schemes FC and PFC result to be more diffusive than CS and CSLI, in agreement with Ref. [5] (the relative error in the entropy and in the L2-norm are always larger).
- Even if the CS scheme seems to conserve energy and particles number

to a good accuracy, we already know that it is affected by two problems: (i) appearance of negative values in f and (ii) insufficient resolution[3].

- the CSLI scheme stops to give energy and number of ions conservation at time $t \simeq 60$ and this is the same time at which small scales become of the order of 10^{-6} (cf. Fig. 2.2, left frame).

But, what happens at time $t \simeq 60$ which destroys the conservations in the CSLI scheme? As shown in Fig. 2.6, vortices appear in phase space near the resonance regions $v \simeq \pm v_{res} = \pm 3.21$ due to the interaction of the wave with the particles trapped inside it. In Fig. 2.4, a sequence of frames which describes the time evolution of the resonant region $v = v_{res}$ in phase space is shown. At the initial time, $t = 0$, the particles are moving on free-streaming orbits. At later times, 'ripples' appear and they develope 'large crests' which eventually break down, leading to the closed orbits of the trapped particles. At time $t \simeq 60$, the 'topology' of the orbits changes from free-streaming to closed orbits. This kind of evolution should be forbidden by the Hamiltonian nature of the Vlasov equation, as discussed in Section 1.1. The CSLI scheme seems to be very sensitive to the time at which the smallest scales come into play and the orbit 'topology' changes. As the phase-space resolution increases, the conservation of the constants of motion obtained with the CSLI scheme is more accurate, as shown in Fig. 2.5, meaning that the Hamiltonian nature of the system is destroyed later in time.

On the other hand, the numerical diffusion introduced by the PFC method regularizes the filamentary structures inside the vortices, as seen from the comparisons of the two frames in Fig. 2.6 obtained with CSLI (left) and PFC (right). In Fig. 2.7, the Fourier transforms in x and v of δf are shown at $t = 60$ for the two schemes. From this figure, one can see how PFC cuts the small scales in comparison with CSLI, both in x and v directions.

2.4 Discussion

The numerical study of the 2D problem has given us useful informations on how different schemes work and on their advantages and disadvantages. Now we want to discuss the implementation of these schemes in the 4D code.

[3]From this we learn that the conservation of the constants of motion is **only sufficient** but not necessary for having a simulation of good quality.

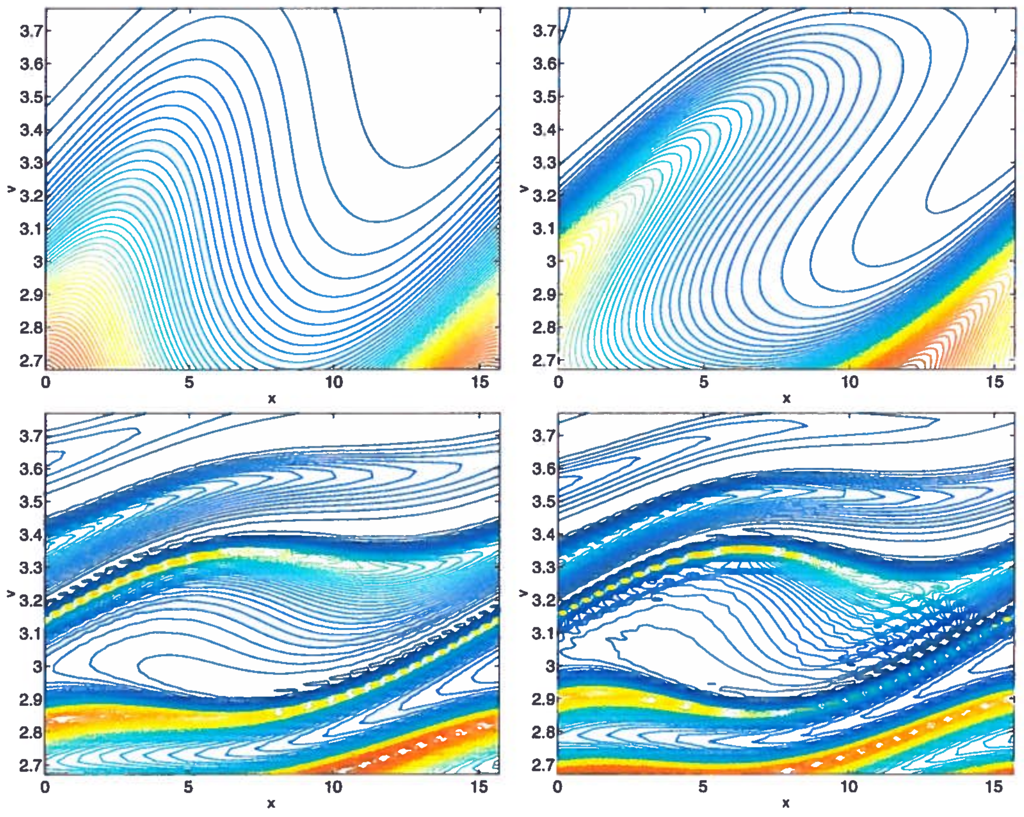

Figure 2.4: Contour plots of f in the resonant region at time $t = 0$ (left top), $t = 15$ (right top), $t = 60$ (left bottom) and $t = 70$ (right bottom).

In the numerical studies of collisionless systems, one should avoid as much as possible to introduce numerical diffusion. As an example, one of the open problems in the study of ITG modes is to understand the mechanisms of saturation of the zonal flow and to establish if it is due to collisionless or collisional phenomena. In order to address this kind of problem numerically one should minimize the introduction of spurious diffusion.

As we have seen in the previous section, the PFC scheme allows us to obtain the conservation of the constants of motion to a good accuracy but to have this conservation we have to pay the price of introducing numerical diffusion to cut the smallest scales.

But there is another way to cut the smallest scales (at least in r and θ directions), which is more 'physical'. It consists in considering gyro-averaged

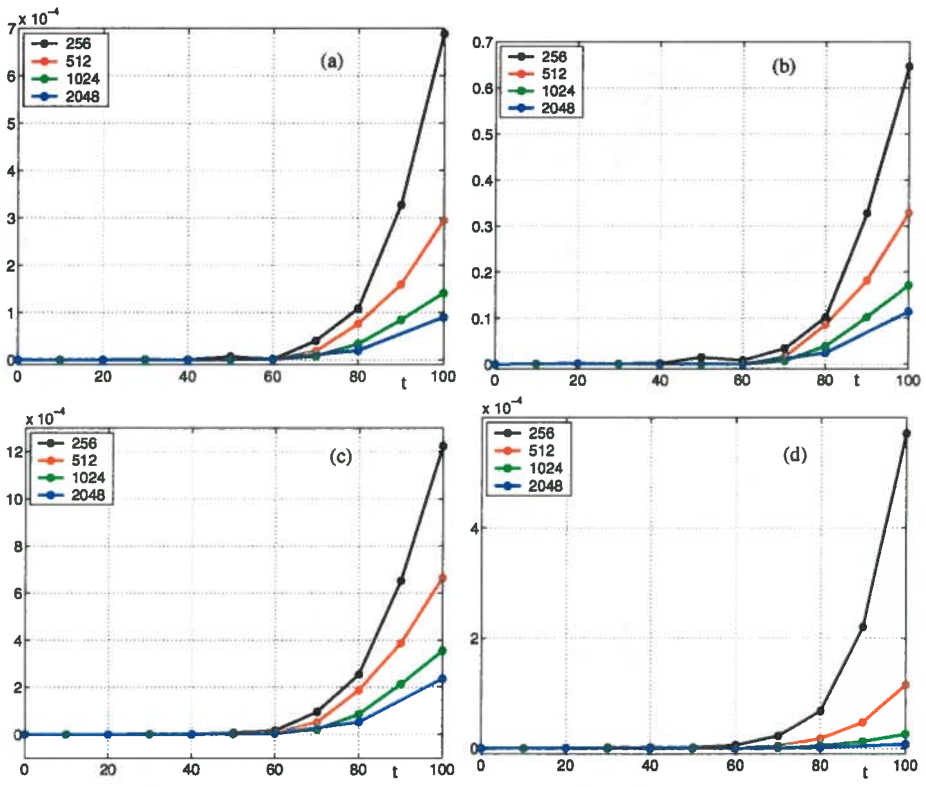

Figure 2.5: Time evolution of the relative error in the ions number (a), the total energy (b), the entropy (c) and the L2-norm (d) for different values of N_v and using the CSLI scheme.

equations (instead of the drift-kinetic approximation). In this way, a 'physical cutoff', given by the Larmor radius, appears in the system and we can hope to obtain the same regularizing effects that we have seen using the PFC method but without introducing spurious numerical diffusion.

In the present version of the 4D code which neglects finite Larmor effects, we have chosen to implement the logarithmic interpolation to solve the negative values problem with a minimal introduction of numerical diffusion. As seen in the previous section, the CSLI scheme is no more conservative when the smallest scales come into play. This can be considered as an advantage of the scheme, which responds quickly to the appearance of unresolved scales with a numerical instability (while, in contrast, the CS method continues to give a good conservation of the constants of motion (even if the spatial

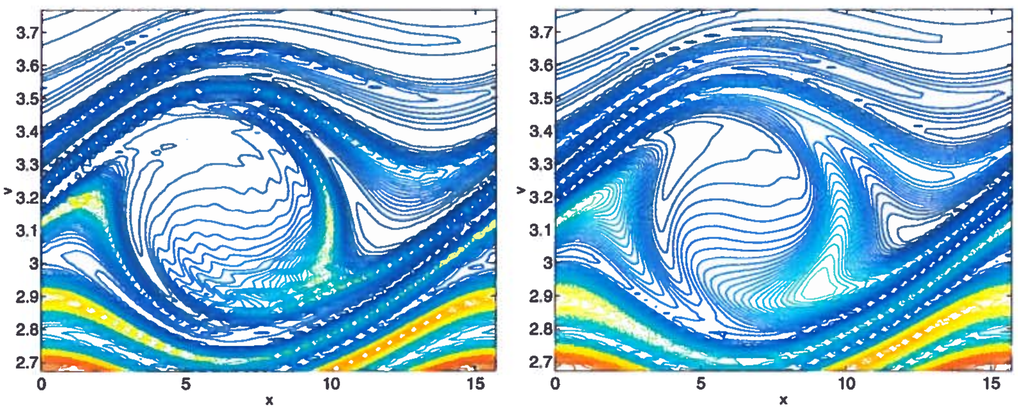

Figure 2.6: Vortex in phase space near $v = v_{res}$ at time $t = 100$ obtained with CSLI (left) and PFC method (right).

