JOURNAL OF COMPUTATIONAL PHYSICS 119, 283–294 (1995)

δf Algorithm

RICHARD E. DENTON* AND M. KOTSCHENREUTHER

Institute for Fusion Studies, University of Texas at Austin, Austin, Texas 78712

Received November 8, 1993; revised January 12, 1995

The δf Algorithm is a low noise particle code algorithm. The perturbation of the distribution function (δf) away from a large equilibrium is evolved rather than the total distribution function. "Particles" in the code are actually Lagrangian markers at which the value of the distribution function is known. The magnitude of the numerical noise is characteristic of the size of the perturbation rather than the equilibrium and scales roughly as the inverse of the number of particles. In this paper, the algorithm is described, and conserved energies are derived for linear and nonlinear sets of equations. Two different forms of the energy principle test separately adequate resolution in time and space and adequacy of the number of simulation particles. A semi-implicit time step method is described which allows violation of the Courant condition. Low noise capabilities of a linear code using the algorithm are demonstrated.

INTRODUCTION

Transport in controlled thermonuclear devices is thought to be caused by small scale turbulence from microinstabilities. Most of these low frequency microinstabilities can be described by the ion gyrokinetic equation [1–4]. For this reason, a number of gyrokinetic particle codes have been constructed in order to simulate this turbulence [4–6]. However, noise problems inherent in conventional particle codes make simulations of fusion plasmas with realistic parameters impractical.

The δf Algorithm was first conceptualized by Tajima and Perkins [7]. Mikic and Morse [8] implemented a linearized version of the algorithm for a Vlasov system. Dimits and Lee [9] first successfully implemented a nonlinear gyrokinetic δf Algorithm, which they called a "partially linearized algorithm." Independently, Kotschenreuther [10] developed the δf Algorithm described here and stressed its low noise advantages. With the δf Algorithm, particle noise is greatly reduced (and therefore the particle requirements) for gyrokinetic simulations if the turbulent density fluctuations are much smaller than the background density, which is the case throughout most of the plasma volume in confinement devices. Using this algorithm, Kotschenreuther *et al.* employed particle simulations for realistic experimental parameters for the first time for low frequency microinstabilities thought to be responsible for tokamak transport [11]. The δf Algorithm has been further developed by others. Parker and Lee [12] use of somewhat different algorithm which evolves a weight rather than δf itself; they keep higher-order terms than those maintained here. The generalized weighting scheme of Hu and Krommes [13] is applicable to compressible and non-Hamiltonian systems.

This paper represents the first detailed description of the δf Algorithm as implemented by Kotschenreuther. Kotschenreuther's algorithm introduces the concept of "markers" which has been utilized in the generalized δf weighting scheme of Hu and Krommes [13]. In addition to low noise, this algorithm has the advantages of an extremely simple set of equations, simplified energy conservation relations, and a semi-implicit method which can relax time step restrictions such as the Courant condition for the electrons ($\Delta t < 1/(k_\parallel v_{\parallel e})$). In this paper, we describe an implementation of the δf Algorithm in an electrostatic lowest order gyrokinetic initial value particle code. We examine the scaling of noise with the number of simulation particles, demonstrate the use of the semi-implicit algorithm, and show how different energy terms can provide information on resolution in time and space and sufficiency of the number of simulation particles.

In a conventional particle code, the density of plasma is related to the density of simulation particles. In the δf Algorithm, however, each "particle" is a marker at which the value of the distribution function is known. The δf code is in this sense similar to a Vlasov fluid code with a Lagrangian grid. The value of the distribution function for each marker is evolved using the method of characteristics. The position of each marker is the tip of a characteristic of the distribution function, that is, the latest position of one of the natural paths for the solution of the equations. The markers are like particles in that they move in space with the same equations of motion. However, rather than representing single particles, they represent evolving values of the distribution function.

In the δf Algorithm, the total distribution function f is decomposed into two parts, a background distribution f^0, and the remaining component, δf. The essential feature of the δf Algorithm is that only δf is represented by markers. Typically, f^0 will be represented by an analytical formula (such as a Maxwellian with density and temperature gradients as is used in this

* Presently at Dartmouth College, Hanover, New Hampshire 03755.

0021-9991/95 $12.00

paper). If the density fluctuation $\delta\rho$ ($=\int d\mathbf{v}\ \delta f$) is smaller than the background density ρ^0 ($\int d\mathbf{v}\ f^0$), then the statistical noise for a δf simulation with N markers will be significantly lower than that of a conventional particle code using the same number of particles. This is because in a conventional particle simulation, there is numerical noise associated with the representation of f^0 by particles; and when $\delta\rho \ll \rho^0$, most of the noise comes from f_0. In the δf Algorithm, the numerical noise comes only from the representation of the smaller δf by the markers.

The algorithm has much in common with linear Vlasov codes [14] in that only a small part of the distribution function is evolved; however, in the δf Algorithm the equations may be nonlinear. As long as $\delta\rho$ is less than ρ^0, the method will be advantageous even when the plasma is nonlinearly saturated. In tokamak experiments, $\delta\rho/\rho^0$ is typically of the order of 10^{-2} in most of the confinement volume [15]. Thus for realistic parameters, the δf Algorithm will have a tremendous advantage over conventional techniques.

We will apply the δf Algorithm to the electrostatic gyrokinetic equations. These equations are derived from the Vlasov equation for the case of a plasma in a strong magnetic field by averaging over the gyromotion of the particle orbits and maintaining only the lowest order terms in the expansion parameter ρ_i/L_n, where ρ_i is the ion gyroradius and L_n is the equilibrium density scale length. We will present the nonlinear equations and then the linearized equations. The results in this paper will be from a linearized code; results from a nonlinear code will be presented elsewhere. The most efficient way to write a linear code would be to use particles which have the shape of a Fourier mode in the symmetry direction, effectively eliminating this dimension from the problem. In the linear code described here, we maintain the symmetry direction in order to replicate the numerical noise properties of a two-dimensional code.

In practice, conventional particle code turbulence simulations use unrealistically large background gradients to drive strongly unstable modes so that the saturation amplitude is above the noise. For example, simulations of η_i turbulence usually use large values of η_i ($\eta_i = L_n/L_{T_i}$ where L_{T_i} is the ion temperature scale length) and strong equilibrium gradients ($\rho_i/L_n \sim 1/40$), whereas realistic η_i values are usually not very far from marginal stability and $\rho_i/L_n \sim 10^{-3}$. Also, in conventional simulations the equilibrium temperature can vary significantly over the mode width, introducing physical effects not present for experimental device parameters. The results of such simulations are instructive, but the practice of using parameters which are unrealistic for fusion conditions reduces their relevance for description of fusion plasmas or prediction of future plasma conditions. However, using the δf Algorithm, it is possible to simulate turbulent states of fusion plasmas in three-dimensional geometry using realistic parameters.

The paper is organized as follows: The equations and a description of the algorithm are given in Section 1, numerical results are described in Section 2, and conclusions follow in Section 3.

