Massively parallel Vlasov simulation of electromagnetic drift-wave turbulence

Frank Jenko
Max-Planck-Institut für Plasmaphysik, EURATOM Association, Boltzmannstraße 2, 85748 Garching, Germany
Received 20 September 1999

Abstract
We use a hybrid model of drift-kinetic electrons and fluid ions to treat electromagnetic drift-wave turbulence in an inhomogeneous collisionless plasma confined by a strong external magnetic field. The electrons are described in terms of a distribution function in five-dimensional phase space, and the basic nonlinear equations are solved numerically on a massively parallel computer using explicit finite-difference methods of the upwind and predictor-corrector type. The algorithm as well as its parallelization are explained, the parallel speedup is measured, and various numerical tests are performed. © 2000 Elsevier Science B.V. All rights reserved.

1. Introduction
One of the most important open problems in fusion research is the computation of turbulent (“anomalous”) particle and heat transport in magnetically confined plasmas which can exceed collision-induced (“neoclassical”) transport by up to two orders of magnitude [1]. In addition to being an inherently nonlinear phenomenon involving many spatial and temporal scales, plasma turbulence in fusion devices is characterized by the following difficulties:
(1) it can be driven by a large zoo of (linear and nonlinear) instabilities,
(2) it takes place in a rather complicated toroidal magnetic field geometry, and
(3) it needs to be described by kinetic equations (one for each particle species) valid in the long mean-free-path regime that the plasma particles are usually in.
For these reasons, the numerical simulation of plasma turbulence is considered to be one of the Grand Challenges in High Performance Computing [2].

Drift-wave instabilities are believed to contribute significantly to the anomalous transport in the edge of fusion plasmas where they are in a marginally electromagnetic and collisionless regime [3,4]. As the parallel ion dynamics only plays a minor role in drift-wave physics, we use a hybrid model of drift-kinetic electrons and fluid ions to treat electromagnetic drift-wave turbulence in a strongly magnetized, collisionless plasma. The electrons are described in terms of a distribution function in five-dimensional phase space, leading to algorithms which are both very CPU time and memory consuming. Therefore studies like the present one are only feasible on massively parallel computers like the Cray T3E we are using. For reasons to be described later, our numerical scheme is based on explicit finite-difference methods, contrasting with the more conventional gyrokinetic particle approach [5].
which consists of solving the equations of motion for an ensemble of “superparticles”. In Section 2 we present the dimensionless nonlinear equations defining our model. The parallelized algorithm used to solve them is explained in Section 3. Special attention is paid to the inclusion of electromagnetic effects which allow for a larger time step but are somewhat difficult to deal with numerically [6–9]. The measured parallel speedup reveals that our kinetic turbulence code scales almost linearly with the number of processors if the problem size per processor is held fixed. In Section 4 we present numerical tests concerning the correct representation of linear waves as well as convergence studies and comparisons with a companion Landau-fluid model. Finally, we summarize some main points in Section 5.

2. Model equations

To treat collisionless drift-wave turbulence in a strongly magnetized plasma we use a hybrid model of drift-kinetic electrons and (singly charged) cold ions in a three-dimensional sheared slab geometry. The derivation of the model equations is given in detail in two previous papers [4,10]. Therefore we will immediately write down the dimensionless nonlinear equations for the fluctuating part of the electron distribution function, \( f(x, y, z, w_1, \mu, t) \) (\( w_1 \) and \( \mu \) are the parallel velocity and magnetic moment of the electrons), the parallel ion velocity, \( u_\parallel(x, y, z, t) \), the electrostatic potential, \( \phi(x, y, z, t) \), and the parallel component of the vector potential, \( A_\parallel(x, y, z, t) \). They are given by

\[
\begin{align*}
\frac{dx_i}{dt} &= (v_x + \alpha e w_1) B_i, \\
\frac{d}{dt}f_m &= \alpha e w_1 \nabla_\parallel (f - f_0) + \alpha e w_1 f_m \beta \partial_t A_\parallel, \\
\frac{d}{dt}A_\parallel &= -\nabla_\parallel \phi - \beta \partial_t A_\parallel, \\
\frac{d}{dt}w_1 &= -\nabla_\parallel \psi - \nabla_\parallel J_\parallel, \\
\end{align*}
\]

with the auxiliary equation

\[
J_\parallel = u_\parallel - \int \alpha e w_1 f \, d^3 w = -\nabla_\parallel \psi,
\]

the differential operators

\[
\begin{align*}
\frac{d}{dt} &= \partial_t + z \times \nabla \phi \cdot \nabla = \partial_t + v_x \partial_x + v_y \partial_y, \\
\nabla_\parallel &= \partial_x - \beta z \nabla A_\parallel, \\
\nabla_\parallel^2 &= (\partial_x + \beta z \partial_y)^2 + \partial_y^2,
\end{align*}
\]

where

\[
\alpha_T = 1 + \eta_e (w^2 - 3/2),
\]