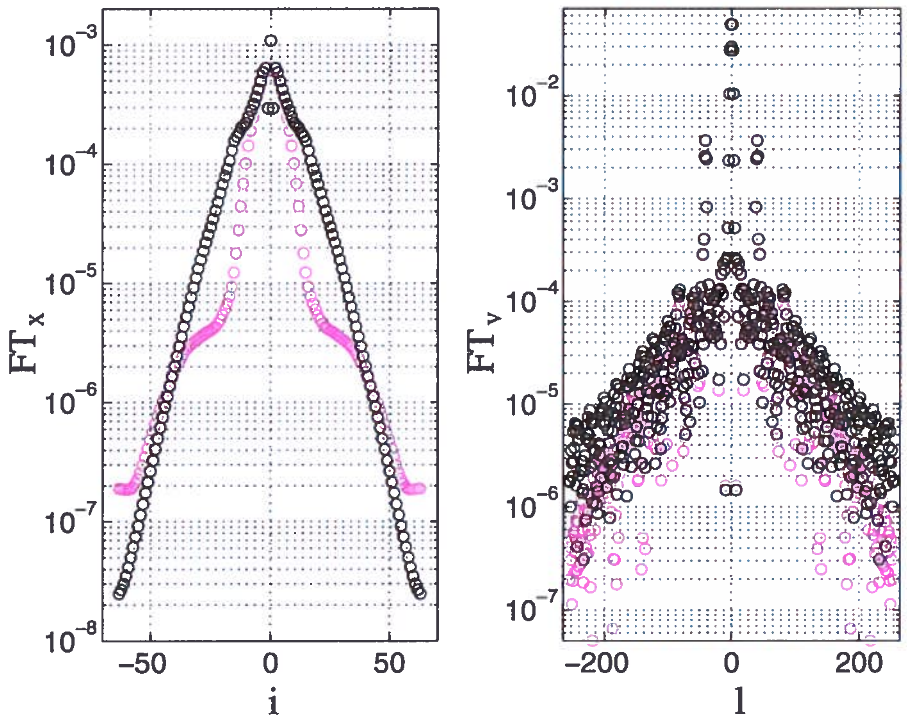

Figure 2.7: Spectra of δf in x (left) and v (right) directions at time $t = 60$ obtained with PFC (magenta) and CSLI (black).

resolution is not sufficient) developing negative values in f).

We have not chosen for the moment the PFC method because we plan to introduce in the next future gyro-averaged equations which, as discussed before, should be able to regularize the small structures in a more physical way.

Chapter 3

Increasing resolution

In this Chapter, we will present the results obtained with the semi-Lagrangian 4D code described in Chapter 1. If one applies the logarithmic interpolation method to solve the problem of negative values in f and if one uses a phase-space resolution equal to 64 points in each direction (cf. results presented in Chapter 1 and in Ref. [3]), one obtains a very poor conservation of the constants of motion (see Section 3.1). Why?

As discussed in the previous chapter, the problem is that the resolution is not sufficient. This enforces us to introduce a non-equidistant mesh (NEM) in the non-periodic directions, r and $v_{\|}$ (see Section 3.2), which enables us to resolve properly the regions where the physics take place and, at the same time, to gain in CPU time.

In Section 3.3, we will discuss the results obtained with the present version of the code, which includes NEM in r and $v_{\|}$ directions and the logarithmic interpolation.

3.1 Numerical results with logarithmic interpolation and N = 64 in each direction

If we consider the same parameters as RUN1 (cf. Section 1.4) and if the logarithmic interpolation is included in the semi-Lagrangian code to guarantee the positivity of f, the following results are obtained.

In Fig. 3.1, the Fourier transforms at time $t = 720$ of δf with respect to r, θ, z and $v_{\|}$ are shown. This figure shows that the θ spectrum at time $t = 720$ is flat at the boundaries $m = \pm 32$ and the mode amplitudes are of

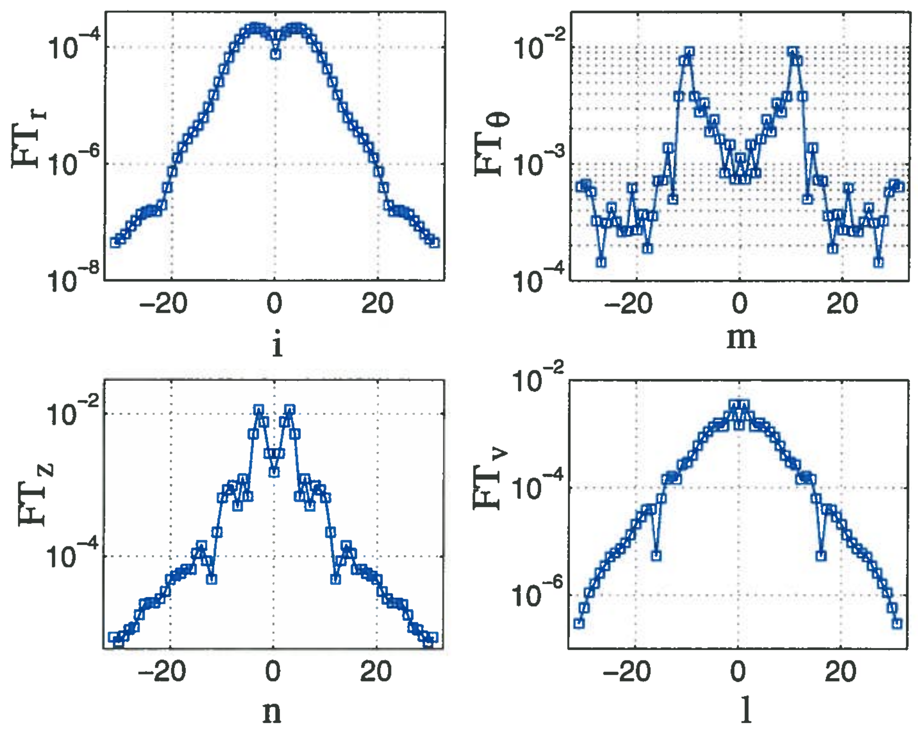

Figure 3.1: Spectrum in r (top left), θ (top right), z (bottom left) and $v_{||}$ (bottom right) at time $t = 720$.

the order of 10^{-4}, meaning that the smallest scales are not well resolved (the same occurred for RUN1, cf. Section 1.4: Fig. 1.2 is almost equal to Fig. 3.1).

Fig. 3.2 shows the comparison between the results of RUN1 (dotted line) and those obtained using the logarithmic interpolation (RUN1log, solid line). This figure shows the time evolution of the relative error in the number of ions (a), the perturbed energy (b), the entropy (c) and the L2-norm (d) until time $t = 720$, which is the time at which the smallest scale come into play in θ direction. Note that, as occurred also in the 2D problem discussed in Chapter 2, with the CSLI scheme the particles number and the total energy start to be not conserved when the smallest scales appear, i.e. at time $t \simeq 700$. Before this time, the conservation of the constants of motion obtained with CS and CSLI is comparable.

Thus, we need to increase resolution in order to study further in time the evolution of the system.

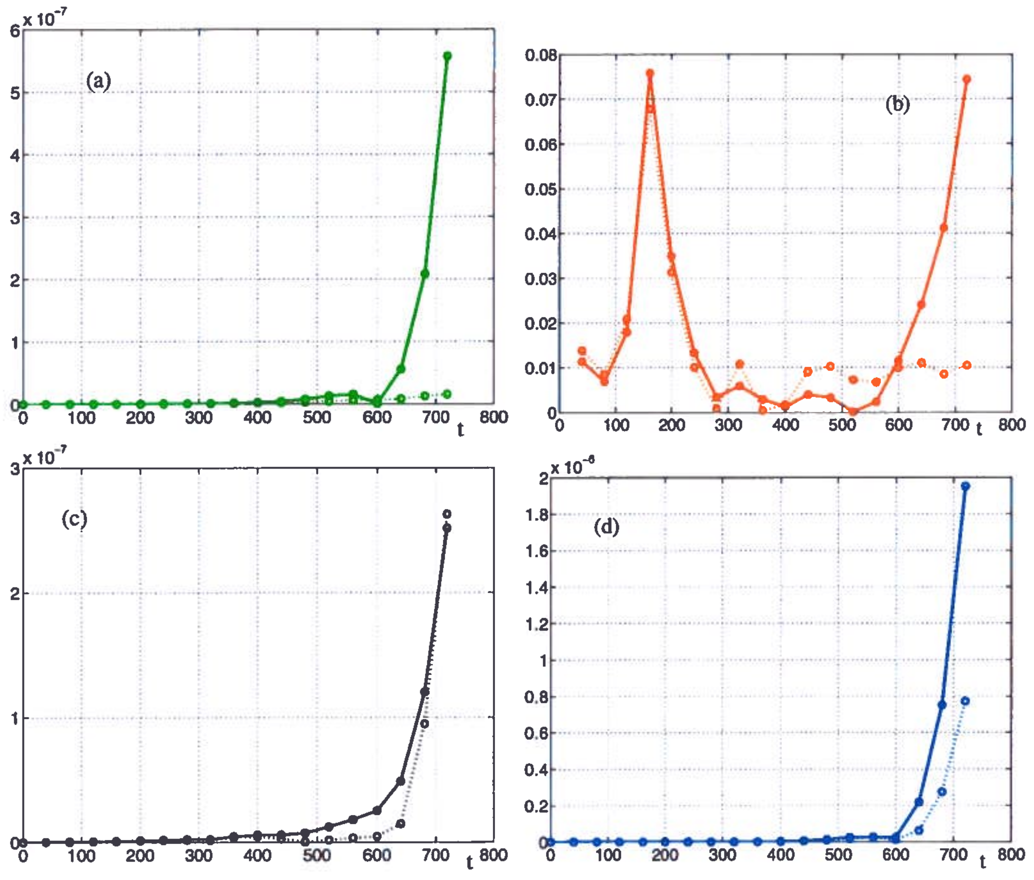

Figure 3.2: Time evolution of the relative error in the number of ions (a), the total energy (b), the entropy (c) and the L2-norm (d) for RUN1 (parameters given in Section 1.4) (dotted line) and using logarithmic interpolation (RUN1log, solid line).