1. EQUATIONS AND DESCRIPTION OF THE ALGORITHM

The electrostatic gyrokinetic equation is derived from the Vlasov equation by expanding in ρ_i/L_n, averaging over the gyroorbits, and keeping only the lowest order terms. The derivation assumes $\rho_i/L_n \sim \omega/\Omega_{c_i} \sim \delta f/f^0 \sim e\phi/T_i$, where ω is the amplitude of the complex frequency of the mode; $\Omega_{c_i} = eB_0/m_i c$ is the ion cyclotron frequency; e is the electron charge; B_0 is the large ambient magnetic field; m_i is the ionic mass; c is the speed of light; ϕ is the electrostatic scalar potential; T_i is the temperature of the background Maxwellian of the ions. With these assumptions, one obtains to lowest order in slab geometry [3, 4]

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{v}_{E_\alpha}\cdot\nabla f_\alpha + v_{\parallel_\alpha}\nabla_\parallel f_\alpha + \frac{q_\alpha}{m_\alpha}\langle E_\parallel\rangle_\alpha \frac{\partial f_\alpha^0}{\partial v_{\parallel_\alpha}} = 0, \quad (1.1)$$

where

$$\langle \mathbf{E}\rangle_\alpha = -\nabla\langle\phi\rangle_\alpha$$

and

$$\mathbf{v}_{E_\alpha} = \frac{c\langle \mathbf{E}\rangle_\alpha \times \hat{z}}{B_0}.$$

Here, f_α is the gyrophase averaged distribution function for the α species, where α = i/e for ions/electrons. The distribution function f_α is related to the gyrophase averaged nonadiabatic distribution function, h_α, by $f_\alpha = h_\alpha - (q_\alpha\langle\phi\rangle_\alpha/T_\alpha)f_\alpha^M$, where f_α^M is a Maxwellian. Like h_α, f_α is a gyroaveraged quantity and is a function of the guiding center coordinates. The component of the α species particle velocity in the direction of the magnetic field is $v_{\parallel_\alpha}$ while $v_{\perp_\alpha}$ is the magnitude of the perpendicular velocity; m_α is the mass of a particle of the α species; $\mathbf{E}$ is the electric field; the parallel component of $\mathbf{E}$ is $E_\parallel = \hat{b}\cdot\mathbf{E}$; $\hat{b}$ is a unit vector in the direction of the total magnetic field $\mathbf{B}$ while $\hat{z}$ is the direction of the large component B_0; and $\mathbf{v}_{E_\alpha}$ is the $\mathbf{E}\times\mathbf{B}$ drift velocity.

In Eq. (1.1), $\langle Q\rangle_\alpha$ indicates that the quantity Q is gyroaveraged. In Fourier space, with $Q_\mathbf{k} = \int (d\mathbf{r}/2\pi)e^{-i\mathbf{k}\cdot\mathbf{r}}\,Q(\mathbf{r})$, where $\mathbf{r}$ is the gyrocenter position, the gyroaverage is easily evaluated [1]

$$\langle Q_\mathbf{k}\rangle_\alpha = J_0\left(\frac{v_{\perp_\alpha}}{\Omega_{c_\alpha}}k_\perp\right)Q_\mathbf{k}, \quad (1.2)$$

where J_0 is the zeroth order cylindrical Bessel function and $k_\perp$ is the component of the wave vector $\mathbf{k}$ perpendicular to $\mathbf{B}$. The

α subscript on the gyroaverage brackets $\langle\ \rangle_\alpha$ indicates that the gyroaveraged quantity is viewed differently by the ions and electrons.

We will be using two-dimensional sheared slab geometry in which all plasma quantities are independent of z. The $\hat{x}$-direction will be the direction of the equilibrium inhomogeneities in density and temperature. There is a large ambient magnetic field B_0 in the $\hat{z}$-direction. A sheared component varies in the $\hat{x}$-direction with scale length L_s but points in the $\hat{y}$-direction so that the total magnetic field $\mathbf{B} = B_0(\hat{z} + (x/L_s)\,\hat{y})$. The sheared component of $\mathbf{B}$ enters the problem only in the evaluation of $\nabla_\parallel = \hat{b}\cdot\nabla \Rightarrow (x/L_s)(\partial/\partial y)$.

We note that these equations are easily generalized to three dimensions. In that case it is cheaper computationally to evaluate the gyroaverage by averaging over several points in real space [5]. Although the present method (using Bessel functions) is not ultimately the cheapest computational method, it is sufficient for studying the energy conservation, noise properties, and semi-implicit scheme, which is our purpose here.

We now divide f_α into two parts, the background distribution f_α^0 and the remaining component δf_α, so that $f_\alpha = f_\alpha^0 + \delta f_\alpha$. We use a Maxwellian distribution for f_α^0,

$$f_\alpha^0 = f_\alpha^M = \frac{\rho^0}{\pi^{3/2}\, v_{\mathrm{th}_\alpha}^3}\, e^{(-m_\alpha v_\alpha^2/(2T_\alpha))}, \tag{1.3}$$

where ρ^0 is the background density, and T_α and $v_{\mathrm{th}_\alpha} = (2T_\alpha/m_\alpha)^{1/2}$ are the temperature and thermal velocity of the α species. (Quasi-neutrality has been assumed so that the ion and electron densities are equal.) We assume that f_α^0 varies with scale lengths L_n and L_{T_α} but that L_n and L_{T_α} are large compared to the simulation region so that the $\hat{x}$-direction variation appears only in the representation of $\partial/\partial x$, where $\partial/\partial x$ acts on f_α^0. (The density ρ^0 will not appear in the normalized form for f_α^0 which follows in Eq. (1.4).) Thus

$$\frac{\partial f_\alpha^0}{\partial x} \Rightarrow \frac{1}{L_n}\left[1 + \eta_\alpha\left(\frac{v_\alpha^2}{v_{\mathrm{th}_\alpha}^2} - \frac{3}{2}\right)\right] f_\alpha^0,$$

where $v_\alpha^2 = v_{\parallel_\alpha}^2 + v_{\perp_\alpha}^2$ and $\eta_\alpha = L_n/L_{T_\alpha}$. We stress again this key point of the δf Algorithm: since we are representing f_α^0 by an analytical formula, there will be no noise associated with this large part of the distribution function. We note that it is in general possible for f_α^0 to be time-dependent [16] and spatial-dependent. Although the analytical form for f_α^0 is especially convenient, f_α^0 might be represented in a numerical form (that is, by data). The value of f_α^0 might even be adjusted during the simulation to keep δf_α small [17].

We now introduce the normalized quantities, X, Y, $V_{\parallel_\alpha}$, $V_{\perp_\alpha}$, F_α^0, δF_α, Φ, and τ. The corresponding unnormalized quantities, x, y, $v_{\parallel_\alpha}$, $v_{\perp_\alpha}$, f_α^0, δf_α, ϕ, and t are related to these in the following way: $x = (\rho_i)X$, $y = (k_{y_0}^{-1})Y$, $v_{\parallel_\alpha} = (v_{\mathrm{th}_\alpha})V_{\parallel_\alpha}$, $v_{\perp_\alpha} = (v_{\mathrm{th}_\alpha})V_{\perp_\alpha}$, $f_\alpha^0 = (\rho^0 v_{\mathrm{th}_\alpha}^{-3})F_\alpha^0$, $\delta f_\alpha = ((\rho_i/L_n)\,\rho^0 v_{\mathrm{th}_\alpha}^{-3})\delta F_\alpha$, $\phi = ((\rho_i/L_n)(T_i/e))\Phi$, and $t = (\omega_{*_i}^{-1})\tau$. The ion gyroradius $\rho_i = v_{\mathrm{th}_i}/\Omega_{c_i}$; k_{y_0} is the component of $\mathbf{k}$ in the symmetry direction $\hat{Y}$ corresponding to the longest wavelength in the system; and the ion drift frequency

$$\omega_{*_i} = \frac{ck_{y_0}T_i}{eB_0 L_n}.$$

In addition, we define a normalized gyrocenter position, $\mathbf{R} = X\hat{X} + Y\hat{Y}$, a normalized gradient

$$\overline{\nabla} = \hat{X}\frac{\partial}{\partial X} + \hat{Y}\frac{\partial}{\partial Y},$$

and a normalized wavevector $\mathbf{K} = (\rho_i)k_x\hat{X} + (k_{y_0}^{-1})k_y\hat{Y}$. (Note that these three vector quantities are normalized differently in the $\hat{X}$- and $\hat{Y}$-directions.) Upon substituting $f_\alpha = f_\alpha^0 + \delta f_\alpha$ into Eq. (1.1) and using these normalized quantities, we get to lowest order in ρ_i/L_n