and the background Maxwellian \( f_m = \pi^{-3/2} e^{-w^2} \) where \( w^2 = w_{\parallel}^2 + \mu \). The model parameters are defined in the following way: \( \hat{\beta} = \left( q R / L_n \right)^2 \) characterizes the ratio of the connection length \( 2\pi q R \) and the background density scale length \( L_n = |\nabla \ln n_0|^{-1} \); \( \hat{\mu} = \mu e \hat{\beta} = 2\alpha e^{-2} \) with the electron to ion mass ratio \( \mu e = m_e / M_i \); \( \hat{\beta} = \beta e \hat{\beta} \) with the electron plasma beta \( \beta e = 4\pi n_0 T_{e0} / B_0^2 \); \( \eta_e = L_n / L_{Te} \) with the background electron temperature scale length \( L_{Te} = |\nabla \ln T_{e0}|^{-1} \); \( \hat{s} \) is the usual magnetic shear parameter. The dependent and independent variables, \( f, u_\parallel, \phi, A_\parallel \), and \( x, y, z, \mu, \psi \), are given in units of \( \delta n_0 v_T^2, \delta c e q R / L_n, \delta T_{e0} / c, \delta \beta e B_0 \rho_s q R / L_n \), and \( \rho_s, \mu_s, q R, \nu_T, \mu_T, L_n / c_s \), respectively, where \( \mu_T = m_e v_T^2 / 2 B_0 \), \( \nu_T = (2T_{e0} / m_e)^{1/2} \), \( \delta = \rho_s / L_n \), \( \rho_s = c_s / \Omega_i \), \( c_s = (T_{e0} / M_i)^{1/2} \), \( \Omega_i = e B_0 / M_i c \), and \( B_0 \) is the equilibrium magnetic field strength.

The boundary conditions in the spatial computational domain

\[
D = [-L_x/2, L_x/2] \times [-L_y/2, L_y/2] \times [-\pi, \pi]
\]

are given by
for any scalar quantity $S$ [11,12]. Note that for consistency reasons, the constraint $2\pi \hat{s} = n_s (L_y/L_x)$ with integer $n_s$ needs to be obeyed, resulting in a quantization of the aspect ratio $L_y/L_x$ of the simulation box holding $\hat{s} \neq 0$ constant or vice versa.

The following feature of the above equations can be used to improve the numerical efficiency greatly. Due to the absence of collisions and magnetic curvature effects, the velocity space variable $\mu$ can be integrated out such that Eq. (1) is replaced by the decoupled equations

$$
\frac{df}{dt} = (v_x + \alpha_e w_{||} B_z) \omega_f^N f_m^N - \alpha_e w_{||} \nabla_\parallel (f^N - f^0) + \alpha_e w_{||} f_m^N \partial_\parallel A
$$

for the distribution functions

$$
f^N(x, y, z, w_{||}, t) = \int_{0}^{\infty} \pi \mu^N f(x, y, z, w_{||}, \mu, t) d\mu
$$

where $f_m^N = N! \pi^{-1/2} e^{-w_{||}^2}$ and $\omega_f^N = 1 + \eta_e (w_{||}^2 + N - 1/2)$. It is important to note that this is not an approximation but an exact reformulation of the original problem, just reducing the set of fluid moments that can be computed from the solution. In principle, it is sufficient to keep $f^0$ which is needed in Eq. (4). However, depending on which velocity space moments one is interested in, the maximum $N$ can be chosen to be greater than zero. In the nonlinear simulations presented in an earlier paper [4] and in Section 4, we kept $f^0$ and $f^1$.

3. Numerical algorithm

3.1. Choice of the method

A fairly wide variety of numerical methods could be employed to solve the system (1)–(7) of nonlinear partial integro-differential equations. This necessitates some remarks concerning the nature of these different methods and why the Vlasov approach was chosen.

The algorithms for solving kinetic equations can be roughly divided into two groups corresponding to the Lagrangian and Eulerian description of phase space dynamics:

(a) Particle simulations [13,14] are based on the numerical solution of the equations of motion for an ensemble of $N_p$ “superparticles” sampling the phase space dynamics. As $N_p$ needs to be much smaller than the actual number of physical particles, this leads to fairly large numerical noise (proportional to $N_p^{-1/2}$) which can be reduced but not eliminated by $\delta f$ methods [15]. Therefore the convergence with particle number is always an important (and tough) issue to address. On the other hand, this method is very intuitive and fairly efficient in more than one spatial dimension. Particle simulations have been used very often to study collisionless ion temperature gradient (ITG) turbulence [5,16,17].

(b) One can also solve kinetic equations directly on a grid in phase space [18]. There are mainly two reasons for considering this approach instead of particle methods. Firstly, the time evolution of the distribution function is computed at fixed positions so that all regions of phase space are equally well represented. Secondly, Vlasov methods do not produce numerical noise, an advantage that is particularly important with respect to fast particles which are generally characterized by low phase space densities but can still have a big effect on higher-order velocity space moments. The drawback of Vlasov simulations is that they usually are computationally expensive due to the use of a grid in velocity space and therefore higher dimensionality.
However, the question whether the Vlasov or the particle approach leads to more efficient numerical algorithms has no general answer. The choice of the method depends on many details of the problem under consideration such as its dimensionality, (non)linearity, or phase space homogeneity. For three-dimensional computations of plasma turbulence both techniques seem to be similarly well suited. E.g., Dimits et al. [19] used four million particles on a grid of the size 128×128×32 to study ITG turbulence. Because each particle is characterized by five phase space coordinates and at least one weight (using a $\delta f$ method), one has to keep track of some 50 variables per spatial grid cell. This number is expected to be even larger for drift wave turbulence with its very small scale fluctuations of the vorticity $\nabla^2 \phi$ [20]. This is because $N_p$ is inversely proportional to the smallest relevant wavelength in the system [13]. Therefore one expects the numerical effort of particle and Vlasov simulations of drift-wave turbulence to be comparable. We will verify this statement in Section 4.2.

