Design of an object oriented Vlasov-Fokker-Planck code based on the expansion of the distribution function to spherical harmonics

Contact m.tzoufras1@physics.ox.ac.uk

M. Tzoufras and A. R. Bell

Department of Physics, University of Oxford, Clarendon Laboratory, Parks Road, Oxford OX1 3PU, UK

The design of a parallel object oriented Vlasov-Fokker-Planck code which utilizes the expansion of the electron distribution function to spherical harmonics is presented. The main ideas of this design have been instantiated for a 1D parallel electrostatic Vlasov code. The performance of the code in terms of speed, accuracy and stability is discussed for two standard 1D plasma physics problems: the expansion of hot plasma and the two-stream instability.

Fast ignition for inertial confinement fusion relies on the electrons at the surface of a compressed pellet to absorb the energy of an ultraintense laser pulse and efficiently transfer it to the core of the pellet [1]. However, what happens when hot electrons propagate through dense plasma is not well understood. Electron transport involves a variety of physical mechanisms and instabilities, which can slow down and even stop the flow of electrons. In order to understand the physics of electron transport, a new parallel Vlasov-Fokker-Planck simulation code is needed, such that the full 3D distribution of electrons can be modelled.

The Vlasov-Maxwell system of equations can be expressed in terms of normalized units, where velocity is normalized to the speed of light c, length to the skin depth c/ω_e, density to the plasma density n_e, charge to the electron charge e, and fields to $mc\omega_e/e$. The Vlasov equation is:

$$\frac{\partial f_e}{\partial t} + \upsilon\cdot\nabla_r f_e - (E+\upsilon\times B)\cdot\nabla_p f_e = 0 \tag{1}$$

and Maxwell's equations are:

$$\nabla_r \times E = -\frac{\partial B}{\partial t} \tag{2}$$

$$\nabla_r \times B = J + \frac{\partial E}{\partial t} \tag{3}$$

where the current J is given by the constitutive relation:

$$J = \sum_j q_j \int (dp)\upsilon f_j(p) \tag{4}$$

and the initial condition is given by Gauss's law:

$$\nabla_r \cdot E = \rho = \sum_j q_j \int (dp) f_j(p) \tag{5}$$

The system of equations (1)-(3) can be used to advance the electron distribution function f_e and the fields E,B in time. We may define the "state" of the system $Y = [f_e,E,B]$ to rewrite these equations in the general form:

$$\frac{\partial Y}{\partial t} = F(Y) \tag{6}$$

where the operator F involves Eqs. (1)-(5). The particular form of the operator F depends on the expansion of the distribution function and the integration in time can be performed using a standard Runge-Kutta method.

The electron distribution function f_e can be expressed in spherical harmonics as in (see Ref. [2]):

$$f_e(r,p,t) = \sum_{\ell=0}^{\infty}\sum_{m=-\ell}^{\ell} f_\ell^m(r,p,t)P_\ell^{|m|}(cos\theta)e^{im\phi} \tag{7}$$

where $f_\ell^{-m} = (f_\ell^m)^*$ and P_ℓ^m are the associated Legendre functions:

$$P_\ell^m(x) = \frac{(-1)^m}{2^\ell \ell!}(1-x^2)^{m/2}\frac{d^{\ell+m}}{dx^{\ell+m}}(x^2-1)^\ell \tag{8}$$

The expansion in Eq. (7) may then be truncated for some ℓ_0 and $m_0 \le \ell_0$ to yield an expression for the approximate distribution function $\tilde{f}_e$.

Substituting $f_\ell^m e^{im\phi} + f_\ell^{-m}e^{-im\phi} = 2\Re[f_\ell^m e^{im\phi}]$:

$$\tilde{f}_e(r,p,t) = \sum_{\ell=0}^{\ell_0} f_\ell^0 P_\ell^0(cos\theta) + \sum_{\ell=0}^{\ell_0}\sum_{m>0}^{min(\ell,m_0)} 2\Re[f_\ell^m e^{im\phi}]P_\ell^m(cos\theta) \tag{9}$$

The distribution function f_e is then represented by the complex amplitudes $f_\ell^m(r,p,t)$ of the spherical harmonics, and the operator F needs to be expressed in terms of combinations of f_ℓ^m.