3.2 Non-equidistant mesh

In order to increase resolution mainly in the region where the physics take place, a non-equidistant mesh (NEM) has been introduced in the non-periodic directions r and $v_{\|}$. The grid mesh is finer in r where the temperature gradient is steeper; in $v_{\|}$ direction, the grid mesh is finer where the distribution function is larger, more precisely in the regions near the phase velocity of the most unstable n mode, $v_{\|} \simeq \pm 1$ (see Fig. 3.3).

The cubic spline interpolation has been modified to consider a variable

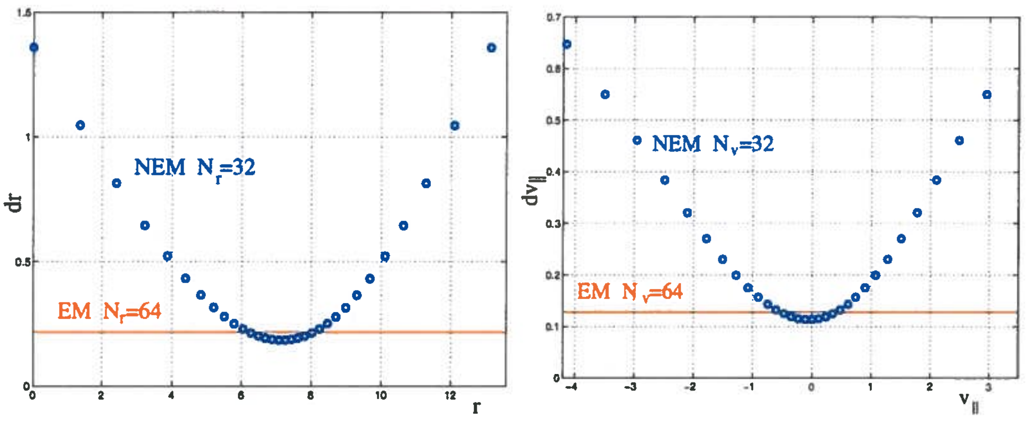

Figure 3.3: Distribution of mesh points in r (left) and $v_{||}$ (right) directions for $N = 32$. Also shown (in red) is the value of the equidistant mesh size corresponding to double N.

mesh size, as explained in details in Appendix B. The introduction of NEM allows us to obtain the same physical results with half the number of points in r and $v_{||}$ directions and to gain in CPU time.

As an example, RUN2 with $Nr = N_{v_{||}} = 64$, $N_\theta = 128$ and $N_z = 32$ with an equidistant-mesh gives the same results of RUN3 with $Nr = N_{v_{||}} = N_z = 32$ and $N_\theta = 128$ with NEM; the difference in CPU time is (22 h/16 processors) against (9 h/16 processors) with NEM.

3.3 Numerical results with increased resolution

Since small scales appear first in the θ direction, the number of points have been increased as shown in the following table (all the other parameters are the same as those of RUN1, cf. Section 1.4):

RUN	color	N_r	N_θ	N_z	$N_{v_\parallel}$	mesh
RUN1log	green	64	64	64	64	EM
RUN3	red	32	128	32	32	NEM
RUN4	blue	32	256	32	32	NEM
RUN5	black	32	512	32	32	NEM
RUN6	magenta	64	512	32	32	NEM
RUN7	cyan	128	256	32	32	NEM

In this table, NEM means that a Non Equidistant Mesh has been used in r and $v_\parallel$ directions, every simulation is associated to a name (RUN) and a color, which corresponds to the color used in Fig. 3.4. All the runs in the table have been performed using logarithmic interpolation. As discussed in the previous chapter, increasing resolution allows us to study the evolution of the system further in time and the logarithmic interpolation scheme guarantees that the introduction of spurious dissipation is minimised.

In Fig. 3.4, the time evolution of the constants of the motion is shown for the different RUNs. The resolution of RUN6 is sufficient to study the saturation phase of ITG modes with a relative error in the perturbed total energy less than 6%.

The development of small scales in the nonlinear phase of the instability can be seen in Fig. 3.5, where the contour plots of $f(r, \theta, L_z/2, v_\parallel = 0)$ are shown for different times. As time evolves, filaments develop in the distribution function which become finer and finer, until gradients become so steep that the scheme used is no more able to follow their evolution.

From Fig. 3.5, it is evident how important will be to introduce gyro-averaged equations, which can regularize structures of the size of the Larmor radius in a physical way, without introducing spurious dissipative effects.

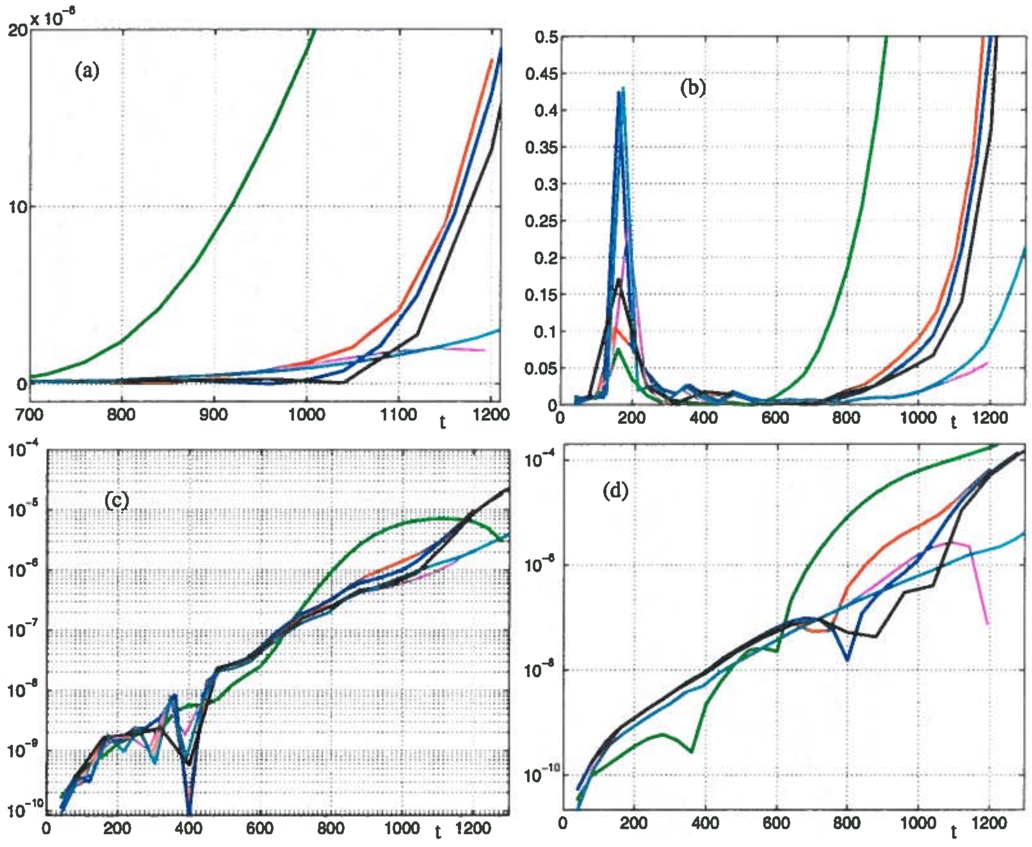

Figure 3.4: Time evolution of the relative error in the number of ions (a), the total energy (b), the entropy (c) and the L2-norm (d) (the last two plots are semilogarithmic plots); colors correspond to resolutions listed in the table.

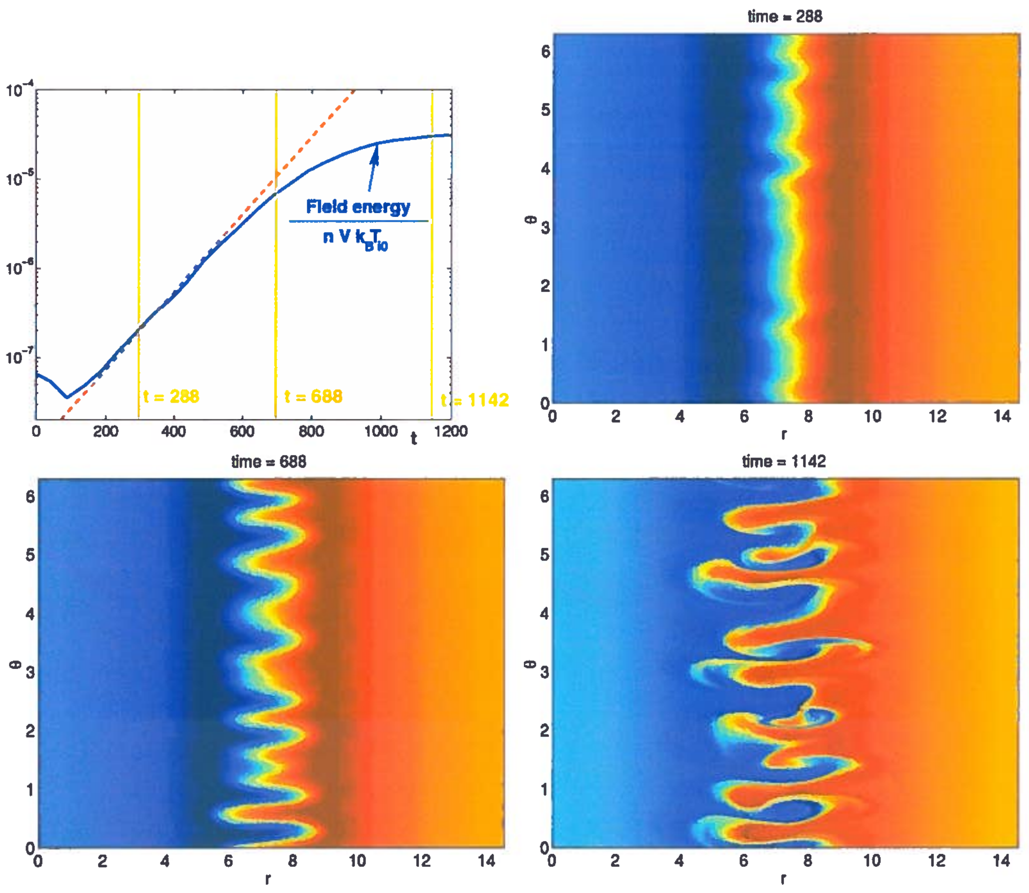

Figure 3.5: Contour plots of $f(r, \theta, L_z/2, v_\parallel = 0)$ at time $t = 288$, $t = 688$ and $t = 1142$ for RUN6.