Because it exhibits no numerical noise despite being similarly efficient, we use a Vlasov approach to solve the nonlinear kinetic model equations (1)–(7) numerically on a massively parallel computer, a Cray T3E. On such a distributed memory machine it is crucial to minimize the inter-processor communication time, especially for applications running on hundreds of processors. Therefore it is computationally advisable to employ an explicit time stepping scheme which (in contrast to implicit or spectral methods) only operates on local data. This is because it can be parallelized in a straightforward manner using domain decomposition and it does not involve major data redistributions. Moreover it allows for an easy generalization of numerical methods originally developed for hydrodynamic conservation laws, e.g., second-order upwind schemes. The major disadvantage of explicit schemes is, of course, the strict stability limit they put on the time step. In practice, however, we can live with that restriction in situations where accuracy requirements lead to a time step which is of the same order as, or smaller than, the stability limit. We will show later in the paper that this is indeed the case for drift wave turbulence; nonlinearities set the time step and we are better off with a simpler explicit time stepping scheme. Next, we will give a detailed description of the employed numerical algorithm.

3.2. Explicit finite difference scheme on a phase space grid

3.2.1. Parallel canonical momentum method

The numerical scheme outlined in this section is a collection of methods borrowed from computational fluid dynamics (CFD) and applied to the current problem under investigation. To do this, however, a few critical issues must be addressed, most notably the treatment of the $\partial_t A_j$ term in Eqs. (1) and (3). In the past, two ways were found to get around the problems associated with this term: (1) one can replace $-\nabla \phi - \hat{\beta} \partial_t A_j$ by $E_\parallel$, but one then has to solve a fairly complicated nonlinear equation for $E_\parallel$ [6], or (2) one can use the canonical momentum formulation of the drift-kinetic equation in which the $\partial_t A_j$ term does not occur at all [8,21]. Unfortunately, the former approach runs into serious numerical problems, namely it develops a numerical instability which hampers efforts to penetrate the experimentally relevant regime of moderate plasma beta ($\mu_e \ll \beta_e \ll 1$). In the second approach, the parallel velocity $w_\parallel$ gets replaced by the parallel canonical momentum per unit mass, $p_\parallel = w_\parallel - eA_\parallel/m_e c$ (in unnormalized units). The main drawback here is that for electrons the $p_\parallel$ space grid has to be larger than a corresponding $w_\parallel$ space grid to account for the $A_\parallel$ fluctuations, especially at relatively high values of $\beta_e$, i.e. $\beta_e \gg \mu_e$. In contrast to this method we will work in $w_\parallel$ space, but deal with the $\partial_t A_j$ term by rewriting Eqs. (1)–(3) in the following way:

$$\frac{d}{dt} F = (v_x + \alpha_e w_\parallel B_x) \omega_T f_m - \alpha_e w_\parallel \nabla_\parallel^2 (f - f_m \phi) - \alpha_e w_\parallel \nabla_\parallel f,$$

$$\frac{d}{dt} U_\parallel = -\varepsilon^{-1} \nabla_\parallel^2 \phi,$$

$$\frac{d}{dt} \Omega = \nabla_\parallel^2 J_\parallel + \nabla_\parallel J_\parallel,$$

where

$$\langle F, U_\parallel, \Omega \rangle = \langle f - \alpha_e w_\parallel f_m \hat{\beta} A_\parallel, u_\parallel + \beta_e A_\parallel, \nabla^2_\parallel \phi \rangle.$$
\( \nabla_q = \partial_q \) and \( \nabla_{\parallel} = -\hat{\beta} \times \nabla A_{\parallel} \cdot \nabla \). This set of equations can be written in a more abstract way as

\[
d_q u' = F(u, A_{\parallel})
\]

(18)

where \( u' = (F, U_{\parallel}, \Omega) \), and \( F \) is the nonlinear functional (depending on \( u = (f, u_{\parallel}, \phi) \) and \( A_{\parallel} \)) defined by the right-hand sides (RHS) of Eqs. (14)–(16). The two sets of dependent variables, \( u \) and \( u' \), are equivalent to each other because the auxiliary equation

\[
[(1 + \mu_e) \beta_e/\mu_e - \nabla_{\parallel}^2] A_{\parallel} = U_{\parallel} - \int \alpha_e w_f F d^3w
\]

(19)

(which is derived readily from Eq. (4)) allows us to extract \( A_{\parallel} \) and then compute \( f \) and \( u_{\parallel} \) from \( F \) and \( U_{\parallel} \). Thus the numerical difficulty of treating the partial time derivatives of \( A_{\parallel} \) in Eqs. (1) and (3) is by-passed by updating \( F \) and \( U_{\parallel} \) instead of \( f \) and \( u_{\parallel} \). It is interesting to note that this procedure can be viewed as a different version of the above parallel canonical momentum formalism: \( U_{\parallel} \) and \( V_{\parallel} = \int \alpha_e w_f F d^3w \) are simply the (unnormalized) ion and electron parallel canonical momenta, \( u_{\parallel} + eA_{\parallel}/m_i c \) and \( v_{\parallel} - eA_{\parallel}/m_e c \) (where \( v_{\parallel} = \mu_0^{-1} \int w_{\parallel} f d^3w \)), normalized to \( \delta c_{\parallel} d R/L_n \). However, in this case \( U_{\parallel} \) and \( V_{\parallel} \) are dependent variables, not independent variables as in the “classical” parallel canonical momentum formalism.

3.2.2. Explicit time stepping scheme

Eq. (18) is solved numerically using the following two-step explicit time stepping scheme which is second-order accurate in time:

(1) In a first step, we neglect the nonlinear \( E \times B \) convection terms in Eq. (18) and employ a Lax-Wendroff scheme [22] by first computing predictor values

\[
u^* = u^n - (\delta t/2) F(u^n, A^n_{\parallel})
\]

(20)

at the intermediate time step \( t^* = (n + 1/2)\delta t \), and then using these values to obtain the new values

\[
u^{n+1} = u^n - \delta t F(u^*, A^*_{\parallel})
\]

(21)

in a second corrector step. Note that to obtain \( u \) and \( A_{\parallel} \) from \( u' \) one has to do a Poisson solve and a Helmholtz solve, i.e. two of each solves are required per time step.