For the 1D2V electrostatic Vlasov equations, for which $m_0 = 0$, Eq. (9) reduces to:

$$\tilde{f}_e(r,p,t) = \sum_{\ell=0}^{\ell_0} f_\ell^0(r,p,t)P_\ell^0(cos\theta) \tag{10}$$

and the Vlasov-Maxwell equations become:

$$\frac{\partial}{\partial t}\begin{bmatrix} E_x \\ f_\ell^0 \end{bmatrix} = \begin{bmatrix} -J_x \\ \mathsf{E}_{\ell,x}^0 + \mathsf{A}_{\ell,x}^0 \end{bmatrix} \tag{11}$$

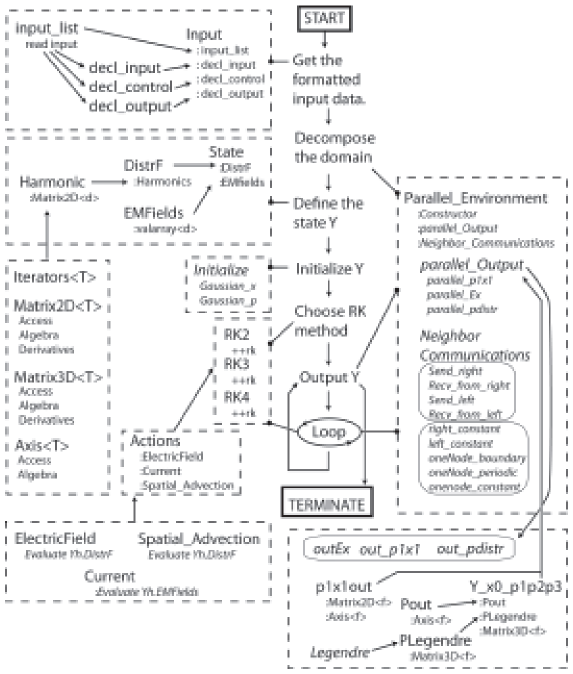

Figure 1. Structure of the 1D2V Vlasov code. The broken rectangles represent the different logical entities (and files) in the code.

where J_x is the current, $\mathsf{A}^0_{\ell,x}$ the contribution of the advection term and $\mathsf{E}^0_{\ell,x}$ is the contribution of the electric field (see Ref. [2]).

The simulation code used to model Eq. (11) is written in C++ [3] and is composed of the following entities (developed and tested independently and shown graphically in Fig. 1):

- The "input" class, which reads the input values and formats them appropriately.
- The "matrices" library which contains the basic data structures---axis, matrices and iterators to these matrices---and their operations.
- The definition of the "state" $Y = [f, E_x]$ of the system, where f is a collection of the spherical harmonics.
- The initialization of the "state" $Y = [f, E_x]$.
- The definition of the Runge-Kutta methods that are available.
- An interface for the actions included in the operator F. This interface allows us to isolate non-interacting components of the operator F.
- Each component of the operator F is implemented as a separate class. For the 1D2V electrostatic code there are three classes (J_x, $\mathsf{A}^0_{\ell,x}$ and $\mathsf{E}^0_{\ell,x}$).
- A parallel module separated in four parts: (1) the decomposition of the computational domain, (2) the parallel output, (3) the exchange of information between precesses and (4) the boundary effects.
- An output module that formats and exports the output data.

The main loop of the code involves the integration of the state Y using a Runge-Kutta method accurate to order 2,3 or 4. High order methods can be expected to be more accurate and stable at the expense of additional computational resources. The costs associated with using high order methods are due to the increased amount of calculations required between "states", due to a larger number of operator F calculations and due to the additional memory needed. Tests using a single process to investigate the difference in speed between these methods yield:

$$RK2 : RK3 : RK4 = 1 : 1.6 : 2.8 \tag{12}$$

where the right hand side is in arbitrary time units.

Even though lower order methods are much faster, the lower the order of the Runge-Kutta method the less stable it is. In Ref. [4] the stability of RK methods of order one to four is discussed, and it is shown that the stability properties of a simulation can be improved by either increasing the order of the Runge-Kutta method or by reducing the time step Δt.

Here we discuss the issue of stability of the RK methods for a hot expanding plasma blob. The initial electric field is assumed to be $E_x = 0$, which means that there are ions overlapping the hot electrons. The electrons expand because of the thermal pressure and the resulting charge separation causes an electric field to build up around the blob, which starts pulling some electrons back on axis. As these reflecting electrons cross $p = 0$ there are two distinct numerical effects that can cause the code to be inaccurate or unstable.