Conclusions and future work

In conclusion, let us summarise the main points of this report.

We have analysed the numerical results obtained with a semi-Lagrangian scheme based on the Bulirsch-Stoer algorithm. We have seen that a resolution equal to $N = 64$ points in each direction is not sufficient, because small scales develop and negative values appear in the distribution function.

We have studied different numerical schemes to address the problem of preserving the positivity of f: the Positive Flux Conservative (PFC) method and a method based on the cubic spline interpolation of the logarithm of f (CSLI). We have seen that PFC is more dissipative than CSLI but that it is more conservative. We have decided to implement CSLI in the 4D code to avoid as much as possible to introduce spurious numerical diffusion. The PFC method will be reconsidered when the physical model will be generalised to gyro-averaged equations. In this case, scales in r and θ directions smaller that the Larmor radius are 'physically' eliminated. However, the PFC method may be important to cut small scales in z and $v_{\|}$ directions.

We have presented numerical results obtained using logarithmic interpolation and increased resolution. We are now able to study the evolution of the ITG modes until the saturation occurs with a good control of the constants of the motion.

For the future, we need to

- conclude the study of convergence of the code with resolution, in particular how the evolution of the system is modified if the number of points in z and $v_{\|}$ is changed
- study the convergence of the code with the time step at fixed (and sufficient) resolution
- include finite Larmor radius effects

Appendix A

Splitting scheme

If we denote by $\hat{Q}$ the solution of the quasi-neutrality equation and with $z/2$, $v_{\|}/2$ and $\widehat{r\theta}$ the shifts, respectively, in z, $v_{\|}$ (over $dt/2$) and r-θ directions (over dt), the following sequence of shifting of the distribution function is implemented in the code when the predictor-corrector scheme is used (see Section 1.2:

$$\left(\frac{v_{\|}}{2}\,\frac{z}{2}\,\hat{Q}\,\widehat{r\theta}\,\hat{Q}\,\frac{z}{2}\,\hat{Q}\,\frac{v_{\|}}{2}\right) \tag{A.1}$$

or more explicitly:

$$\begin{aligned}
f^{*}(r,\theta,z,v_{\|}) &= f^{n}(r,\theta,z,v_{\|}-E_z dt/2) && \text{(A.2)}\\
f^{**}(r,\theta,z,v_{\|}) &= f^{*}(r,\theta,z-v_{\|}dt/2,v_{\|}) && \text{(A.3)}\\
f^{***}(r,\theta,z,v_{\|}) &= f^{**}(\bar{r},\bar{\theta},z,v_{\|}) && \text{(A.4)}\\
f^{****}(r,\theta,z,v_{\|}) &= f^{***}(r,\theta,z-v_{\|}dt/2,v_{\|}) && \text{(A.5)}\\
f^{n+1}(r,\theta,z,v_{\|}) &= f^{****}(r,\theta,z,v_{\|}-\tilde{E}_z dt/2) && \text{(A.6)}
\end{aligned}$$

where the electric field component E_z is calculated at time t, that is $E_z = E_z(t) \equiv E_z(r(t),\theta(t),z(t),t)$ while $\tilde{E}_z$ in Eq. (A.6), is calculated after the second shift in the z direction, that is $\tilde{E}_z = E_z(t+dt) \equiv E_z(r(t+dt),\theta(t+dt),z(t+dt),t+dt)$. Moreover, $\bar{r}$ and $\bar{\theta}$ are obtained by using the Newton-Raphson algorithm together with the predictor-corrector scheme, as described in Chapter 1.

By substituting successively the former equations in the last one, we obtain:

$$\begin{aligned} f^{n+1}(r,\theta,z,v_{\|}) &= f^{****}(r,\theta,z,v_{\|}-\tilde{E}_z dt/2) \\ &= f^{***}(r,\theta,z-v_{\|}dt/2,v_{\|}-\tilde{E}_z dt/2) \\ &= f^{**}(\bar{r},\bar{\theta},z-v_{\|}dt/2,v_{\|}-\tilde{E}_z dt/2) \\ &= f^{*}(\bar{r},\bar{\theta},z-v_{\|}dt,v_{\|}-\tilde{E}_z dt/2) \\ &= f^{n}(\bar{r},\bar{\theta},z-(v_{\|}-E_z dt/2)dt,v_{\|}-(E_z+\tilde{E}_z)dt/2) \end{aligned}$$

We find that the previous equation is equivalent to the following equations of the characteristics:

$$r(t) = \bar{r}(t+dt) \tag{A.7}$$
$$\theta(t) = \bar{\theta}(t+dt) \tag{A.8}$$
$$z(t) = z(t+dt) - dt\ (v_{\|}(t+dt) - E_z dt/2) \tag{A.9}$$
$$v_{\|}(t) = v_{\|}(t+dt) - (E_z+\tilde{E}_z)dt/2 \tag{A.10}$$

We have already discussed in Section 1.2 the accuracy of the predictor-corrector scheme for Eqs. (A.7)-(A.8). Let us see now which is the order of accuracy of the last two equations. Eq. (A.10) gives the following scheme:

$$\begin{aligned} v_{\|}(t+dt) - v_{\|}(t) &= [E_z(t) + E_z(t+dt)]dt/2 \\ &= E_z(t+dt/2)dt + O(dt^3) \end{aligned} \tag{A.11}$$

Thus, Eq. (A.10) is second order accurate. Eq. (A.9) gives the following two equations:

$$z(t) = z(t+dt) - dt(v_{\|}(t+dt) - E_z(t)dt/2) \tag{A.12}$$

and

$$z(t-dt) = z(t) - dt(v_{\|}(t) - E_z(t-dt)dt/2) \tag{A.13}$$

the difference of which is:

$$\begin{aligned} z(t-dt) - 2z(t) + z(t+dt) &= dt(v_{\|}(t+dt) - v_{\|}(t)) \\ &\quad + dt^2(E_z(t-dt) - E_z(t))/2 \\ &= dt^2(E_z(t+dt) + E_z(t-dt))/2 \\ &\simeq dt^2 E_z(t) \end{aligned} \tag{A.14}$$

which is again second order accurate in time.

Appendix B

Cubic splines

B.1 Cubic spline interpolation in 1D

Let us write a 1D function $g(x)$ in terms of cubic spline basis $\Lambda_\nu(x)$ for all $x \in [x_0, x_N]$ as:

$$g(x) = \sum_{\nu=-1}^{N+1} c_\nu \Lambda_\nu(x)$$

where Λ_ν are piecewise cubic polynomials (shown in Fig. B.1), twice continuously differentiable, defined by:

$$\Lambda_\nu(x) = \frac{1}{6h^3} \begin{cases} (x - x_{\nu-2})^3 & \text{if } x_{\nu-2} \leq x \leq x_{\nu-1} \\ h^3 + 3h^2(x - x_{\nu-1}) + 3h(x - x_{\nu-1})^2 & \\ \quad -3(x - x_{\nu-1})^3 & \text{if } x_{\nu-1} \leq x \leq x_\nu \\ h^3 + 3h^2(x_{\nu+1} - x) + 3h(x_{\nu+1} - x)^2 & \\ \quad -3(x_{\nu+1} - x)^3 & \text{if } x_\nu \leq x \leq x_{\nu+1} \\ (x_{\nu+2} - x)^3 & \text{if } x_{\nu+1} \leq x \leq x_{\nu+2} \\ 0 & \text{otherwise} \end{cases}$$

where $h = (x_N - x_0)/N$. The previous definition holds if we consider an equidistant mesh. We will discuss how to generalize it for the case of non-equidistant mesh in section B.3.

The coefficients c_ν are computed solving the following system of equations:

$$g(x_i) = \sum_{\nu=-1}^{N+1} c_\nu \Lambda_\nu(x_i) \qquad i = 0, N \tag{B.1}$$

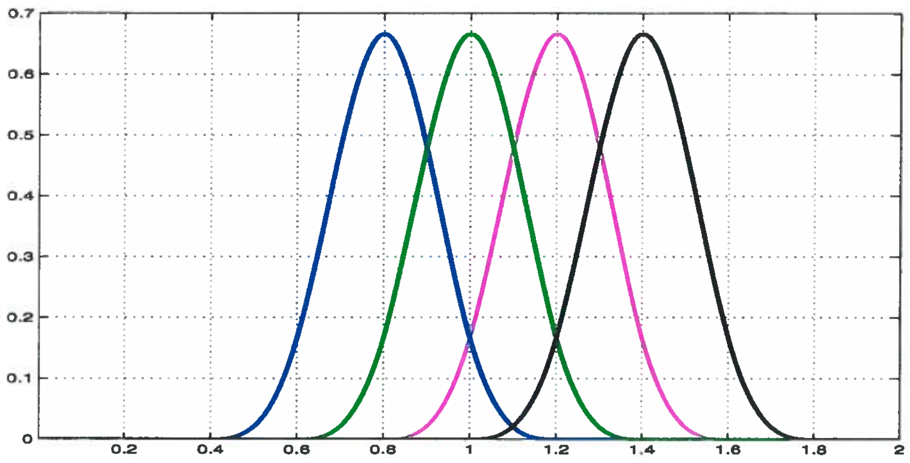

Figure B.1: Cubic spline basis in the case of equidistant-mesh

The system (B.1) can be written in the following matricial form:

$$[\Lambda]_{(N+1)\times(N+1)} \begin{pmatrix} c_{-1} \\ \vdots \\ c_{N+1} \end{pmatrix} = \begin{pmatrix} g(x_0) \\ \vdots \\ g(x_N) \end{pmatrix}$$

Thus, there are $(N+1)$ equations and $(N+3)$ unknowns. Imposing periodic or non-periodic boundary conditions gives rise to other two equations, as explained in the following subsections.