$$\frac{d}{d\tau}\delta F_\alpha = S_\alpha \tag{1.4}$$

with

$$\frac{d}{d\tau} = \frac{\partial}{\partial\tau} + \hat{z}\times\overline{\nabla}\langle\Phi\rangle_\alpha\cdot\overline{\nabla} + V_{\parallel_\alpha}k_\alpha X\frac{\partial}{\partial Y}$$

and

$$S_\alpha = F_\alpha^0\left([\eta]_\alpha - T_\alpha^* V_{\parallel_\alpha}k_\alpha X\right)\frac{\partial\langle\Phi\rangle_\alpha}{\partial Y}$$

and where

$$F_\alpha^0 = \frac{1}{\pi^{3/2}}\, e^{-V_\alpha^2}.$$

We have introduced the following quantities: $T_\alpha^* = 1$ for ions, but $-T_i/T_e$ for electrons; the shear parameter $k_\alpha = 2L_n/L_s$ for ions, and $(v_{\mathrm{th}_e}/v_{\mathrm{th}_i})(2L_n/L_s)$ for electrons; $[\eta]_\alpha = 1 + \eta_\alpha(V_\alpha^2 - \frac{3}{2})$. The quantity $K_{\perp_\alpha}$ is defined so that $|\mathbf{k}| = (\rho_i^{-1})\,K_{\perp_i}$ and $K_{\perp_e} = (T_e m_e/(T_i m_i))^{1/2}K_{\perp i}$. With this definition of $K_{\perp_\alpha}$, the gyroaverage is evaluated with [1]

$$\langle Q_{\mathbf{K}}\rangle_\alpha = J_0(V_{\perp_\alpha}K_{\perp_\alpha})\,Q_{\mathbf{K}}. \tag{1.5}$$

If we take the mass ratio, m_i/m_e, to be realistic, then the remaining parameters are η_α, and k_α, or η_α, k_i, and T_i/T_e. Since these parameters are all roughly of order unity, the final nonlinear saturated state in a nonlinear run will have δF_α roughly of order unity. We note also that the nonlinear parallel

acceleration term, $q_\alpha \langle E \rangle_\alpha \cdot (\partial/\partial \mathbf{v}) \delta f_\alpha$ is one order higher in ρ_i/L_n (as can be seen from the normalization constants for δF_α and F^0_α). Since $\rho_i/L_n \sim 10^{-3}$ in the cases of interest to us, this term is utterly negligible. Nonlinear effects which are associated with this term in a Vlasov plasma, such as detrapping of particles, occur through the $\mathbf{E} \times \mathbf{B}$ motion rather than through parallel acceleration.

Instead of using the fluid form of Eq. (1.4) to solve for the evolution of δF_α, we will solve for the evolution of δF_α along a number of paths in phase space using the method of characteristics. Our "particles" are actually markers at the tip of the characteristics; they are the phase space positions at which we know the value of δF_α. Note that $d/d\tau$ in Eq. (1.4) represents motion in space but no acceleration. The parallel acceleration term acting on F^0_α, $q_\alpha \langle E \rangle_\alpha \cdot (\partial/\partial v) F^0_\alpha$ is of the same order as the other terms and is included in the source term S_α in Eq. (4); however, this term represents the combination of a high order (and thus very small) acceleration and the very large F^0_α. The lowest order characteristics have no variation in velocity space.

Thus we will initialize X_j, Y_j, $V_{\parallel_j}$, $V_{\perp_j}$, F^0_j, and δF_j for a set of markers with index j and evolve them according to the following equations. The index j includes the information about the species index α (so α, which appears in these equations, is implicitly a function of the marker index j) in order that we can later sum over ions and electrons in a single sum over j. We give the nonlinear version first,

$$\frac{dX_j}{d\tau} = -\frac{\partial \langle \Phi \rangle_j}{\partial Y}$$

$$\frac{dY_j}{d\tau} = \frac{\partial \langle \Phi \rangle_j}{\partial X} + V_{\parallel_j} k_\alpha X_j \tag{1.6}$$

$$V_{\parallel_j} = \text{const}, \quad V_{\perp_j} = \text{const}$$

$$\frac{d}{d\tau} \delta F_j = S_j$$

with

$$S_j = F^0_j ([\eta]_j - T^*_\alpha V_{\parallel_j} k_\alpha X_j) \frac{\partial \langle \Phi \rangle_j}{\partial Y}.$$

The linearized (in $\delta F_j \sim \Phi \sim \rho_i/L_n$) equations are similar. The only difference is that the $\hat{z} \times \overline{\nabla} \langle \Phi \rangle_\alpha \cdot \overline{\nabla} \delta F_\alpha$ term from Eq. (4) is dropped, so that the resulting characteristic equations are

$$X_j = \text{const}$$

$$\frac{dY_j}{d\tau} = V_{\parallel_j} k_\alpha X_j \tag{1.7}$$

$$V_{\parallel_j} = \text{const}, \quad V_{\perp_j} = \text{const}$$

$$\frac{d}{d\tau} \delta F_j = S_j$$

with

$$S_j = F^0_j ([\eta]_j - T^*_\alpha V_{\parallel_j} k_\alpha X_j) \frac{\partial \langle \Phi \rangle_j}{\partial Y}.$$

Here, $\langle \Phi \rangle_j$ is the gyroaveraged Φ evaluated at the jth position as seen by the $\alpha(j)$ species. Also $[\eta]_j = 1 + \eta_\alpha (V^2_j - \frac{3}{2})$.

Now, with Eqs. (1.6) or (1.7) we can explicitly evolve the distribution function along the characteristics. We then use the quasineutrality relation to solve for Φ.

It can be shown that the nonadiabatic part of the nongyroaveraged distribution function, $h = f + (q_\alpha \Phi/T_\alpha) f^M$, is gyrophase independent to lowest order in the gyrokinetic expansion [2, 3] (this is not true of the nongyroaveraged f). In that case, the Fourier component of the normalized real space particle density, $\delta P_{\alpha_\mathbf{K}}$ (δP_α is the normalized $\delta \rho = \int d\mathbf{v}\, f$), can be found from [3]

$$\delta P_{\alpha_\mathbf{K}} = -T^*_\alpha \Phi_\mathbf{K} + \int d\mathbf{V}_\alpha J_0 (V_{\perp_\alpha} K_{\perp_\alpha}) \delta H_{\alpha_\mathbf{K}}. \tag{1.8}$$

In Eq. (1.8), $\delta P_{\alpha_\mathbf{K}}$ is a non-gyroaveraged quantity. The J_0 factor is present in order to transform the gyroaveraged quantity, $\delta H_\alpha = \delta F_\alpha + T^*_\alpha \langle \Phi \rangle_\alpha F^M$, back to real (nongyroaveraged) space.

Quasineutrality is then

$$\delta P_{i_\mathbf{K}} = \delta P_{e_\mathbf{K}}. \tag{1.9}$$

Using Eq. (1.8) in Eq. (1.9) and converting δH_α back to δF_α, we get

$$D_\mathbf{K} \Phi_\mathbf{K} = \int d\mathbf{V}_i J_0 (V_{\perp_i} K_{\perp_i}) \delta F_{i_\mathbf{K}} - \int d\mathbf{V}_e J_0 (V_{\perp_e} K_{\perp_e}) \delta F_{e_\mathbf{K}}, \tag{1.10}$$

where

$$D_\mathbf{K} = \int d\mathbf{V}_i F^0_i (1 - J^2_0 (V_{\perp_i} K_{\perp_i})) + \frac{T_i}{T_e} \int d\mathbf{V}_e F^0_e (1 - J^2_0 (V_{\perp_e} K_{\perp_e})).$$

In practice, we will use a discrete set of velocities and markers. The parallel velocities are chosen by picking a maximum value for $V^{\max}_{\parallel_\alpha}$ and dividing the range between $-V^{\max}_{\parallel_\alpha}$ and $+V^{\max}_{\parallel_\alpha}$ into a large number of equal divisions (for the electrons, one might do better by choosing unequal divisions favoring small values). The values of $E_{\perp_\alpha} = V^2_{\perp_\alpha}$ and their weights w are chosen according to Gaussian integra-

tion rules [18] (the values of $E_{\perp_\alpha}$ are the zeros of the Laguerre polynomials) in order to optimize the evaluation of the $E_\perp$ integrals. This is an optimal choice since $E_\perp$ is a time-independent parameter in these equations, and Gaussian integration rules give better accuracy than other rules for integration over $E_\perp$. In our nonlinear code, we accumulate particles to discrete X and Y positions in order to use an FFT. In the linear code, the markers are stationary in X.