The reason for the possible loss of accuracy lies in that, for a system described in spherical coordinates, a boundary condition needs to be invoked at $p = 0$. The inaccuracy can be ameliorated by increasing the resolution of the grid. On the other hand, numerical instability may occur if the time step   is not small enough compared to the cell size, therefore the cells near $p = 0$, by virtue of being the smallest are also the most unstable ones. The instability can be suppressed by using finer time resolution (or a higher order method). As a result, if we increase the resolution of the grid to improve the accuracy of the code we must also increase the resolution in time.

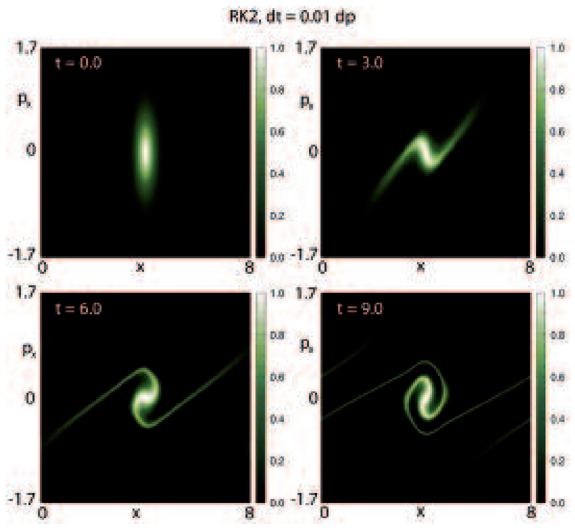

Figure 2. Four snapshots of the simulation of an expanding plasma blob. The simulation parameters are $\ell_0 = 180$, $nump = 256$, $p_{max} = 1.7$, $numx = 384$, $x_{min} = 0$, $x_{max} = 8$, $\Delta t = 0.01\Delta p$, $RKlevel = 2$.

The blob keeps folding under the influence of the thermal pressure and the electric field until it reaches the limits of the computational grid. (The boundaries are periodic).

In Fig. 2 a simulation with RK2 is shown, in which no numerical instability developed. The plasma blob continually folds due to the combination of thermal expansion and electric fields. The plasma structure becomes finer and finer over time until it reaches the limits of the computational grid, at which point (not shown in Fig. 2) the simulation loses accuracy. The simulation parameters are:

$\ell_0 = 180$, $nump = 256$, $p_{max} = 1.7$, $numx = 384$, $x_{min} = 0$, $x_{max} = 8$, $\Delta t = 0.01\Delta p$.

The simulation ran on 8 processors for 28 hours to reach $\omega_p t = 9$.

In order to examine the stability of the different Runge-Kutta methods we increased the time step Δt to $0.02\Delta p$, $0.04\Delta p$ and $0.08\Delta p$ and repeated the above simulation for RK2, RK3 and RK4. For $\Delta t = 0.02\Delta p$ the RK2 method fails early as the first reflected electrons cross $p = 0$, while for both RK3 and RK4 the electrons cross $p = 0$ without any issues. The simulations were repeated for $\Delta t = 0.04\Delta p$ and again RK3 and RK4 did not exhibit unstable behavior near $p = 0$. For $\Delta t = 0.08\Delta p$ the RK3 simulation is unstable near $p = 0$ while RK4 is stable. Multiplying the ratio between the largest time step for which RK2 is stable and that for which RK4 is stable, 0.01/0.08, with the advantage of the RK2 method shown in Eq. (12) we obtain $2.8/8 \approx 0.35$ so that RK4 can be expected to be 2.9 times faster than RK2! In this last case the RK4 simulation generated output ($\omega_e t_{out} = 1$) every 50 minutes, so that to reach $\omega_e t = 9$ it would take 7.5 hours.

$$RK2 : RK4 = 2.8 : 7.5 = 3.7 : 1 \tag{13}$$

So the actual advantage of RK4 is even larger than the back-of-envelope calculation above suggested. This is because the use of a large time step (in RK4 compared to RK2) minimizes the need for communications between processes. Therefore, when numerical stability is one of the main concerns for a given problem, high order methods can be much faster than low order methods. We note that for the high resolution cases discussed above the simulation results were exactly identical regardless of the order of the method used (provided the methods were stable). On the other hand, if the grid resolution is low, in which case Δt can be large, the main numerical issue is loss of accuracy due to the inadequate resolution. For such low resolution cases, the RK2 method can be significantly faster than the higher order methods.