(2) In a second step, the \( E \times B \) nonlinearities are computed according to a multidimensional second order upwind method by Colella [23]. The results are then added to the RHS of Eq. (21) in an operator-splitting manner. Let us remark here that we choose to treat these terms separately because the Colella scheme is perfectly suited for them. It avoids the occurrence of unphysical oscillations typical for many “standard” advection schemes (including the Lax-Wendroff scheme) by generalizing the concept of geometrical limiters by van Leer [24] to multi-dimensional problems.

The quantities \( f, u_{\parallel}, \phi \) and \( A_{\parallel} \) at the new time step \( t^{n+1} \) are thus computed from those at the old time step \( t^n \) in the following way:

- Compute \( u^* \) from \( u^n \) and \( A^n_{\parallel} \), Eq. (20).
- Poisson and Helmholtz solves yield \( u^* \) and \( A^*_{\parallel} \).
- Evaluate the RHS of Eq. (21).
- Add the \( E \times B \) contribution to obtain the final \( u^{n+1} \).
- Poisson and Helmholtz solves yield \( u^{n+1} \) and \( A^{n+1}_{\parallel} \).

3.2.3. Phase space discretization

The spatial simulation domain \( D \) is represented by the equidistant grid \((i \Delta x = L_x/2, j \Delta y = L_y/2, k \Delta z = \pi)\) with \( \Delta x = L_x/N_x, \Delta y = L_y/N_y, \Delta z = 2\pi/N_z, i = 1, \ldots, N_x, j = 1, \ldots, N_y, \) and \( k = 1, \ldots, N_z \). As was shown in Section 2, the basic equations can be reduced by one velocity space dimension. Thus instead of working with \( f(x, y, z, w_{\parallel}, \mu, t) \) we work with \( f^0(x, y, z, w_{\parallel}, t) \) and \( f^1(x, y, z, w_{\parallel}, t) \), saving a considerable amount of
computer resources. The grid spacing of the remaining velocity space coordinate \( w_k \) in the computational domain \([-L_{w||}, L_{w||}]\) is, in general, not chosen to be constant. As the variations of the distribution function (due to Landau resonance effects) are particularly strong for \( w_k\) \( 0 \), we use a finer grid within an inner region \( [L_{w||} - L_{w||}^i, N_{w||}^i] \) than in the rest of \( w_k \) space. Thus the grid points are given by \( l \Delta w_k = (L_{w||} - L_{w||}^i)/(N_{w||}^i - N_{w||}^i) \) for \( |l| \leq N_{w||}^i \) and \( l \Delta w_k = \pm L_{w||}^i \), with \( N_{w||}^i = (L_{w||} - L_{w||}^i)/(N_{w||}^i - N_{w||}^i) \) for \( N_{w||}^i < |l| \leq N_{w||}^i \). Using this grid, all spatial differential operators are finite differenced by centered second-order expressions. To compute moments of the electron distribution function one has to integrate over \( w_k \). Here we apply the trapezoidal rule [25] which is also a second-order method.

### 3.2.4. Field solves

To obtain \( \phi \) and \( A_k \) from \( F \) and \( U \), and \( \Omega \) both at the intermediate time step \( t^* \) and at the new time step \( t^{n+1} \), one has to solve Poisson and Helmholtz equations, \( \nabla^2 \phi = \Omega \) and Eq. (19), where the perpendicular Laplacian is given by

\[
\nabla^2 \phi = \frac{\partial^2 \phi}{\partial x^2} + 2\Delta z \frac{\partial^2 \phi}{\partial x \partial y} + (1 + \Delta z^2) \frac{\partial^2 \phi}{\partial y^2}.
\]

The use of periodic boundary conditions in the perpendicular \( x \) and \( y \) directions suggests the use of a Fourier solver to accomplish this. In order to stay consistent with the finite difference computations in real space, we choose the representation of the spatial discretization

\[
\frac{\partial^2 \phi}{\partial x^2} (x, y, z) \rightarrow \frac{\phi_{i+1,j,k} - 2\phi_{i,j,k} + \phi_{i-1,j,k}}{\Delta x^2},
\]

\[
\frac{\partial^2 \phi}{\partial y^2} (x, y, z) \rightarrow \frac{\phi_{i,j+1,k} - 2\phi_{i,j,k} + \phi_{i,j-1,k}}{\Delta y^2},
\]

\[
\frac{\partial^2 \phi}{\partial x \partial y} (x, y, z) \rightarrow \frac{\phi_{i+1,j+1,k} - \phi_{i+1,j-1,k} - \phi_{i-1,j+1,k} + \phi_{i-1,j-1,k}}{4\Delta x \Delta y}
\]

in Fourier space:

\[
\frac{\partial^2 \phi}{\partial x^2} (k_x, k_y, z) \rightarrow \frac{2}{\Delta x^2} (\cos(k_x \Delta x) - 1) \phi(k_x, k_y, z),
\]

\[
\frac{\partial^2 \phi}{\partial y^2} (k_x, k_y, z) \rightarrow \frac{2}{\Delta y^2} (\cos(k_y \Delta y) - 1) \phi(k_x, k_y, z),
\]

\[
\frac{\partial^2 \phi}{\partial x \partial y} (k_x, k_y, z) \rightarrow \frac{1}{\Delta x \Delta y} \sin(k_x \Delta x) \sin(k_y \Delta y) \phi(k_x, k_y, z),
\]

where \( (k_x, k_y) = (2\pi n_x/L_x, 2\pi n_y/L_y) \) with integers \( n_x \) and \( n_y \). Thus to do the Poisson and Helmholtz solves, one performs a two-dimensional Fourier transform, a multiplication in Fourier space and an inverse Fourier transform. As in the Poisson equation the amplitude of the \((n_x, n_y) = (0, 0)\) modes is not uniquely determined, it is set to zero.