In the discrete form, the integrals over velocity and the integrals over velocity and space (the integral over space is needed to compute the Fourier transform) in Eq. (1.10) need to be converted to sums. The procedure for evaluating these sums is greatly simplified by the fact that the equations of motion for the markers in Eqs. (1.6) or (1.7) are phase space conserving. Thus each marker has a constant weighting factor which is proportional to the initial phase space volume occupied by that marker. The phase space conserving property yields another significant benefit. If we load markers in all the important regions of phase space (neglecting large values of $E_{\perp_\alpha}$ and $V_{\parallel_\alpha}$ for which the δF_α's will be insignificant), these regions of phase space will continue to be occupied by the same phase space density of markers, and thus we will never lose information about the value of the distribution function in any phase space region. (Because in the nonlinear system, the regions of phase space represented by a marker can become twisted in a complicated way, it will still be necessary to have an adequate number of markers to well represent all regions of phase space; conservation of energy as discussed below should help to determine the adequacy of the number of simulation markers.)

The velocity integral of a quantity Q_α in discrete form becomes

$$\sum_\alpha \int d\mathbf{V}_\alpha\, Q_\alpha(V_{\parallel_\alpha}, E_{\perp_\alpha}) \Rightarrow \pi \sum_b \Delta V_{\parallel_\alpha} w_b Q_b = \sum_b \overline{W}_b Q_b \quad (1.11)$$

In the integral, $d\mathbf{V}_\alpha = 2\pi V_{\perp_\alpha} dV_{\perp_\alpha} dV_{\parallel_\alpha} = \pi dE_{\perp_\alpha} dV_{\parallel_\alpha}$. The b (beam) index represents symbolically a triple index for a value of the parallel velocity $V_{\parallel_b}$, a value of the perpendicular particle energy $E_{\perp_b} = V_{\perp_b}^2$, and the α index (as was the case with the j index, α is an implicit function of b when it appears in a sum over b). In addition, $\Delta V_{\parallel_\alpha}$ is the range of $\Delta V_{\parallel_\alpha}$ ($2V_{\parallel_\alpha}^{\max}$) divided by the number of discrete $V_{\parallel_b}$'s. The weighting factors for the Gaussian integration of $\int dE_\perp Q(E_\perp)$ are w_b (these only depend on $E_{\perp_b}$). As implied in Eq. (1.11), $\overline{W}_b = \Delta V_{\parallel_\alpha} \pi w_b$.

We note that alternative accumulation rules to Eq. (1.11) have been implemented which attempt to correct for random fluctuations in the number of markers per cell and thus further reduce noise. However, these have encountered severe numerical stability or long time accuracy problems [19].

An integral over velocity and space (one species) becomes

$$\left\{\frac{1}{2\pi 2X_0}\int d\mathbf{V}\int dX\,dY\,Q(V_\parallel, E_\perp, X, Y)\right\}_\alpha \Rightarrow \frac{\Delta X \Delta Y \Delta V_{\parallel_\alpha} \pi}{2\pi 2X_0} \sum_{\substack{j \\ (\text{one } \alpha)}} w_j Q_j = \sum_{\substack{j \\ (\text{one } \alpha)}} W_j Q_j. \quad (1.12)$$

Here j is the particle index and $\Delta X \Delta Y$ is the spatial volume (area) occupied by each marker, which is equal to $2\pi 2X_0/((\text{number of particles})_\alpha/(\text{number of beams})_\alpha)$. Thus,

$$W_j = \Delta V_{\parallel_\alpha} \frac{(\text{number of beams})_\alpha}{(\text{number of particles})_\alpha} \pi w_j. \quad (1.13)$$

If the $V_{\parallel_j}$ values are loaded nonuniformly (in order to improve resolution around some resonant velocity, for example), one need only make $\Delta V_\parallel$ a function of the marker j (that is, a function of the precise marker index value, and not just of the species α) in Eq. (1.13). (Actually, $\Delta V_\parallel$ would in that case be a function of $V_{\parallel_j}$ only.)

In order to integrate over the velocities and compute the Fourier transform, we use

$$\int d\mathbf{V}\, Q_\mathbf{K} \Rightarrow \sum_j W_j e^{-i\mathbf{K}\cdot\mathbf{R}_j} Q_j. \quad (1.14)$$

With these relations, we can express Eq. (1.10) in discrete form

$$D_\mathbf{K}\Phi_\mathbf{K} = \sum_j s_\alpha W_j J_0(V_{\perp_j} K_{\perp_\alpha})\, e^{-i\mathbf{K}\cdot\mathbf{R}_j}\, \delta F_j, \quad (1.15)$$

where

$$D_\mathbf{K} = \sum_b |T_\alpha^*|\, \overline{W}_b (1 - J_0^2(V_{\perp_b} K_{\perp_\alpha}))\, F_b^0,$$

where we have used $s_\alpha = 1/-1$ for ions/electrons. The sums in Eq. (1.15) should be taken over both ions and electrons. Note that $D_\mathbf{K}$ is found from the sum over beams (discrete velocities) rather than over particles, thus only the right-hand side of Eq. (1.15) depends explicitly on the particles.

Although written compactly as a one step operation, the evaluation of the sum over the marker index j in Eq. (1.15) requires a careful procedure. In our linear code, we sum over the different $V_\parallel$'s and Y positions simultaneously, adding the δF_j values to calculate the real and imaginary parts of $e^{iY}\delta F$ at the individual $E_\perp$ and X values. Then in the second step, we use an FFT to sum over the X positions at each $E_\perp$ and calculate the K_X components. Then, knowing the $\mathbf{K}$ values (and thus $K_\perp$, which is needed within the J_0 factor), we finish the operation by summing over the $E_\perp$'s.

In the Y-direction, we have periodic boundary conditions

(the box extends from $Y = 0$ to 2π). We use a boundary condition at the box edge $X = \pm X_0$ of $\Phi = 0$ and $\delta F = 0$.

If the assumption is made that the electrons are adiabatic, then we need only sum over the ion values of the indices in Eq. (1.15). In that case, δP_{e_K} becomes $(T_i/T_e)\Phi$ and we need therefore to add T_i/T_e to the right-hand side of the definition of D_K in Eq. (1.15).