B.1.1 Non-periodic boundary conditions

In non-periodic case, we suppose that the first (case (a)) or the second derivative (case (b)) are known at the boundaries. Using the values of the cubic spline basis listed in the following table:

x	$x_{\nu-2}$	$x_{\nu-1}$	x_ν	$x_{\nu+1}$	$x_{\nu+2}$
$\Lambda_\nu(x)$	0	1/6	2/3	1/6	0
$\Lambda'_\nu(x)$	0	$1/(2h)$	0	$-1/(2h)$	0
$\Lambda''_\nu(x)$	0	$1/h^2$	$-2/h^2$	$1/h^2$	0

the $(N+3, N+3)$ matricial system to be solved becomes in case (a):

$$\begin{pmatrix} -1/(2h) & 0 & 1/(2h) & & \\ 1/6 & 2/3 & 1/6 & & 0 \\ & 1/6 & 2/3 & 1/6 & \\ & \ddots & \ddots & \ddots & \\ 0 & & 1/6 & 2/3 & 1/6 \\ & & -1/(2h) & 0 & 1/(2h) \end{pmatrix} \times \begin{pmatrix} c_{-1} \\ c_0 \\ \vdots \\ \vdots \\ c_N \\ c_{N+1} \end{pmatrix} = \begin{pmatrix} g'(x_0) \\ g(x_0) \\ \vdots \\ \vdots \\ g(x_N) \\ g'(x_N) \end{pmatrix} \tag{B.2}$$

For the case (b), only the first and the last rows change and are replaced by:

$$\begin{pmatrix} 1/h^2 & -2/h^2 & 1/h^2 & 0 & \cdots & 0 \end{pmatrix} \times \vec{c} = g''(x_0)$$
$$\begin{pmatrix} 0 & \cdots & 0 & 1/h^2 & -2/h^2 & 1/h^2 \end{pmatrix} \times \vec{c} = g''(x_N)$$

where $\vec{c} = \begin{pmatrix} c_{-1} & c_0 & \cdots & c_N & c_{N+1} \end{pmatrix}^t$.

If we permute the matrix to keep the boundary conditions in the last two rows, Eq. (B.2) reads:

$$\tilde{A}\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \text{where:} \begin{cases} u' = (c_0, \cdots, c_N)^t \\ v' = (c_{N+1}, c_{-1})^t \\ u = (g(x_0), \cdots, g(x_N))^t \\ v = (g'(x_N), g'(x_0))^t \end{cases}$$

and

$$\tilde{A} = \left(\begin{array}{c|cc} A & \multicolumn{2}{c}{\gamma} \\ \hline \lambda & \xi_1 & \xi_2 \\ & \xi_3 & \xi_4 \end{array}\right)$$

where:

$$\begin{cases} \cdot\, A \text{ is the } (N+1)\times(N+1) \text{ tridiagonal symmetric matrix : } \begin{pmatrix} 2/3 & 1/6 & & & \\ 1/6 & 2/3 & 1/6 & & \\ & \ddots & \ddots & \ddots & \\ & & 1/6 & 2/3 & 1/6 \\ & & & 1/6 & 2/3 \end{pmatrix} \\ \cdot\, \lambda \text{ is } 2\times(N+1) \text{ matrix : } \begin{pmatrix} 0 & \cdots & 0 & -1/(2h) & 0 \\ 0 & 1/(2h) & 0 & \cdots & 0 \end{pmatrix} \\ \cdot\, \gamma \text{ is } (N+1)\times 2 \text{ matrix : } \begin{pmatrix} 1/6 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 1/6 \end{pmatrix}^t \\ \cdot\, \delta = \begin{pmatrix} \xi_1 & \xi_2 \\ \xi_3 & \xi_4 \end{pmatrix} = \begin{pmatrix} 1/(2h) & 0 \\ 0 & -1/(2h) \end{pmatrix} \end{cases}$$

Besides, $\tilde{A}$ can be factorised in the LU form as:

$$\tilde{A} = \begin{pmatrix} A & 0 \\ \lambda & \bar{\delta} \end{pmatrix} \times \begin{pmatrix} I & A^{-1}\gamma \\ 0 & I \end{pmatrix} \qquad \text{with:} \qquad \bar{\delta} = \delta - \lambda A^{-1}\gamma$$

This LU factorisation is used to solve, by forward and backward substitutions, the (B.2) equivalent system:

$$\begin{pmatrix} A & 0 \\ \lambda & \bar{\delta} \end{pmatrix} \times \begin{pmatrix} I & A^{-1}\gamma \\ 0 & I \end{pmatrix} \times \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}$$

This means, at first, to solve

$$\begin{pmatrix} A & 0 \\ \lambda & \bar{\delta} \end{pmatrix} \times \begin{pmatrix} u'' \\ v'' \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}$$

and then

$$\begin{pmatrix} I & A^{-1}\gamma \\ 0 & I \end{pmatrix} \times \begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u'' \\ v'' \end{pmatrix}$$

Thus, the computation of the coefficients c_ν can be performed in the following steps:

1. Initialisation :
   (a) Factorise and store A
   (b) Compute and store $A^{-1}\gamma$
   (c) Compute and store the (2×2) matrix $\bar{\delta} = \delta - \lambda A^{-1}\gamma$

2. Time loop :
   (a) Compute and store $u'' = A^{-1}u$ using the previously computed factorisation of A
   (b) Compute $v - \lambda A^{-1}u$,
   (c) Solve the (2×2) system $\bar{\delta}v'' = v - \lambda A^{-1}u$ using the Cramer formulae for $\bar{\delta}^{-1} = \frac{1}{\det(\bar{\delta})} \begin{pmatrix} \bar{\xi}_4 & -\bar{\xi}_2 \\ -\bar{\xi}_3 & \bar{\xi}_1 \end{pmatrix}$
   (d) Compute u' using the previous storage of $A^{-1}\gamma$ by $u' = u'' - A^{-1}\gamma v'$, where v' is trivially equal to v''.

These steps involve at first the resolution of a tridiagonal symmetric system in the initialisation and then one at each time loop. These systems are solved by the LAPACK library subroutines which use a LDL^t factorisation for A.

B.1.2 Periodic boundary conditions

In the periodic case, we cannot use the condition $g(x_0) = g(x_N)$ because it is a linear combination of the equations (B.1). Thus, we use the continuity property of the cubic spline basis on the first and second derivatives:

$$\begin{cases} g'(x_0) = g'(x_N) \\ g''(x_0) = g''(x_N) \end{cases}$$

which gives respectively:

$$-\frac{1}{2h}c_{-1} + \frac{1}{2h}c_1 + \frac{1}{2h}c_{N-1} - \frac{1}{2h}c_{N+1} = 0$$

and

$$\frac{1}{h^2}c_{-1} - \frac{2}{h^2}c_0 + \frac{1}{h^2}c_1 - \frac{1}{h^2}c_{N-1} + \frac{2}{h^2}c_N - \frac{1}{h^2}c_{N+1} = 0$$

The equivalent $(N+3) \times (N+3)$ matricial system becomes:

$$\begin{pmatrix} 2/3 & 1/6 & 0 & & \cdots & \cdots & 0 & 0 & 1/6 \\ & 1/6 & 2/3 & 1/6 & & & & \vdots & 0 \\ & & & \ddots & \ddots & \ddots & 1/6 & 0 & \vdots \\ & & & & & 1/6 & 2/3 & 1/6 & 0 \\ 0 & \frac{1}{2h} & 0 & \cdots & 0 & \frac{1}{2h} & 0 & -\frac{1}{2h} & -\frac{1}{2h} \\ -\frac{2}{h^2} & \frac{1}{h^2} & 0 & \cdots & 0 & -\frac{1}{h^2} & \frac{2}{h^2} & -\frac{1}{h^2} & \frac{1}{h^2} \end{pmatrix} \times \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \\ c_N \\ c_{N+1} \\ c_{-1} \end{pmatrix} = \begin{pmatrix} g(x_0) \\ g(x_1) \\ \vdots \\ g(x_{N-1}) \\ g(x_N) \\ 0 \\ 0 \end{pmatrix}$$

which can be solved with the same method described in the previous subsection, the only differences being:

$$\begin{cases} \cdot\, v = (0,0)^t, \\ \cdot\, \lambda = \begin{pmatrix} 0 & 1/(2h) & 0 & \cdots & 0 & 1/(2h) & 0 \\ -2/h^2 & 1/h^2 & 0 & \cdots & 0 & -1/h^2 & 2/h^2 \end{pmatrix} \\ \cdot\, \delta = \begin{pmatrix} \xi_1 & \xi_2 \\ \xi_3 & \xi_4 \end{pmatrix} = \begin{pmatrix} -1/(2h) & -1/(2h) \\ -1/h^2 & 1/h^2 \end{pmatrix} \end{cases}$$

B.2 Cubic spline interpolation in 2D

Any 2D function $g(x,y)$ can be written in terms of cubic spline basis as follows:

$$g(x,y) = \sum_{\nu=-1}^{N_x+1} \sum_{\mu=-1}^{N_y+1} c(\nu,\mu)\Lambda_\nu(x)\Lambda_\mu(y)$$

for all $x \in [x_0, x_{N_x}]$ and $y \in [y_0, y_{N_y}]$, where N_x and N_y are the number of grid points. Thus, in order to compute the coefficients $c(\nu, \mu)$, we need to solve the following system:

$$g(x_i, y_j) = \sum_{\nu=-1}^{N_x+1} \sum_{\mu=-1}^{N_y+1} c(\nu, \mu)\Lambda_\nu(x_i)\Lambda_\mu(y_j) \qquad i = 0, N_x \text{ and } j = 0, N_y$$

This computation can be replaced by solving a succession of 1D systems, since $g(x_i, y_j)$ can be written as:

$$g(x_i, y_j) = \sum_{\nu=-1}^{N_x+1} \gamma(\nu, j)\Lambda_\nu(x_i) \quad \text{where} \quad \gamma(\nu, j) = \sum_{\mu=-1}^{N_y+1} c(\nu, \mu)\Lambda_\mu(y_j)$$

For these 1D interpolation problems, we can refer to the results of the previous sections.