### 3.2.5. Time step considerations

For reasons of numerical stability, the time step \( \Delta t \) in our explicit scheme is limited by the following constraints:

\[
\alpha_c w_{||} \Delta t < 1, \quad v_B \Delta t < 1, \quad \alpha_c w_{||} \beta A_{||} k_{\perp}^2 \Delta t < 1, \quad \omega_B \Delta t < 1.
\]

These correspond to parallel electron dynamics, perpendicular \( \mathbf{E} \times \mathbf{B} \) advection, magnetic field fluctuations, and linear waves (collisionless drift Alfvén waves and kinetic shear Alfvén waves), respectively. Assuming that in fully developed turbulence the normalized fluctuation amplitudes are of order unity, the first three of these conditions
translate into $\Delta t < O(\sqrt{\mu})$, $\Delta t < O(1)$, and $\Delta t < O(\sqrt{\mu}/\hat{\beta})$. The prefactors depend on the parameters of the computational grid. The most important conclusion that can be drawn from these relations is this: For tokamak core parameters, $\hat{\beta} \simeq \mu \sim 10^{-3}$, the time step is limited by the parallel electron motion. But in the edge of a tokamak where we have $\hat{\beta} \simeq \mu \sim 10$, the magnetic nonlinearities lead to a more stringent constraint. As we always have $\omega_r \lesssim 1$ for collisionless drift Alfvén waves (see Section 4.1), they put no extra limit on the time step. However, the inverse frequency of kinetic shear Alfvén waves becomes as small as $\sqrt{\mu k} \sim 1$ in the electrostatic limit [5]. This means that the time step is restricted by $(\sqrt{\mu k} \sim 1)$ which is inversely proportional to the perpendicular simulation box size. Therefore bigger boxes lead to a smaller time step. Fortunately, electromagnetic effects allow for larger time steps which are independent of the perpendicular box size. In the edge parameter regime, the time step is thus limited by a nonlinear effect, namely the motion of fast electrons along perturbed magnetic field lines. Accuracy and stability requirements basically mean the same thing in this case and we are better off with a simpler explicit numerical scheme.

3.3. Parallelization

3.3.1. Parallelization strategy

To solve the equations given in Section 2, we use a massively parallel computer, i.e. the 784 processor Cray T3E of the Computing Center at Garching. Single runs typically take 6 hours on at least 128 processors, achieving a performance of more than 50 MFlops per processor (including diagnostics and checkpointing). The use of fewer processors is impossible with respect to both CPU time and memory requirements, a feature which is characteristic of kinetic turbulence codes. Parallelization across the computational nodes of this distributed memory machine is done by three-dimensional domain decomposition in real space using the Cray message passing library. The subdomains contain additional border cells to allow for intermediate computation without interprocessor communication. This parallelization strategy is straightforward and leads to very good load balancing. Also, the computation-to-communication ratio is very high as long as the number of interior grid points is large compared to the number of border cells. To ensure this, the problem size and the number of used processors should be in a reasonable relation. Below, we will give specific numbers concerning this issue.

Whereas the general structure of the parallelized code is rather simple, there is one part of the algorithm that requires special attention. As we pointed out earlier, the Poisson and Helmholtz equations for the fields $\phi$ and $A_k$ are solved with Fast Fourier Transforms (FFTs). Whereas all other computations are completely local (i.e. they only involve data at neighboring grid points), these FFTs operate on entire lines of the domain in the $x$ and $y$ directions. As FFTs are known not to be perfectly scalable, we prefer to perform data redistributions in order to achieve optimum parallelism. Besides the primary decomposition into three-dimensional subdomains we therefore utilize secondary decompositions in the following way:

- Redistribute the data representing the RHS of the Poisson or Helmholtz equation such that each processor holds the same number of lines in the $x$ direction (parallelization over $y$ and $z$).
- Perform multiple 1D FFTs ($x \rightarrow k_x$) on each processor.
- Redistribute the data such that each processor now holds the same number of lines in the $y$ direction (parallelization over $k_x$ and $z$).
- Perform multiple 1D FFTs ($y \rightarrow k_y$) on each processor.
- Solve for $\phi$ or $A_k$.
- Perform multiple 1D inverse FFTs ($k_y \rightarrow y$) on each processor.
- Redistribute the data such that each processor holds the same number of lines in the $k_x$ direction.
- Perform multiple 1D inverse FFTs ($k_x \rightarrow x$) on each processor.
- Go back to the primary three-dimensional decomposition.

This procedure involves four data redistributions, two of which are merely matrix transpositions. Thus every FFT operates on local data only. For load balancing reasons, the number of grid points in all three spatial directions is taken to be a power of two.
Fig. 1. Parallel performance of our kinetic turbulence code measured in terms of the scaled speedup for a problem size of $16 \times 16 \times 8$ spatial grid points per processor. For comparison, the ideal speedup curve is shown as a dashed line.

3.3.2. Parallel performance

An adequate measure to quantify the performance of programs on massively parallel computers is the “scaled speedup” [26]. Here, in contrast to the more traditional definition of parallel speedup (with fixed problem size), the problem size per processor is fixed. The scaled-size paradigm leads to a better characterization of parallelism for systems with very large numbers of processors. This is because on distributed memory machines like the Cray T3E, the fixed-size speedup line is limited at both ends by memory constraints: a computational domain of realistic size can be distributed neither on too few nor on too many processors – either too much total memory is needed, or there are less grid points than processors. However, in practice the problem size generally tends to scale with the available memory and computing power. This is certainly true for plasma turbulence applications like the one considered in this paper.