Using Eqs. (1.6) or (1.7) and then Eq. (1.15), we can explicitly advance the gyrokinetic equations. With a predictor–corrector scheme, we can do this with second-order accuracy in time. However, we are able to improve the numerical stability of the algorithm by introducing a semi-implicit scheme for the calculation of Φ. Such a scheme is possible because of the analytical nature of the source function S_j with which we evolve δF_j. For a linear code, we can make the calculations of Φ totally implicit. In order to demonstrate how to do this, we start with the equation for the evolution of the linear δF_j in Eqs. (1.7), which is

$$\frac{d}{d\tau}\delta F_j = S_j = F_j^0\,([\eta]_j - T_\alpha^*\,V_{\parallel_j}\,k_\alpha\,X_j)\frac{\partial\langle\Phi\rangle_j^{n+1}}{\partial Y}.$$

After advancing δF_j one time step $\Delta\tau$, we have

$$\begin{aligned}\delta F_j^{n+1} &= \delta F_j^n + F_j^0\left\{\Delta\tau[\eta]_j\frac{\partial\langle\Phi\rangle_j^{n+1}}{\partial Y}(Y_j^{n+1/2})\right.\\ &\qquad\left. - T_\alpha^*\,(\langle\Phi\rangle_j^{n+1}(Y_j^{n+1}) - \langle\Phi\rangle_j^{n+1}(Y_j^n))\right\}\\ &= \delta F_j^n + F_j^0\left\{\Delta\tau[\eta]_j\frac{\partial\langle\Phi\rangle_j^{n+1}}{\partial Y}(Y_j^{n+1/2})\right.\\ &\qquad\left. - T_\alpha^*\,\langle\Phi\rangle_j^{n+1}(Y_j^{n+1})\,(1 - e^{-i\Delta\tau V_{\parallel_j}k_\alpha X_j})\right\},\end{aligned}$$

where Y_j^m in $\langle\Phi\rangle_j^{n+1}(Y_j^m)$ or $(\partial\langle\Phi\rangle_j^{n+1}/\delta Y)(Y_j^m)$ is the Y_j value at which $\langle\Phi\rangle_j^{n+1}$ or $\partial\langle\Phi\rangle_j^{n+1}/\partial Y$ is evaluated. The superscripts n and $n+1$ indicate the nth and $(n+1)$th time steps, respectively, while $n+\frac{1}{2}$ indicates a centered value. We have placed the $n+1$ superscript above the $\langle\Phi\rangle$ quantities to indicate that we are going to evaluate the terms implicitly. In order to obtain stability, we have found it important to time difference the second term of S_j in Eq. (1.7) in an unusual manner. We have used the fact that the linear mode varies as e^{iY} and that the convection velocity of the marker gyrocenters is $\dot{Y}_j = V_{\parallel_j}k_\alpha X_j^0$ ($\dot{Y}_j$ is a constant for each marker in the linear case). We use the notation X_j^0 to indicate the constant unchanging X values of the elements in the linear code (to distinguish this from a later use).

The resulting expression for δF_j must be inserted into the quasineutrality relation (Eq. (1.15)) at the $(n+1)$th time step in order to evaluate Φ^{n+1}. In order to obtain a tractable expression for Φ^{n+1}, we approximate the sum over the markers by an integral. That is, we sum over the discrete $V_\parallel$'s for the sum over δF_j's while integrating analytically over the parallel velocity for the evaluation of the source term S_j. This integration over $V_\parallel$ in the source term is thus calculated as if we had an infinite set of $V_{\parallel_j}$'s. At the same time, we expand the second term on the right-hand side of the equation in a sin series in K_x,

$$\begin{aligned}&\Delta V_\parallel\sum_{V_\parallel\text{'s}}\delta F_j^{n+1}\\ &\quad= \Delta V_\parallel\sum_{V_\parallel\text{'s}}\delta F_j^n + \frac{e^{-E_\perp}}{\pi}\left\{\Delta\tau[\eta]_p'\frac{\partial\langle\Phi\rangle^{n+1}}{\partial Y}\right.\\ &\qquad\left. - T_\alpha^*\sum_{K_x'}\Phi_{K_x'}^{n+1}\,J_0(V_{\perp_p}K_{\perp_\alpha}')\,2\sin(K_x'X)\,(1 - e^{-aX^2})\right\}.\end{aligned}$$

We have defined $a = \Delta\tau^2k_\alpha^2/4$ and $[\eta]_p' = 1 + \eta_\alpha\,[E_{\perp_p} - 1]$; the index p (perpendicular beam) stands for a double index indicating a perpendicular velocity and species. Now we sum over $E_{\perp_j}$, take the Fourier transform, and transform to nongyro-averaged real space,

$$\begin{aligned}&\sum_j s_\alpha W_j\,J_0(V_{\perp_j}K_{\perp_\alpha})e^{-i\mathbf{K}\cdot\mathbf{R}_j}\,\delta F_j^{n+1}\\ &\qquad\Rightarrow\sum_j s_\alpha W_j\,J_0(V_{\perp_j}K_{\perp_\alpha})\,e^{-i\mathbf{K}\cdot\mathbf{R}_j}\,\delta F_j^n\\ &\qquad\quad - A_{\mathbf{K}}\Phi_{\mathbf{K}}^{n+1} - \sum_{\mathbf{K}'}B_{\mathbf{K},\mathbf{K}'}\,\Phi_{\mathbf{K}'}^{n+1},\end{aligned}\tag{1.16}$$

where

$$A_{\mathbf{K}} = \sum_p s_\alpha w_p e^{-E_{\perp_p}}J_0^2(V_{\perp_p}K_{\perp_\alpha})\,[-\Delta\tau[\eta]_p'\,i + T_\alpha^*]$$

and

$$\begin{aligned}B_{\mathbf{K},\mathbf{K}'} = -\sum_p w_p\,|T_\alpha^*|\,e^{-E_{\perp_p}}J_0(V_{\perp_p}K_{\perp_\alpha})\,J_0(V_{\perp_p}K_{\perp_\alpha}')\\ \times\frac{1}{2X_0}\int dX\,e^{-iK_x'X}\,[e^{-aX^2}\,2\sin(K_x'X)].\end{aligned}$$

Here, w_p is the Gaussian weight for perpendicular integration and is exactly the same as w_b defined earlier (which only depended on $V_{\perp_b}$); w_p is the part of the beam weight $\overline{W}_b$ which is left after parallel integration. We note that $B_{\mathbf{K},\mathbf{K}'}$ is a tensor coupling the K_x' components of Φ to the K_x component. Also, $s_\alpha = 1/-1$ for ions/electrons has been inserted on both sides of the equation (using $s_\alpha T_\alpha^* = |T_\alpha^*|$).

Now we can use the result of Eq. (16) to modify Eq. (15) for the calculation of Φ,

$$\sum_{\mathbf{K}'}D_{\mathbf{K},\mathbf{K}'}\,\Phi_{\mathbf{K}'}^{n+1} = \sum_j s_\alpha W_j\,J_0(V_{\perp_j}K_{\perp_\alpha})\,e^{-i\mathbf{K}\cdot\mathbf{R}_j}\,\delta F_j^n,\tag{1.17}$$

where

$$D_{\mathbf{K},\mathbf{K}'} = \left[\sum_b |T_\alpha^*|\,\overline{W}_b(1 - J_0^2(V_{\perp_b}K_{\perp_\alpha}))\,F_b^0 + A_{\mathbf{K}}\right]\delta_{\mathbf{K},\mathbf{K}'} + B_{\mathbf{K},\mathbf{K}'}.$$

The quantity $\delta_{\mathbf{K},\mathbf{K}'}$ is 1 for $\mathbf{K} = \mathbf{K}'$ and 0 otherwise.

After using Eq. (1.17) to advance Φ, it will be necessary to advance the δF_j's using Eq. (1.7) with the new Φ. By evaluating half of the evolution of δF_j explicitly with the old Φ using Eq. (1.7) and half implicitly using the result of Eq. (1.17), the scheme can be time-centered and thus second-order accurate in τ. We have found this to be as stable numerically as the fully implicit algorithm. All the results from our code presented in this paper use this time-centered partially implicit scheme. In a nonlinear code, we can use the semi-implicit method, explicitly advancing the δF_j's nonlinearly, but subtracting the explicit linear terms while adding these linear terms implicitly.

Implicit techniques for particle codes can be divided into two broad categories, moment and direct methods [21]. The moment techniques solve for a particular term by means of fluid-like equations; the direct methods solve the particle equations themselves implicitly (perhaps with approximation). The technique we have described is a direct method, but the analytical form for F_j^0, combined with the constancy of a marker's $V_{\parallel_j}$, has allowed us to integrate over $V_\parallel$ and derive quite simple moment-like terms.