B.3 Non-equidistant mesh

In the case of a non-equidistant mesh, let us start from the recurrence relation for constructing spline functions $B_{i,k}$:

$$B_{i,k}(x) = \frac{x - x_i}{x_{i+k-1} - x_i} B_{i,k-1}(x) + \frac{x_{i+k} - x}{x_{i+k} - x_{i+1}} B_{i+1,k-1}(x) \tag{B.3}$$

where $B_{i,1}(x) = 1$ if $x_i \le x \le x_{i+1}$ and 0 otherwise. If we call $h_i = x_i - x_{i-1}$, we obtain for the linear spline basis the following expression

$$B_{i,2}(x) = \begin{cases} (x - x_i)/h_{i+1} & \text{if } x_i \le x \le x_{i+1} \\ (x_{i+2} - x)/h_{i+2} & \text{if } x_{i+1} \le x \le x_{i+2} \\ 0 & \text{otherwise} \end{cases}$$

for the quadratic spline basis:

$$B_{i,3}(x) = \begin{cases} (x - x_i)^2/(h_{i+1}h_{12}) & \text{if } x_i \le x \le x_{i+1} \\ (x_{i+2} - x)(x - x_i)/(h_{i+2}h_{12}) + & \\ +(x_{i+3} - x)(x - x_{i+1})/(h_{i+2}h_{23}) & \text{if } x_{i+1} \le x \le x_{i+2} \\ (x_{i+3} - x)^2/(h_{i+3}h_{23}) & \text{if } x_{i+2} \le x \le x_{i+3} \\ 0 & \text{otherwise} \end{cases}$$

where $h_{12} = h_{i+1} + h_{i+2}$ and $h_{23} = h_{i+2} + h_{i+3}$. Finally, for the cubic spline basis:

$$B_{i,4}(x) = \begin{cases} (x-x_i)^3/(h_{i+1}h_{12}h_{123}) & \text{if } x_i \le x \le x_{i+1} \\ (x_{i+2}-x)(x-x_i)^2/(h_{i+2}h_{12}h_{123}) + & \\ +(x_{i+3}-x)(x-x_i)(x-x_{i+1})/(h_{i+2}h_{23}h_{123}) + & \\ +(x_{i+4}-x)(x-x_{i+1})^2/(h_{i+2}h_{23}h_{234}) & \text{if } x_{i+1} \le x \le x_{i+2} \\ (x_{i+3}-x)^2(x-x_i)/(h_{i+3}h_{23}h_{123}) + & \\ +(x_{i+4}-x)(x_{i+3}-x)(x-x_{i+1})/(h_{i+3}h_{23}h_{234}) + & \\ +(x_{i+4}-x)^2(x-x_{i+2})/(h_{i+3}h_{34}h_{234}) & \text{if } x_{i+2} \le x \le x_{i+3} \\ (x_{i+4}-x)^3/(h_{i+4}h_{34}h_{234}) & \text{if } x_{i+3} \le x \le x_{i+4} \\ 0 & \text{otherwise} \end{cases}$$

where $h_{34} = h_{i+3}+h_{i+4}$, $h_{123} = h_{i+1}+h_{i+2}+h_{i+3}$ and $h_{234} = h_{i+2}+h_{i+3}+h_{i+4}$. One can easily check that the previous expression corresponds to the one given in Section B.1 if $i \to \nu - 2$ and $h_i = h$.

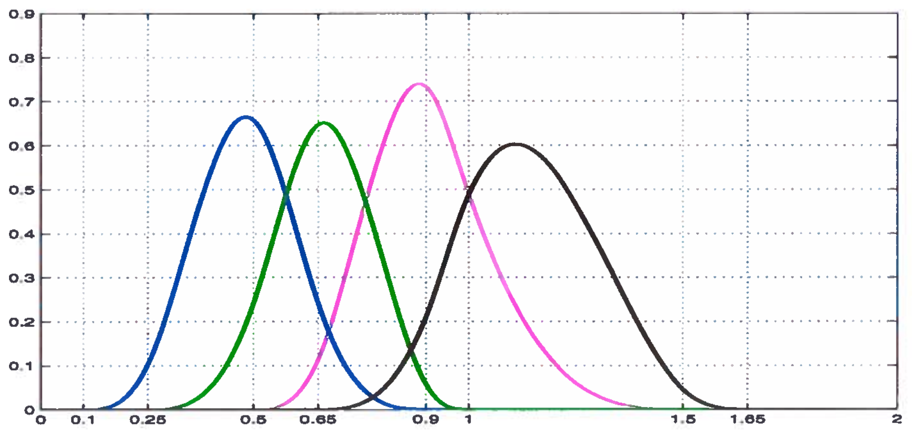

Figure B.2: Cubic spline basis in the case of non-equidistant mesh

B.3.1 Non-periodic boundary conditions

In the non-periodic case, we suppose, as before, that the first derivate is known at the boundaries. For reference, we rewrite the expression derived in the previous section as

$$B_{\nu,3}(x) = \begin{cases} (x-x_{\nu-2})^2/(h_{\nu-1}h_{12}) & \text{if } x_{\nu-2} \le x \le x_{\nu-1} \\ (x_\nu - x)(x-x_{\nu-2})/(h_\nu h_{12}) + & \\ +(x_{\nu+1}-x)(x-x_{\nu-1})/(h_\nu h_{23}) & \text{if } x_{\nu-1} \le x \le x_\nu \\ (x_{\nu+1}-x)^2/(h_{\nu+1}h_{23}) & \text{if } x_\nu \le x \le x_{\nu+1} \\ 0 & \text{otherwise} \end{cases}$$

where $h_{12} = h_{\nu-1} + h_\nu$ and $h_{23} = h_\nu + h_{\nu+1}$. For the cubic spline basis, we have:

$$B_{\nu,4}(x) = \begin{cases} (x-x_{\nu-2})^3/(h_{\nu-1}h_{12}h_{123}) & \text{if } x_{\nu-2} \le x \le x_{\nu-1} \\ (x_\nu - x)(x-x_{\nu-2})^2/(h_\nu h_{12}h_{123}) + & \\ +(x_{\nu+1}-x)(x-x_{\nu-2})(x-x_{\nu-1})/(h_\nu h_{23}h_{123}) + & \\ +(x_{\nu+2}-x)(x-x_{\nu-1})^2/(h_\nu h_{23}h_{234}) & \text{if } x_{\nu-1} \le x \le x_\nu \\ (x_{\nu+1}-x)^2(x-x_{\nu-2})/(h_{\nu+1}h_{23}h_{123}) + & \\ +(x_{\nu+2}-x)(x_{\nu+1}-x)(x-x_{\nu-1})/(h_{\nu+1}h_{23}h_{234}) + & \\ +(x_{\nu+2}-x)^2(x-x_\nu)/(h_{\nu+1}h_{34}h_{234}) & \text{if } x_\nu \le x \le x_{\nu+1} \\ (x_{\nu+2}-x)^3/(h_{\nu+2}h_{34}h_{234}) & \text{if } x_{\nu+1} \le x \le x_{\nu+2} \\ 0 & \text{otherwise} \end{cases}$$

where $h_{34} = h_{\nu+1} + h_{\nu+2}$, $h_{123} = h_{\nu-1} + h_\nu + h_{\nu+1}$ and $h_{234} = h_\nu + h_{\nu+1} + h_{\nu+2}$. For the derivative of spline functions, the following relation holds [8]

$$\frac{d}{dx}\left[\sum_{\nu=-1}^{N+1} c_\nu B_{\nu,k}(x)\right] = \sum_{\nu=-1}^{N+2} (k-1)\frac{c_\nu - c_{\nu-1}}{x_{\nu+k-3} - x_{\nu-2}} B_{\nu,k-1}(x) \qquad \text{(B.4)}$$

where $c_{-2} = c_{N+2} = 0$. Eq. (B.4) can be written as:

$$\begin{aligned} \frac{d}{dx}\left[\sum_{\nu=-1}^{N+1} c_\nu B_{\nu,k}(x)\right] &= \sum_{\nu=-1}^{N+1} (k-1)c_\nu \left(\frac{B_{\nu,k-1}(x)}{x_{\nu+k-3} - x_{\nu-2}} - \frac{B_{\nu+1,k-1}(x)}{x_{\nu+k-2} - x_{\nu-1}}\right) \\ &= \sum_{\nu=-1}^{N+1} c_\nu \tilde{B}_{\nu,k-1}(x) \qquad \text{(B.5)} \end{aligned}$$

Let us write explicitly $\tilde{B}_{\nu,3}$:

$$\tilde{B}_{\nu,3}(x) = 3 \begin{cases} (x - x_{\nu-2})^2/(h_{\nu-1}h_{12}h_{123}) & \text{if } x_{\nu-2} \le x \le x_{\nu-1} \\ (x_\nu - x)(x - x_{\nu-2})/(h_\nu h_{12}h_{123}) + & \\ +(x_{\nu+1} - x)(x - x_{\nu-1})/(h_\nu h_{23}h_{123}) + & \\ -(x - x_{\nu-1})^2/(h_\nu h_{23}h_{234}) & \text{if } x_{\nu-1} \le x \le x_\nu \\ (x_{\nu+1} - x)^2/(h_{\nu+1}h_{23}h_{123}) + & \\ -(x_{\nu+1} - x)(x - x_{\nu-1})/(h_{\nu+1}h_{23}h_{234}) + & \\ -(x_{\nu+2} - x)(x - x_\nu)/(h_{\nu+1}h_{34}h_{234}) & \text{if } x_\nu \le x \le x_{\nu+1} \\ -(x_{\nu+2} - x)^2/(h_{\nu+2}h_{34}h_{234}) & \text{if } x_{\nu+1} \le x \le x_{\nu+2} \\ 0 & \text{otherwise} \end{cases}$$