If $T(N, P)$ denotes the run time of a parallel program for a problem of size $N$ on $P$ processors, we can define the scaled speedup as $PT(N, 1)/T(PN, P)$ whereas the fixed-size speedup is given by $T(N, 1)/T(N, P)$. Thus a scaled speedup of $P$ means that $P$ processors can solve a problem of $P$ times the size in the same amount of time, and a fixed-size speedup of $P$ means that $P$ processors can solve the same problem $P$ times as fast. The measured parallel performance of our kinetic turbulence code is very close to ideal as shown in Fig. 1 for a problem size of $16 \times 16 \times 8$ spatial grid points per processor.

Another very interesting question to address is this: How much time is spent doing what? For the above problem size per processor one gets roughly the following answer. Only about 5% of the time is used for the Poisson and Helmholtz solves (including all the necessary data redistributions), and all the other operations on three-dimensional quantities are virtually negligible compared to those on five-dimensional quantities. The boundary exchanges make up only less than 3% of the total run time. Most of the time, some 90%, is spent (to approximately equal amount) on the Lax-Wendroff and Colella schemes.

4. Numerical tests and results

4.1. Linear runs: Collisionless drift Alfvén waves

Electron Landau damping is known to destabilize collisionless drift waves in an unsheared magnetic field. However, the linear growth rates can be reduced significantly by electromagnetic (“finite-beta”) effects. In this
section we will derive the linear dispersion relation of these collisionless drift Alfvén waves and use it to benchmark the code in the linear regime.

In the absence of magnetic shear we search for solutions of the linearized equations in which all perturbed quantities are proportional to \(\exp(ikx - i\omega t)\). Eq. (1) thus becomes

\[
f = f_m\phi + (\omega r k_y - \omega) f_m(\phi - \alpha_e w_\parallel \hat{A}_\parallel),
\]

(24)
describing the linear response of the electron distribution function \(f\) to small perturbations in the potentials \(\phi\) and \(A_\parallel\). From Eq. (24) we can derive the expression

\[
v_\parallel = \alpha_e [(\sigma - \sigma_\star)(1 + \sigma_\star Z(\sigma)) - \eta_e \sigma \sigma_\star Y(\sigma)](\phi - \alpha_e \sigma \hat{A}_\parallel)
\]

(25)
for the parallel electron velocity \(v_\parallel = \int \alpha_e w_\parallel f \, d^3w\) with the plasma dispersion function \(Z(\sigma)\) [27], \(Y(\sigma) = \sigma + (\sigma_\star^2 - 1/2)Z(\sigma)\). \(\sigma_\parallel = \omega/\alpha_e k_y\) and \(\sigma_\star = k_y/\alpha_e k_y\). Neglecting the parallel ion dynamics – which is not critical for drift Alfvén wave physics – we get the additional equations \(\omega k_\perp^2 \phi = k_\parallel J_\parallel\) and \(J_\parallel = -v_\parallel = k_\perp^2 A_\parallel\) which together with Eq. (25) yield the linear dispersion relation

\[
k_\perp^2 \sigma + [(\sigma - \sigma_\star) - \eta_e \sigma \sigma_\star Y(\sigma)](1 + \sigma_\star Z(\sigma))(1 - 2\sigma_\star^2 \hat{\mu} / \hat{\mu}) = 0.
\]

(26)
Because it involves the special function \(Z\), this dispersion relation in general needs to be treated numerically. Analytical solutions only exist in the Landau limit, \(\max(\hat{\beta}, \hat{\mu}) \ll 1\), and in the Alfvén limit, \(\hat{\beta} \gg \max(\hat{\mu}, 1)\), where, respectively, we get

\[
\omega = \frac{k_y}{1 + k_\perp^2}, \quad \gamma = \omega_r^2 \left(\frac{k_\perp^2}{1 + k_\perp^2} - \frac{\eta_e}{2}\right) \left(\frac{\pi \hat{\mu}}{2k_y}\right)^{1/2}
\]

(27)
and

\[
\omega = \frac{k_\perp^2}{\hat{\beta}}, \quad \gamma = \omega_r^2 \left(\frac{k_\perp^2}{k_y}\right) \left(1 - \frac{\eta_e}{2}\right) \left(\frac{\pi \hat{\mu}}{8\hat{\beta}}\right)^{1/2}
\]

(28)
for \(\omega = \omega_r + i\gamma\). However, as we typically have \(\hat{\beta} \lesssim \hat{\mu} \sim 10\) in a tokamak H-mode edge, we use the Newton method to solve Eq. (26) numerically for complex values of \(\sigma\) (and hence \(\omega\)) given real values of \(\mathbf{k}\), \(\hat{\mu}\), \(\hat{\beta}\), and \(\eta_e\). Usually, this procedure takes only a few iterations to converge. The solution with the largest linear growth rate \(\gamma\) is the collisionless drift Alfvén wave.