Introducing the quantity $\delta\overline{H}_j = \delta H_j + X_j F_j^0[\eta]_j = \delta F_j + T_\alpha^*\langle\Phi\rangle_j F_j^0 + X_j F_j^0[\eta]_j$, we can derive a conserved energy $\mathscr{E}_{\text{tot}}$ from the nonlinear explicit equations in the following way (energy conservation with the semi-implicit equations is comparable to that with the explicit equations). We write out $(d/d\tau)\delta\overline{H}_j$ using S_j from Eq. (1.6) for $(d/d\tau)\delta F_j$. We multiply $(d/d\tau)\delta\overline{H}_j$ by $-W_j\delta\overline{H}_j/(|T_\alpha^*|F_j^0)$, sum over j, and use Eq. (1.15) to get

$$\mathscr{E}_{\text{tot}} = \mathscr{E} + \mathscr{E}_{\text{stat}}, \tag{1.18}$$

where

$$\mathscr{E} = \mathscr{E}_\Phi + \mathscr{E}_p$$

with

$$\mathscr{E}_\Phi = \sum_\alpha |T_\alpha^*| \sum_{\mathbf{K}} \frac{\Phi_{\mathbf{K}}^2}{2},$$

$$\mathscr{E}_p = \sum_j W_j\left[-\frac{1}{|T_\alpha^*|}\frac{\delta\overline{H}_j^2}{2F_j^0} + s_\alpha F_j^0[\eta]_j X_j\langle\Phi\rangle_j\right],$$

$$\mathscr{E}_{\text{stat}} = \sum_j W_j \int dt\,[-s_\alpha F_j^0[\eta]_j X_j \dot{Y}_{\parallel_j}\dot{X}_j + |T_\alpha^*|F_j^0\langle\Phi\rangle_j\dot{Y}_{\parallel_j}\dot{X}_j - s_\alpha F_j^0[\eta]_j\langle\Phi\rangle_j\dot{X}_j]$$

$$+ \sum_j W_j|T_\alpha^*|F_j^0\frac{\langle\Phi\rangle_j^2}{2} - \sum_{\mathbf{K}}\sum_b \overline{W}_b|T_\alpha^*|\frac{\Phi_K^2}{2}J_0^2(V_{\perp_b}K_{\perp_\alpha})F_b^0.$$

Here, $\dot{X}_j = -\partial\langle\Phi\rangle_j/\partial Y$ and is the X component of the marker's gyrocenter velocity; $\dot{Y}_{\parallel_j} \equiv V_{\parallel_j}k_\alpha X_j$ and is that part of the marker's gyrocenter velocity in the $\hat{Y}$-direction which is due to the parallel velocity.

In Eqs. (1.18), in the limit of an infinite number of particles, $\mathscr{E}$ is the energy which can be derived directly from the kinetic equations, Eq. (1.4). It is equivalent to the physical Vlasov energy $\int dX\,dY\,(\int d\mathbf{V}\,f(mV^2/2) + E^2/(8\pi))$ in the quasineutral gyrokinetic limit (this is not obvious). It is therefore what we consider to be the physical energy; it is composed of a particle-dependent part $\mathscr{E}_p$ and a purely field-dependent part $\mathscr{E}_\Phi$. However, in discrete form, the genuinely conserved quantity is $\mathscr{E}_{\text{tot}} = \mathscr{E} + \mathscr{E}_{\text{stat}}$, where $\mathscr{E}_{\text{stat}}$ results from statistics (finite number of particles).

The energy relation is slightly more complicated for the linear equations. Although the X positions can be considered to be constant for the purpose of calculating the evolution of δF_j and Φ (to the order of our equations), the linearly changing X position is needed for the energy relation. Thus we distinguish two different X values in the linear case. First, there is X_j^0, which is the initial X position of the marker and is used for the calculation of $\dot{Y}_j = V_{\parallel_j}k_\alpha X_j^0$. Second, we have $\overline{X}_j$, which is the linearly changing (to high order) position of a particle. In order to check for energy conservation, $\overline{X}_j$ must be evolved using $d\overline{X}_j/d\tau = \dot{X}_j = -\partial\langle\Phi\rangle_j/\partial Y$. In order to derive the linear energy relation, we use $\delta\overline{H}_j$ defined using $\overline{X}_j$; that is, $\delta\overline{H}_j = \delta H_j + \overline{X}_j F_j^0[\eta]_j = \delta F_j + T_\alpha^*\langle\Phi\rangle_j F_j^0 + \overline{X}_j F_j^0[\eta]_j$. Using these definitions, we use a similar derivation to that used for the nonlinear equations to find

$$\mathscr{E}_{\text{tot}} = \mathscr{E} + \mathscr{E}_{\text{stat}}, \tag{1.19}$$

where

$$\mathscr{E} = \mathscr{E}_\Phi + \mathscr{E}_p$$

with

$$\mathscr{E}_\Phi = \sum_\alpha |T_\alpha^*| \sum_{\mathbf{K}} \frac{\Phi_{\mathbf{K}}^2}{2},$$

$$\mathscr{E}_p = \sum_j W_j\left[-\frac{1}{|T_\alpha^*|}\frac{\delta\overline{H}_j^2}{2F_j^0} + s_\alpha F_j^0[\eta]_j\overline{X}_j\langle\Phi\rangle_j + s_\alpha F_j^0[\eta]_j\dot{Y}_j\frac{\overline{X}_j^2}{2}\right],$$

and

$$\mathscr{E}_{\text{stat}} = \sum_j W_j \int dt [|T_\alpha^*| F_j^0 \langle\Phi\rangle_j \dot{Y}_j \dot{X}_j - s_\alpha F_j^0 [\eta]_j \langle\Phi\rangle_j \dot{X}_j]$$
$$+ \sum_j W_j |T_\alpha^*| F_j^0 \frac{\langle\Phi\rangle_j^2}{2} \tag{1.20}$$
$$- \sum_{\mathbf{K}} \sum_b \overline{W}_b |T_\alpha^*| \frac{\Phi_K^2}{2} J_0^2 (V_{\perp_b} K_{\perp_\alpha}) F_b^0.$$

We note that the first term in the nonlinear $\mathscr{E}_{\text{stat}}$ is a part of $\mathscr{E}$ in the linear case. The difference comes about because of the distinction between the simple coordinate X_j for the nonlinear calculation and the linearly changing $\overline{X}_j$.

As previously stated, $\mathscr{E}$ is the physical energy. If $\mathscr{E}$ is conserved, we believe that we are reproducing the correct physics in our simulation. $\mathscr{E}_{\text{stat}}$ is an extra term which vanishes in the limit of an infinite number of markers. (In the limit of infinite particles, the sum becomes an integral which vanishes.) The term $\mathscr{E}_{\text{stat}}$ arises due to the fact that we will be using a discrete set of particles. The sum of the two, $\mathscr{E}_{\text{tot}}$, should be conserved if we are accurately solving the differential equations, even for a very small set of particles. These two quantities, $\mathscr{E}_{\text{tot}}$ and $\mathscr{E}$ serve as diagnostics for different types of numerical inaccuracy. Inaccuracy due to time or space finite differencing errors is manifested in nonconservation of $\mathscr{E}_{\text{tot}}$. If $\mathscr{E}_{\text{tot}}$ is well conserved but $\mathscr{E}$ is not, this signifies that there is an insufficient number of markers for the simulation to well represent the limit of a continuous distribution function. Therefore, we will assume that if $\mathscr{E}_{\text{tot}}$ is not conserved, we need to increase accuracy by decreasing time or space step sizes. If $\mathscr{E}_{\text{tot}}$ is conserved, but $\mathscr{E}$ is not, we need to increase the number of particles.