There is however a way to circumvent having to use very small Δt. The instability for large Δt is caused by the fact that the high harmonics are not resolved appropriately near $p = 0$. One may instead artificially set these harmonics equal to zero. This will generate noise (but no instability) near $p = 0$. However, for $p = 0$ collisions damp the high order harmonics quickly so the artificial noise is smeared out in a sufficiently collisional plasma. It is therefore reasonable to maintain a filter for the high order harmonics near $p = 0$ and to allow the user of the code to set the level of filtering required (including no filtering). One can do many runs with large Δt including substantial filtering, and few runs for small Δt with little or no filtering. Such a filter has been added to our code but was turned off for the simulations shown here.

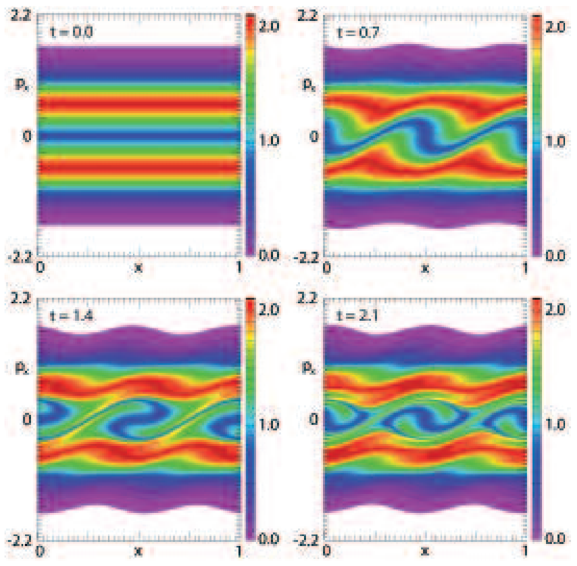

Figure 3. Four snapshots of the simulation of the two-stream instability for two counter-streaming electron beams in the frame of reference of the unstable wave. The simulation parameters are $\ell_0 = 200$, $nump = 348$, $p_{max} = 2.2$, $numx = 400$, $x_{min} = 0$, $x_{max} = 1$, $\Delta t = 0.04\Delta p$, $RKlevel = 4$.

In the absence of collisions and filtering the computational requirements can, for certain problems (e.g. the two-stream instability), be extremely restrictive. In Fig. 3 we present the early evolution of the two-stream instability for two counter-propagating electron beams in the frame of reference of the unstable wave. The instability was seeded with a sinusoidal electric field and its behavior closely mirrors that discussed in standard plasma physics textbooks[5]. However, as the trapped part of the electron distribution continuously folds around $p = 0$ the simulation is susceptible loss of accuracy. Increasing the resolution and decreasing the time step allows us to extend the simulation further into the nonlinear regime. Unfortunately this can only be done at huge computational expense.

Adding a collisional operator to the Vlasov equation will allow us to relax the computational requirements and will make it easier to simulate problems such as the plasma expansion and the two-stream instability. By forcing a group of electrons to continually oscillate around $p = 0$ these two problems are very difficult to approach using spherical coordinates, and they are among the most challenging for our code. It is therefore important to simulate them to test the limits of applicability of the code.

The generalization of this code to two and three spatial dimensions, and the addition of the electromagnetic components of the operator F as well as the collisional operator, will be pursued using the design presented in Fig. 1. We have thus presented a program for developing parallel object oriented Vlasov-Fokker-Planck codes, and we have shown, that even for problems that attack the most vulnerable aspects of the code, simulations can be performed stably and with extremely high accuracy.

Simulations were performed at the Hoffman2 cluster in UCLA.

The instability keeps growing but until particle trapping occurs. (The boundaries are periodic).

References

1. M. Tabak, J. Hammer, M. E. Glinsky, W. L. Kruer, S. C. Wilks, J. Woodworth, W. M. Campbell, M. D. Perry, R. J. Mason, *Phys. Plasmas* **1**, 1626 (1994); S. C. Wilks, W. L. Kruer, M. Tabak, A. B. Langdon, *Phys. Rev. Lett*. **69**, 1383 (1992).
2. A. R. Bell, A. P. L. Robinson, M. Sherlock, R. J. Kingham and W. Rozmus, *"Fast electron transport in laser-produced plasmas and the KALOS code for solution of the Vlasov-Fokker-Planck equation"*, *Plasma Phys. Control. Fusion* **48**, R37-R57 (2006).
3. B. Stroustrup, *The C++ Programming Language, Special Edition*, Addison-Wesley.
4. G. E. Karniadakis and R. M. Kirby II, *Parallel Scientific Computing in C++ and MPI*, Cambridge University Press.
5. F. F. Chen, *Plasma Physics and Controlled Fusion*, Springer, p. 236.