We can then check the accuracy of our turbulence code by comparing the theoretical values \(\gamma_{th}\) with the ones computed from linear simulations choosing an initial state of the form

\[
f^N(x, y, z, w_\parallel, t = 0) = \cos(k_x x) \cos(k_y y) \cos(k_z z) f_m^N(w_\parallel)
\]

(29)
and setting \(\phi\) and \(A_\parallel\) to zero. After a short transient phase, the drift Alfvén wave establishes its linear mode structure and grows exponentially as can be seen in Fig. 2. The corresponding growth rate is computed as \(\gamma_{sim} = d_t \ln \phi_{rms}\) where \(\phi_{rms} = \langle \phi^2 \rangle^{1/2}\) and \(\langle \ldots \rangle\) denotes spatial averaging over the entire simulation domain. For the test cases presented in Fig. 3 we chose \(k_x = 0\), \(k_y = 0.5\), \(k_z = 1\), \(\mu = 10\), and \(\eta_e = 0\). The remaining simulation parameters were \(L_x = 64\), \(L_{w_\parallel} = 3\), \(L_y = 1\), \(N_x = 64\), \(N_y = 32\), \(N_{w_\parallel} = 40\), and \(N_\parallel = 20\). (Note that due to \(k_x = 0\) the values of \(L_x\) and \(N_x\) are irrelevant.) The time step was set to \(\Delta t = 0.1\), and the total simulation time was up to \(T = 100\). As can be inferred from Fig. 3, the linear growth rates derived from theory and simulation match very well. This clearly demonstrates that the code captures the linear physics of collisionless drift Alfvén waves in an unsheared slab. Satisfied with these linear tests, we now proceed with nonlinear results obtained by using the entire algorithm as described above.
Fig. 2. Root mean square of the electrostatic potential as a function of time. After an initial transient phase the drift Alfvén wave grows exponentially. The corresponding linear growth rate is computed as $\gamma_{\text{lin}} = dt \ln \phi_{\text{rms}}$.

Fig. 3. Linear growth rates of collisionless drift Alfvén waves in an unsheared magnetic field for different values of $\hat{\beta}$. The simulation results are plotted as open squares; the theoretical curve is shown as a solid line for comparison.

4.2. Nonlinear runs: Convergence studies

Various nonlinear results for fully developed collisionless drift Alfvén turbulence have been published elsewhere [4]. Among them are amplitude and phase distributions, amplitude correlations, energy spectra, and transport scalings. In this section we will study the numerical convergence properties with respect to phase space resolution. Of course, it is a necessary (but not sufficient) test for the validity of the obtained simulation results, if the purely numerical parameters characterizing the phase space resolution do have a significant impact on these results or not. However, a word of caution may be in place here. In principle, one should do such studies for every single set of physical parameters because the convergence properties might (and usually will) depend on them. E.g., in our case the necessary perpendicular grid spacing depends on the ratio of the collisionless skin depth $\delta_s$ to $\rho_s$ as will be explained below. Because such extensive test runs are usually not feasible, one has do to the next best thing, i.e. take a set of run parameters which are “typical” for a certain region in parameter space. We will adopt the latter approach in the following, monitoring the system’s behaviour through the electrostatic component of the radial heat flux,

$$Q = \left( \int w^2 f \, d^3w \, v_z \right).$$

(30)

The radial transport due to parallel motion along perturbed magnetic field lines is negligibly small by comparison [3,4,28]. As physical parameters we take $\hat{\mu} = \hat{\beta} = 10$, $\hat{\epsilon} = 3600$, $\eta_e = 0$, and $\hat{s} = 3/\pi$; the computational box in phase space is characterized by $L_x = 32$, $L_y = 64$, $L_{w1} = 3$, and $L_{w1}' = 1$; the time step was $\Delta t = 0.01$. These values of the physical parameters are typical for drift Alfvén turbulence in a tokamak edge and are similar to those used in a previous paper [4]. The results for seven runs with varying grid parameters $N_x$, $N_y$, $N_z$, $N_{w1}$, and $N_{w1}'$ are shown in Table 1. As can be inferred from runs 1–5, $Q$ is converged for $N_x = 32$, $N_y = 64$, and $N_z = 16$. That means if the spatial resolution is at least one grid point per $\rho_s$ in the $x$ and $y$ directions and 16 points per connection length in the $z$ direction, the results do not change significantly. This is in accordance with the $\phi$ and $\Omega$ spectra shown in Fig. 4 and [3]. Due to $\Omega \propto k_y^2 \phi$ the $k_y$ spectrum of the dependent variable $\Omega$ is much flatter than that of $\phi$. It falls off only at $k_y \approx 1$, calling for a perpendicular grid spacing of the order of $\rho_s$. Note that a finite $\hat{\beta}$ introduces a new intrinsic length scale in addition to $\rho_s$, i.e. the collisionless skin depth.
Table 1

Numerical convergence tests: Electrostatic heat flux $Q$ as a function of grid parameters $N_x$, $N_y$, $N_z$, $N_{w||}$, and $N_{w\perp}$

| Run | $N_x$ | $N_y$ | $N_z$ | $N_{w||}$ | $N_{w\perp}$ | $Q$ |
|-----|-------|-------|-------|-----------|-------------|-----|
| 1   | 16    | 32    | 16    | 40        | 20          | 0.28|
| 2   | 32    | 64    | 16    | 40        | 20          | 0.83|
| 3   | 64    | 128   | 16    | 40        | 20          | 0.87|
| 4   | 32    | 64    | 8     | 40        | 20          | 2.3 |
| 5   | 32    | 64    | 32    | 40        | 20          | 0.80|
| 6   | 32    | 64    | 16    | 20        | 10          | 0.98|
| 7   | 32    | 64    | 16    | 80        | 40          | 0.84|

Fig. 4. Spectra for the mean squared fluctuations of $\phi$ (solid line) and $\Omega = \nabla^2_x \phi$ (dashed line) in $k_y$ space. Note that compared to the $\phi$ curve the $\Omega$ curve drops only very slowly with increasing $k_y$. This behavior calls for a perpendicular grid spacing of the order of $s$.