2. NUMERICAL RESULTS

We now present data from simulation runs using the linear δf code. In Fig. 1, we plot time dependent traces of ω_r and γ for a simulation of the η_i mode (driven by the ion temperature gradient [22]) using $\eta_i = 4.0$, $k_i (\equiv 2L_n/L_s) = 0.5$, $k_y \rho_i = 0.71$, $T_i/T_e = 1.0$, and $X_0 = 5.0$ with adiabatic electrons assumed. (Note that ρ_i/L_n is arbitrary; it has been assumed to be small and is scaled out of the δf equations.) For our δf particle code run, we used 32,768 markers, with 64 equally spaced X values, 4 Gaussian $E_\perp$'s, and 16 values of the parallel velocity represented. The time step $\Delta\tau$ for the run plotted in Fig. 1 was 0.1. The values of ω_r and γ plotted at time τ in the traces are time averaged over one preceding growth time $\Delta\tau_\gamma \equiv 1/\gamma^d \sim 6.99\ \omega_{*_i}^{-1}$ using values which are based on the variation of the fundamental Fourier harmonic $\phi_K(f)$ over $\Delta\tau_\gamma$ (using $\exp((-i\omega_r(\tau) + \gamma(\tau))\tau) = \phi_K(\tau)/\phi_K(\tau - \Delta\tau_\gamma)$). The data is displayed over a total time equal to 14 growth times after an initial transient.

We now define the following quantities: $\overline{\omega}_r$ and $\overline{\gamma}$ are the average values of ω_r and γ over the time of the run; $\Delta\overline{\omega}_r$ and $\Delta\overline{\gamma}$ are the differences of these quantities from the values given by a linear gyrokinetic dispersion code ω_r^d and γ^d; and $\Delta\tilde{\omega}_r$ and $\Delta\tilde{\gamma}$ are the maximum deviations of ω_r and γ from $\overline{\omega}_r$ and $\overline{\gamma}$ ($\Delta\tilde{\omega}_r$ is roughly the maximum value of $|\omega_r - \overline{\omega}_r|$ over the length of the run—when interpreting the meaning of $\Delta\tilde{\omega}_r$ and $\Delta\tilde{\gamma}$, it is important to remember that ω_r and γ have themselves been averaged over one growth time). Our linear dispersion code solves full integral gyrokinetic equations equivalent to Eqs. (1.7) and (1.15); with it, we find the real frequency and growth rate of the linear mode to be $\omega_r^d = 0.411\ \omega_{*_i}$ and $\gamma^d = 0.143\ \omega_{*_i}$. The horizontal dashed lines in Fig. 1 are plotted at a vertical position corresponding to these values. In Table I we list the values of $\overline{\omega}_r$, $\Delta\overline{\omega}_r/\omega_r^d$, $\Delta\tilde{\omega}_r/\omega_r^d$, $\overline{\gamma}$, $\Delta\overline{\gamma}/\gamma^d$, and $\Delta\tilde{\gamma}/\gamma^d$ for the run previously described (with $\Delta\tau = 0.1$) in the first row of Table I. Since the errors are small (the largest being $\tilde{\gamma}/\gamma^d = 0.14$), we conclude that the δf code represents the linear mode well with 32,768 particles.

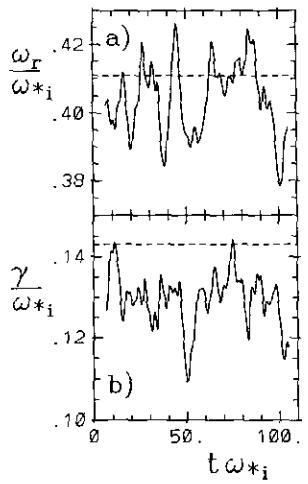

FIG. 1. Time dependent traces of (a) ω_r/ω_{*_i} and (b) γ/ω_{*_i} for a simulation of the η_i mode using $\eta_i = 4.0$, $k_i (\equiv 2L_n/L_s) = 0.5$, $k_y\rho_i = 0.71$, $T_i/T_e = 1.0$, and $X_0 = 5.0$ with adiabatic electrons assumed. The horizontal dashed lines are plotted at a vertical position corresponding to the gyrokinetic dispersion code values (a) $\omega_r^d/\omega_{*_i} = 0.411$, and (b) $\gamma^d/\omega_{*_i} = 0.143$.

The profile of Φ in the $\hat{X}$-direction at a late time is displayed in Fig. 2a. The solid and dotted lines are the real and imaginary components of $\Phi(X)$, respectively ($\Phi(X, Y) = \text{Re}[(\Phi_{\text{re}}(X) + i\Phi_{\text{im}}(X))e^{iY}]$). The profile for the same case generated by the linear dispersion code is displayed in Fig. 2b. There is good agreement between the two.

We have shown that we can accurately simulate an η_i mode with a small number of markers. This is because there is no noice associated with the equilibrium part of the distribution. There is in the δf Algorithm a level of noise, but that noise is associated with the representation of δF_j, and F_j^0. Further evidence of the low level of inherent noise in the algorithm is provided in Fig. 3, where the energy term $\mathscr{E}_\Phi$ is plotted versus time for a one-dimensional simulation of a Landau-damped shearless drift wave [23]. Without shear, we have $k_\parallel \to k_i$ rather than $k_i X$. The value of k_i for this run is 1.0. In addition, $\eta_i = 0.0$ (drift wave only), $T_i/T_e = 1.0$, $X_0 = 10^4$, and $k_y\rho_i \Rightarrow 0$ (drift kinetics). (Again we assume adiabatic electrons.) We use

TABLE I

$\Delta\tau$	$\overline{\omega}_r/\omega_{*_i}$	$\Delta\overline{\omega}_r/\omega_r^d$	$\Delta\bar{\omega}_r/\omega_r^d$	$\overline{\gamma}/\omega_{*_i}$	$\Delta\overline{\gamma}/\gamma^d$	$\Delta\bar{\gamma}/\gamma^d$	$\lvert\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi\rvert$	$\lvert\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi\rvert$
0.1	0.405	−0.015	0.049	0.129	−0.099	0.14	3.6×10^{-4}	0.074
0.8	0.402	−0.022	0.068	0.128	−0.10	0.13	0.023	0.048
3.2	0.338	−0.18	0.049	0.127	−0.11	0.17	0.76	0.80
12.8	−0.05	−1.1		0.04	−0.72	0.24	1.9	1.5

32,768 ion markers with 4 Gaussian $E_\perp$'s and 4096 values of the parallel velocity represented. (A large number of parallel velocities is needed to accurately represent the resonance effects.) In Fig. 3 we see that $\mathscr{E}_\Phi$ decreases by four orders of magnitude (Φ decreases by two orders of magnitude) down to the noise level. The frequency and growth rate produced by the δf particle code over the first 30 $\omega_{*_i}^{-1}$ is on average $\overline{\omega}_r = -1.83\ \omega_{*_i}$ and $\overline{\gamma} = -0.16\ \omega_{*_i}$, which are close to the dispersion code values for these parameters, $\omega_r^d = -1.85\ \omega_{*_i}$ and $\gamma^d = -0.192\ \omega_{*_i}$. If more markers and values of parallel velocity are used, $\mathscr{E}_\Phi$ will damp to a lower level. In runs with a sheared magnetic field, $\mathscr{E}_\Phi$ was similarly damped, but only two orders of magnitude using the same number of markers and values of parallel velocity.

The above results for the δf code are typical. The δf algorithm is capable of accurately describing microinstabilities of interest in fusion with far fewer particles than a standard code. However, it is also important in any numerical scheme to have diagnostics to verify the accuracy of the run. We have found, for example, that the accuracy of the δf code very rapidly deteriorates when the number of particles is insufficient. If the results of a run were erroneous because of an inadequate number of particles, it may not be obvious from the field or particle density plots. Thus, we turn our attention to the quantities $\mathscr{E}_{tot}$ and $\mathscr{E}$ and their use as diagnostics to detect inaccuracy.