Using Eq. (B.4) (or, equivalently, Eq. (B.5)), we find that the following relations hold at the boundaries:

$$\begin{aligned} g'(x_0) &= \frac{3}{h_0 + h_1}\left(\frac{c_0 - c_{-1}}{x_1 - x_{-2}}h_1 + \frac{c_1 - c_0}{x_2 - x_{-1}}h_0\right) \\ &= \frac{3}{h_0 + h_1}\left[c_0\left(\frac{h_1}{x_1 - x_{-2}} - \frac{h_0}{x_2 - x_{-1}}\right) + \frac{c_1 h_0}{x_2 - x_{-1}} - \frac{c_{-1}h_1}{x_1 - x_{-2}}\right] \\ &= c_0 F_0 + c_1 F_1 + c_{-1}F_{-1} \qquad \text{(B.6)} \\ g'(x_N) &= \frac{3}{h_N + h_{N+1}}\left(\frac{c_N - c_{N-1}}{x_{N+1} - x_{N-2}}h_{N+1} + \frac{c_{N+1} - c_N}{x_{N+2} - x_{N-1}}h_N\right) \\ &= \frac{3}{h_N + h_{N+1}}[c_N\left(\frac{h_{N+1}}{x_{N+1} - x_{N-2}} - \frac{h_N}{x_{N+2} - x_{N-1}}\right) \\ &\quad + \frac{c_{N+1}h_N}{x_{N+2} - x_{N-1}} - \frac{c_{N-1}h_{N+1}}{x_{N+1} - x_{N-2}}] \\ &= c_N F_N + c_{N+1}F_{N+1} + c_{N-1}F_{N-1} \qquad \text{(B.7)} \end{aligned}$$

Using the previous two equations and the values of the cubic spline basis listed in the following table:

x	$x_{\nu-1}$	x_ν	$x_{\nu+1}$
$B_{\nu,4}(x)$	$D_\nu^{(-1)} = \frac{h_{\nu-1}^2}{h_{12}h_{123}}$	$D_\nu^{(0)} = \frac{1}{h_{23}}\left(\frac{h_{\nu+1}h_{12}}{h_{123}} + \frac{h_\nu h_{34}}{h_{234}}\right)$	$D_\nu^{(1)} = \frac{h_{\nu+2}^2}{h_{34}h_{234}}$
$B_{\nu,3}(x)$	$h_{\nu-1}/h_{12}$	$h_{\nu+1}/h_{23}$	0
$\tilde{B}_{\nu,3}(x)$	$3h_{\nu-1}/(h_{12}h_{123})$	$3(h_{\nu+1}/h_{123} - h_\nu/h_{234})/h_{23}$	$-3h_{\nu+2}/(h_{34}h_{234})$

the $(N+3, N+3)$ matricial system to be solved becomes[1]:

$$\begin{pmatrix} F_{-1} & F_0 & F_1 & & & 0 \\ D_{-1}^{(1)} & D_0^{(0)} & D_1^{(-1)} & & & \\ & D_0^{(1)} & D_1^{(0)} & D_2^{(-1)} & & \\ & \ddots & \ddots & \ddots & & \\ 0 & & D_{N-1}^{(1)} & D_N^{(0)} & D_{N+1}^{(-1)} \\ & & F_{N-1} & F_N & F_{N+1} \end{pmatrix} \times \begin{pmatrix} c_{-1} \\ c_0 \\ \vdots \\ \vdots \\ c_N \\ c_{N+1} \end{pmatrix} = \begin{pmatrix} g'(x_0) \\ g(x_0) \\ \vdots \\ \vdots \\ g(x_N) \\ g'(x_N) \end{pmatrix} \qquad \text{(B.9)}$$

If we permute the matrix to keep the boundary conditions in the last two rows, Eq. (B.9) reads:

$$\tilde{A}\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \text{where: } \begin{cases} u' = (c_0, \cdots, c_N)^t \\ v' = (c_{N+1}, c_{-1})^t \\ u = (g(x_0), \cdots, g(x_N))^t \\ v = (g'(x_N), g'(x_0))^t \end{cases}$$

[1]The coefficients c_ν are computed solving the following system of equations

$$\begin{aligned} g(x_i) &= \sum_{\nu=-1}^{N+1} c_\nu B_{\nu,4}(x_i) \\ &= c_{\nu-1}D_{\nu-1}^{(1)} + c_\nu D_\nu^{(0)} + c_{\nu+1}D_{\nu+1}^{(-1)} \end{aligned} \qquad \text{(B.8)}$$

and

$$\tilde{A} = \left(\begin{array}{c|cc} A & \multicolumn{2}{c}{\gamma} \\ \hline \lambda & \begin{array}{cc} \xi_1 & \xi_2 \\ \xi_3 & \xi_4 \end{array} \end{array}\right)$$

where:

$$\begin{cases} \cdot\, A \text{ is the } (N+1)\times(N+1) \text{ tridiagonal matrix} : \begin{pmatrix} D_0^{(0)} & D_1^{(-1)} & & & & \\ D_0^{(1)} & D_1^{(0)} & D_2^{(-1)} & & & \\ & \ddots & \ddots & \ddots & & \\ & & & D_{N-2}^{(1)} & D_{N-1}^{(0)} & D_N^{(-1)} \\ & & & & D_{N-1}^{(1)} & D_N^{(0)} \end{pmatrix} \\ \cdot\, \lambda \text{ is } 2\times(N+1) \text{ matrix} : \begin{pmatrix} 0 & \cdots & 0 & F_{N-1} & F_N \\ F_0 & F_1 & 0 & \cdots & 0 \end{pmatrix} \\ \cdot\, \gamma \text{ is } (N+1)\times 2 \text{ matrix} : \begin{pmatrix} D_{-1}^{(1)} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & D_{N+1}^{(-1)} \end{pmatrix}^t \\ \cdot\, \delta = \begin{pmatrix} \xi_1 & \xi_2 \\ \xi_3 & \xi_4 \end{pmatrix} = \begin{pmatrix} F_{N+1} & 0 \\ 0 & F_{-1} \end{pmatrix} \end{cases}$$

which can be solved with the same method described in Subsection B.1.1.

B.3.2 Periodic boundary conditions

In the periodic case, we use, as before, the continuity property on the first and second derivatives, which gives the following relations:

$$c_1F_1 + c_0F_0 + c_{-1}F_{-1} = c_{N-1}F_{N-1} + c_NF_N + c_{N+1}F_{N+1} \quad \text{(B.10)}$$
$$c_1G_1^{(-1)} + c_0G_0^{(0)} + c_{-1}G_{-1}^{(1)} = c_{N-1}G_{N-1}^{(1)} + c_NG_N^{(0)} + c_{N+1}G_{N+1}^{(-1)} \quad \text{(B.11)}$$

where the coefficients F_j are defined in the previous sections and G_j are given in the following table:

x	$x_{\nu-1}$	x_ν	$x_{\nu+1}$
$\tilde{B}_{\nu,2}(x)$	$G_\nu^{(-1)} = \frac{6}{h_{12}h_{123}}$	$G_\nu^{(0)} = -\frac{6}{h_23}\left[\frac{1}{h_{123}} + \frac{1}{h_{234}}\right]$	$G_\nu^{(1)} = \frac{6}{h_{34}h_{234}}$

Let us write explicitly $\tilde{B}_{\nu,2}$:

$$\tilde{B}_{\nu,2}(x) = 3 \begin{cases} 2(x - x_{\nu-2})/(h_{\nu-1}h_{12}h_{123}) & \text{if } x_{\nu-2} \le x \le x_{\nu-1} \\ (x_\nu + x_{\nu-2} - 2x)/(h_\nu h_{12}h_{123}) + & \\ +(x_{\nu+1} + x_{\nu-1} - 2x)/(h_\nu h_{23}h_{123}) + & \\ -2(x - x_{\nu-1})/(h_\nu h_{23}h_{234}) & \text{if } x_{\nu-1} \le x \le x_\nu \\ -2(x_{\nu+1} - x)/(h_{\nu+1}h_{23}h_{123}) + & \\ -(x_{\nu+1} + x_{\nu-1} - 2x)/(h_{\nu+1}h_{23}h_{234}) + & \\ -(x_{\nu+2} + x_\nu - 2x)/(h_{\nu+1}h_{34}h_{234}) & \text{if } x_\nu \le x \le x_{\nu+1} \\ 2(x_{\nu+2} - x)/(h_{\nu+2}h_{34}h_{234}) & \text{if } x_{\nu+1} \le x \le x_{\nu+2} \\ 0 & \text{otherwise} \end{cases}$$

The $(N+3, N+3)$ matricial system to be solved becomes

$$\tilde{A}\begin{pmatrix} u' \\ v' \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} \qquad \text{where:} \begin{cases} u' = (c_0, \cdots, c_N)^t \\ v' = (c_{N+1}, c_{-1})^t \\ u = (g(x_0), \cdots, g(x_N))^t \\ v = (0,0)^t \end{cases}$$

and

$$\tilde{A} = \left(\begin{array}{c|cc} A & \multicolumn{2}{c}{\gamma} \\ \hline \lambda & \xi_1 & \xi_2 \\ & \xi_3 & \xi_4 \end{array}\right)$$

where:

$$\begin{cases} \cdot\, A \text{ is the } (N+1)\times(N+1) \text{ tridiagonal matrix : } \begin{pmatrix} D_0^{(0)} & D_1^{(-1)} & & & \\ D_0^{(1)} & D_1^{(0)} & D_2^{(-1)} & & \\ & \ddots & \ddots & \ddots & \\ & & D_{N-2}^{(1)} & D_{N-1}^{(0)} & D_N^{(-1)} \\ & & & D_{N-1}^{(1)} & D_N^{(0)} \end{pmatrix} \\ \cdot\, \lambda \text{ is } 2\times(N+1) \text{ matrix : } \begin{pmatrix} F_0 & F_1 & 0 \cdots & -F_{N-1} & -F_N \\ G_0^{(0)} & G_1^{(-1)} & 0 \cdots & -G_{N-1}^{(1)} & -G_N^{(0)} \end{pmatrix} \\ \cdot\, \gamma \text{ is } (N+1)\times 2 \text{ matrix : } \begin{pmatrix} D_{-1}^{(1)} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & D_{N+1}^{(-1)} \end{pmatrix}^t \\ \cdot\, \delta = \begin{pmatrix} \xi_1 & \xi_2 \\ \xi_3 & \xi_4 \end{pmatrix} = \begin{pmatrix} -F_{N+1} & F_{-1} \\ -G_{N+1}^{(-1)} & G_{-1}^{(1)} \end{pmatrix} \end{cases}$$

which can be solved with the same method described in Subsection B.1.1.