Fig. 5. Typical snapshot of $f^{0}(w_{||})$ at a fixed point in space. The dashed line shows $f_{m\phi}$ for comparison.

$$\delta_k = c/\omega_{pe} = \rho_s (\hat{\mu}/\hat{\beta})^{1/2}$$ where $\omega_{pe}$ is the electron plasma frequency. So for $\hat{\beta} > \hat{\mu}$, $\delta_k$ becomes the smallest perpendicular length scale one has to resolve – but for typical tokamak H-mode edge parameters, $\hat{\beta} \lesssim \hat{\mu} \sim 10$, it is sufficient to choose $\rho_s$ as the perpendicular grid spacing.

One main difference between Vlasov and particle simulations is the presence of a velocity space grid in the former case. Therefore it is especially interesting to find out how many points in velocity space are necessary to achieve convergence in a nonlinear run. As was explained in Section 3.2.3, we use a nonequidistant grid in $w_{||}$ space to account for the fact that the variations of the distribution function (due to Landau resonance effects) are particularly strong for $w_{||} \approx 0$. Thus it is computationally more efficient to use a smaller grid spacing within an inner region around $w_{||} = 0$ than in the rest of $w_{||}$ space. A good example for the special role played by the inner region is shown in Fig. 5. It shows a typical snapshot of $f^{0}(w_{||})$ at a fixed point in space. The dashed line shows $f_{m\phi}$ for comparison; deviations from that curve indicate nonadiabatic electron dynamics. One observes a sharp
peak at \( w_j \approx 0 \), but otherwise \( f^0(w_j) \) is fairly smooth. Figures like that strongly suggest the use of grids with better resolution in the center than at the fringes.

The results of runs 2, 6, and 7 listed in Table 1 indicate that for the parameters used here, \( Q \) is converged for \( N_{w_j} = 40 \) and \( N_{i w_j} = 20 \). Even taking only half the number of velocity space points does not change the result dramatically (by some 20%). These numbers can now be used to compare the efficiency of Vlasov and particle simulations. Very roughly, one can say that if the number of variables (not particles!) per grid cell in a particle simulation and the number of velocity space points in a Vlasov simulation are comparable, so are the efficiencies of the methods (assuming that the number of operations per variable are approximately the same). In the particle simulations referred to in Section 3.1, some 50 variables per spatial grid cell needed to be evolved. Therefore we can say that both approaches need about the same amount of numerical effort. This finding somewhat contradicts the “conventional wisdom” that particle simulations are in general more efficient for three-dimensional studies of plasma turbulence. Therefore it opens the door for Vlasov simulations to supplement particle simulations in the area of core plasma turbulence, driven, e.g., by ion temperature gradients [29]. Work to achieve this is underway and will be reported in the near future.

4.3. Nonlinear runs: Comparison with other codes

Ideally, complex nonlinear codes are benchmarked against each other to rule out algorithmic or coding errors as much as possible. However, so far there does not exist a second code that solves the exact same set of equations. Faced with this situation, the best we can do is to compare with an existing Landau-fluid model for drift Alfvén turbulence [3]. This particular fluid model has been extended into the collisionless regime to incorporate kinetic effects such as Landau damping and is supposed to serve as a less costly substitute for fully kinetic models like the one under consideration here. Since systematic derivations of such models are only possible in very idealized situations (e.g., no magnetic shear or nonlinear effects), one cannot expect perfect agreement for fully developed turbulence in inhomogeneous magnetic fields, however. In other words: Such a comparison is as much a test for the Landau-fluid model as for the kinetic model. Nevertheless, it shows at least if the results are in the right ballpark or not.

Fig. 6 shows a \( \hat{\beta} \) scaling of the electrostatic component of the radial particle transport,

\[
\Gamma = \left\{ \int f \, \mathrm{d}^3 w \, v_x \right\}.
\]
as obtained by our kinetic and the companion Landau-fluid model. The simulation parameters were the same as above except for $L_x = 64$, $L_y = 256$, $N_x = 64$, and $N_y = 256$. The transport levels are observed to lie within 50% of each other over the whole range of $\dot{\beta}$. The problem that the trends appear to be different can be traced back to two deficiencies of the fluid model. First, it uses an isotropic temperature instead of separate parallel and perpendicular temperatures which are decoupled in the collisionless regime. Second, the standard fluid closure schemes [30,31] involve $k$ dependent damping terms which are only approximated in [3] due to the nonlinearity of the parallel derivative, $\nabla_j$. Given the finding that better closure schemes lead to a weaker $\dot{\beta}$ dependence of the particle transport [4], we can expect the fluid model curve to become flatter with these refinements incorporated. But even without them, we can conclude that the kinetic and the Landau-fluid model agree fairly well both qualitatively and quantitatively.

5. Summary

We presented a nonlinear Vlasov code to treat electromagnetic drift-wave turbulence in an inhomogeneous collisionless plasma confined by a strong external magnetic field. In contrast to the more conventional particle approach, Vlasov simulations exhibit no numerical noise despite being similarly efficient. The basic equations describing the dynamics of drift-kinetic electrons and fluid ions are rewritten in terms of parallel canonical momenta in order to by-pass certain numerical problems. The algorithm basically consists of two parts, a multidimensional second-order upwind scheme for the nonlinear $E \times B$ convection terms and a predictor-corrector scheme for all the other terms (which are mostly linear). The code is parallelized via spatial domain decomposition and runs on a massively parallel computer, i.e. the 784 processor Cray T3E of the Computing Center at Garching. Single runs typically take 6 hours on at least 128 processors, achieving a performance of more than 50 MFlops per processor. The parallel speedup is almost linear all the way up to 512 processors. Various numerical tests both in the linear and nonlinear regime have shown that the code yields reliable results. Numerical convergence issues also have been addressed. Extensions of this code to treat core plasma turbulence (e.g., driven by ion temperature gradients) are underway and will be reported in the near future.

Acknowledgement

I would like to thank Bruce D. Scott for numerous useful discussions.

References