B.4 Integration of cubic splines

In the case of equidistant mesh the following relations can be easily calculated

$$\begin{aligned}
\int_{x_{\nu-2}}^{x_{\nu-1}} \Lambda_\nu(x)dx &= \int_{x_{\nu+1}}^{x_{\nu+2}} \Lambda_\nu(x)dx = h/24 && \text{(B.12)}\\
\int_{x_{\nu-1}}^{x_{\nu}} \Lambda_\nu(x)dx &= \int_{x_{\nu}}^{x_{\nu+1}} \Lambda_\nu(x)dx = 11h/24 && \text{(B.13)}
\end{aligned}$$

so that the sum gives

$$\int_{x_{\nu-2}}^{x_{\nu+2}} \Lambda_\nu(x)dx = h \tag{B.14}$$

The case of non-equidistant mesh presents more algebraic calculations. The final result is

$$\begin{aligned}
\int_{x_{\nu-2}}^{x_{\nu-1}} B_{\nu,4}(x)dx &= h^3_{\nu-1}/(4h_{12}h_{123}) && \text{(B.15)}\\
\int_{x_{\nu-1}}^{x_{\nu}} B_{\nu,4}(x)dx &= h_\nu G_\nu/(4h_{12}h_{23}h_{123}h_{234}) && \text{(B.16)}\\
\int_{x_{\nu}}^{x_{\nu+1}} B_{\nu,4}(x)dx &= h_{\nu+1}H_\nu/(4h_{23}h_{34}h_{123}h_{234}) && \text{(B.17)}\\
\int_{x_{\nu+1}}^{x_{\nu+2}} B_{\nu,4}(x)dx &= h^3_{\nu+2}/(4h_{34}h_{234}) && \text{(B.18)}
\end{aligned}$$

where G_ν is the following function

$$\begin{aligned}
G_\nu &= (x_{\nu-1}+x_\nu)(x^2_{\nu-1}+x^2_\nu)(x_{\nu-1}-x_{\nu+1}) + \\
&- (x^3_{\nu-1}+x_{\nu-1}x^2_\nu + x^2_\nu(x_\nu - 3x_{\nu+1}) + x^2_{\nu-1}(x_\nu - x_{\nu+1}))\, x_{\nu+2} + \\
&+ x^2_{\nu-2}(3x^2_{\nu-1} - x^2_\nu + 4x_{\nu+1}x_{\nu+2} + 2x_{\nu-1}(x_\nu - 2(x_{\nu+1}+x_{\nu+2}))) + \\
&+ x_{\nu-2}(-3x^3_{\nu-1} + 3x^2_{\nu-1}(-x_\nu + x_{\nu+1} + x_{\nu+2}) + \\
&+ x_\nu(x^2_\nu - 6x_{\nu+1}x_{\nu+2} + x_\nu(x_{\nu+1}+x_{\nu+2})) + \\
&+ x_{\nu-1}(-3x^2_\nu - 2x_{\nu+1}x_{\nu+2} + 4x_\nu(x_{\nu+1}+x_{\nu+2}))) && \text{(B.19)}
\end{aligned}$$

and H_ν has an analogous expression. But, instead of using the previous expression, it is easier to use Gaussian quadrature to calculate the integrals in Eqs. (B.16)-(B.17). Indeed, for a polynomial of degree 3, the result obtained with a 2-point Gaussian quadrature is exact[2]. By the linear x transformation

$$t = [2x - (x_a + x_b)]/(x_b - x_a), \qquad x = [(x_b - x_a)t + (x_a + x_b)]/2 \quad \text{(B.20)}$$

the interval $[x_a, x_b]$ can be transformed to $[-1, 1]$ and then

$$\begin{aligned} \int_{x_a}^{x_b} f(x)dx &= \frac{x_b - x_a}{2} \int_{-1}^{1} f(t)dt \\ &= x_c[f(x_m - x_c t_G) + f(x_m + x_c t_G)] \end{aligned} \quad \text{(B.21)}$$

where $x_m = (x_a + x_b)/2$, $x_c = (x_b - x_a)/2$ and t_G is the Gaussian point $t_G = 0.577350269189626$ (for the 2-point formula, the Gaussian weigth is equal to 1).

B.5 Dirichlet boundary conditions

In order to impose Dirichlet boundary conditions, we need to change only the three splines nearest to the boundary. The transformation is such that the sum of the basis functions at any given point remains equal to one[3].

In the case of equidistant mesh, the transformation is given by

$$\begin{pmatrix} \hat{B}_{-1,4} \\ \hat{B}_{0,4} \\ \hat{B}_{1,4} \end{pmatrix} = \begin{pmatrix} 6 & 0 & 0 \\ -4 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} B_{-1,4} \\ B_{0,4} \\ B_{1,4} \end{pmatrix} \quad \text{(B.22)}$$

and similarly on the outside boundary:

$$\begin{pmatrix} \hat{B}_{N-1,4} \\ \hat{B}_{N,4} \\ \hat{B}_{N+1,4} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & -4 \\ 0 & 0 & 6 \end{pmatrix} \times \begin{pmatrix} B_{N-1,4} \\ B_{N,4} \\ B_{N+1,4} \end{pmatrix} \quad \text{(B.23)}$$

[2]For the integration of the product of two Λs, 4-points Gaussian quadrature is needed, and the same rule can be generalized to integrate polynomials of higher order.

[3]One can easily check that this condition is satisfied by $B_{\nu,k}$. For $k = 4$, we have indeed:

$$\begin{aligned} B_{\nu,4}(x_\nu) + B_{\nu-1,4}(x_\nu) + B_{\nu+1,4}(x_\nu) &= \frac{1}{h_{23}} \left(\frac{h_{\nu+1} h_{12}}{h_{123}} + \frac{h_\nu h_{34}}{h_{234}} \right) \\ + \quad \frac{h_\nu^2}{h_{23} h_{234}} + \frac{h_{\nu+1}^2}{h_{23} h_{123}} &= \frac{1}{h_{23}}(h_\nu + h_{\nu+1}) = 1 \end{aligned}$$

In the case of non-equidistant mesh, the transformation is

$$\begin{pmatrix} \hat{B}_{-1,4} \\ \hat{B}_{0,4} \\ \hat{B}_{1,4} \end{pmatrix} = \begin{pmatrix} E_0 & 0 & 0 \\ C_0 & 1 & 0 \\ A_0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} B_{-1,4} \\ B_{0,4} \\ B_{1,4} \end{pmatrix} \tag{B.24}$$

and on the outside boundary:

$$\begin{pmatrix} \hat{B}_{N-1,4} \\ \hat{B}_{N,4} \\ \hat{B}_{N+1,4} \end{pmatrix} = \begin{pmatrix} 1 & 0 & A_N \\ 0 & 1 & C_N \\ 0 & 0 & E_N \end{pmatrix} \times \begin{pmatrix} B_{N-1,4} \\ B_{N,4} \\ B_{N+1,4} \end{pmatrix} \tag{B.25}$$

where:

$$\begin{aligned}
A_0 &= -\frac{h_0^2(h_{-1}+h_0+h_1)}{h_1^2(h_0+h_1+h_2)} && \text{(B.26)} \\
C_0 &= -[(h_{-1}+h_0)/h_1 && \text{(B.27)} \\
&+ \frac{h_0(h_1+h_2)(h_{-1}+h_0+h_1)}{h_1^2(h_0+h_1+h_2)}] \\
E_0 &= 1-(A_0+C_0) && \text{(B.28)} \\
A_N &= -\frac{h_{N+1}^2(h_N+h_{N+1}+h_{N+2}}{h_N^2(h_{N-1}+h_N+h_{N+1})} && \text{(B.29)} \\
C_N &= -[(h_{N+1}+h_{N+2})/h_N \\
&+ \frac{h_{N+1}(h_N+h_{N-1})(h_N+h_{N+1}+h_{N+2})}{h_N^2(h_{N-1}+h_N+h_{N+1})}] && \text{(B.30)} \\
E_N &= 1-(A_N+C_N) && \text{(B.31)}
\end{aligned}$$

Bibliography

[1] V. Grandgirard *et al*, *29th EPS Conference on Controlled Fusion and Plasma Physics* Montreaux, 17-21 June 2002, poster P4.095 (see http://crppwww.epfl.ch/archives)

[2] M. Brunetti *et al*, *29th EPS Conference on Controlled Fusion and Plasma Physics* Montreaux, 17-21 June 2002, poster P4.102 (see http://crppwww.epfl.ch/archives)

[3] V. Grandgirard, N. Besse and E. Sonnendrücker, *Amélioration des schemas numeriques semi-lagrangiens associes a la resolution des equations de Vlasov non-lineaires 4D pour l'etude de la turbulence plasma*, Project CEMRACS 2003

[4] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes in Fortran*, 2nd edition (Cambridge University Press, 1992), p. 716.

[5] F. Filbet, E. Sonnendrücker and P. Bertrand, *Conservative Numerical Schemes for the Vlasov Equation*, J. Comp. Physics **172**, 166 (2001)

[6] C. Z. Cheng and G. Knorr, *The integration of the Vlasov equation in configuration spaces*, J. Comp. Physics **22**, 330 (1976)

[7] G. Manfredi, *Long-time Behavior of Nonlinear Landau Damping*, Phys. Rev. Lett. **79**, 2815 (1997)

[8] C. deBoor, *A practical guide to splines*, Applied Mathematical Sciences, 27 (Springer-Verlag, New York, 2001)