Table I lists data for runs with large timesteps. As $\Delta\tau$ is increased from 0.1 to 0.8, the errors in the mean values $\overline{\omega}_r$ and $\overline{\gamma}$ change very little. However, as $\Delta\tau$ is further increased to 3.2, the errors become a significant fraction of the mean (~20%), and at $\Delta\tau = 12.8$, the errors in $\overline{\omega}_r$ and $\overline{\gamma}$ are larger than the correct values. (That the accuracy decreases at such large values of $\Delta\tau$ is not surprising considering that the growth time of the mode is $1/\gamma^d = 6.99\ \omega_{*_i}^{-1}$.)

This deterioration is paralleled by an increase in the quantity $\lvert\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi\rvert$. As described in the last section, $\mathscr{E}_{tot}$ is the sum of the physical energy $\mathscr{E}$ and a statistics-dependent term $\mathscr{E}_{stat}$ (which vanishes as the number of markers becomes infinite). The change in $\mathscr{E}_{tot}$ over the total length of the run is $\Delta\mathscr{E}_{tot}$ while $\Delta\mathscr{E}_\Phi$ serves as a convenient reference for the size of $\Delta\mathscr{E}_{tot}$ ($\mathscr{E} = \mathscr{E}_\Phi + \mathscr{E}_p$ and there is no reason why $\mathscr{E}_\Phi$ should be conserved; in fact, $\mathscr{E}_\Phi \propto \Phi^2$). Nonconservation of $\mathscr{E}_{tot}$, as indicated by an appreciable value of $\lvert\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi\rvert$, indicates that the differential equations are not being accurately solved and thus it indicates a lack of resolution in time or space. Indeed, as the timestep, $\Delta\tau$, is increased from the converged value 0.8 to 3.2,

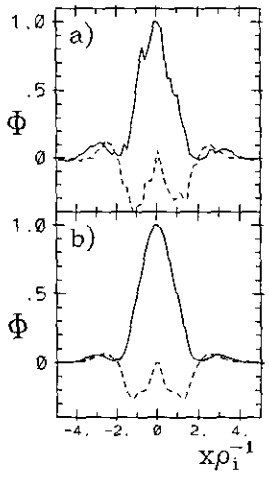

FIG. 2. Profile of Φ (arbitrary units) versus $X \equiv x\rho_i^{-1}$ for the parameters of Fig. 1: (a) the result from the δf particle code at a late time; (b) the result from the gyrokinetic dispersion code.

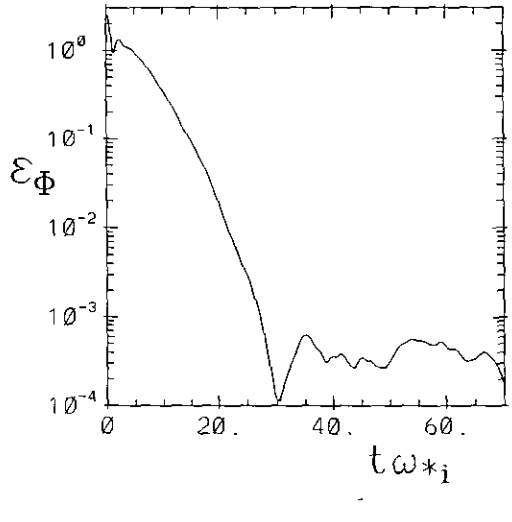

FIG. 3. The energy term $\mathscr{E}_\Phi$ is plotted versus time for a Landau-damped shearless drift wave. The value of k_i for this run is 1.0. In addition, $\eta_i = 0.0$ (drift wave only), $T_i/T_e = 1.0$, $X_0 = 10^4$, and $k_y\rho_i \Rightarrow 0$ (drift kinetics). We assume adiabatic electrons.

TABLE II

N_i	$\bar{\omega}_r/\omega_{*_i}$	$\Delta\bar{\omega}_r/\omega_r^d$	$\Delta\tilde{\omega}_r/\omega_r^d$	$\bar{\gamma}/\omega_{*_i}$	$\Delta\bar{\gamma}/\gamma^d$	$\Delta\tilde{\gamma}/\gamma^d$	$\lvert\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi\rvert$	$\lvert\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi\rvert$
65536	0.387	−0.058	0.024	0.135	−0.056	0.098	1.9×10^{-3}	1.4×10^{-4}
16384	0.43	0.046	0.060	0.135	−0.056	0.25	9.5×10^{-4}	7.0×10^{-3}
8192	0.25	−0.39	0.12	0.11	−0.23	0.49	0.015	0.12
2048	0.5	0.22	1.2	0.05	−0.65	3.5	0.064	0.41

at which the error has increased, $|\Delta\mathscr{E}_{tot}/\mathscr{E}_\Phi|$ increases from 0.023 to 0.76. Our energy relation in the discrete form indicates that $\mathscr{E}_{tot}$, rather than $\mathscr{E}$ should be exactly conserved; thus, we cannot expect that $\mathscr{E}$ will be conserved if $\mathscr{E}_{tot}$ is not, and indeed, the quantity $|\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi|$ which is also tabulated in Table I increases along with $|\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi|$ as $\Delta\tau$ is increased.

The effect of varying the number of markers is demonstrated in Table II, where data is presented for runs with the number of ion markers N_i equal to 64 K, 16 K, 8 K, and 2 K. In all of these cases, the electrons are again assumed to be adiabatic, and $\Delta\tau = 0.1$. As N_i is decreased from 65,536 to 2048, the accuracy in $\bar{\omega}_r$ and $\bar{\gamma}$ deteriorates. In addition (unlike the case when $\Delta\tau$ was increased), the level of fluctuations increases. For 16 K markers, $\Delta\tilde{\gamma}/\gamma^d = 0.25$, and the other error indictors, $\Delta\bar{\omega}_r/\omega_r^d$, $\Delta\tilde{\omega}_r/\omega_r^d$, and $\Delta\bar{\gamma}/\gamma^d$ are all no larger than 6%. When N_i is decreased from 16,384 to 8192, all the error indicators increase quite significantly; and with $N_i = 2048$, all are order unity.

Looking at the variation in the relative fluctuation levels $\Delta\tilde{\omega}_r/\omega_r^d$ and $\Delta\tilde{\gamma}/\gamma^d$ over the entire range of particle numbers, it appears that the fluctuations (and therefore the noise) scale roughly like the inverse of the number of particles, N_i. (For instance $\Delta\tilde{\gamma}/\gamma^d$ decreases by a factor of 37 for the increase in particles of a factor of 32.) Thus not only is the noise greatly reduced using the δf Algorithm, but the scaling is more favorable (the noise decreases at a faster rate as the particle number is increased) since the usual scaling for noise is $N_i^{-1/2}$.

In all of the cases represented in Table II, $\mathscr{E}_{tot}$ is conserved ($|\Delta\mathscr{E}_{tot}/\Delta\mathscr{E}_\Phi|$ being at most 0.064 for the $N_i = 2048$ case). The physical energy, $\mathscr{E}$, however, is certainly not conserved in the 2048 marker case, for which $|\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi| = 0.41$. When $\mathscr{E}_{tot}$ is conserved, indicating that the differential equations are being solved correctly, conservation of $\mathscr{E}$ further indicates that there are enough markers to represent the continuous limit of an infinite number of markers. As we can see from Table II, $\mathscr{E}$ is well conserved for N_i down to 16,384 ($|\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi| = 7.0 \times 10^{-3}$), but $|\Delta\mathscr{E}/\Delta\mathscr{E}_\Phi| = 0.12$ for 8192 markers and 0.41 for 2048 markers. Thus, as expected from the analysis, when $\mathscr{E}_{tot}$ is conserved but $\mathscr{E}$ is not, this is indicative that the number of particles is not adequate. Conservation of $\mathscr{E}_{tot}$ and $\mathscr{E}$ indicates that both the time and space resolution and the number of particles are adequate for accurate simulation of the problem.

So far, all the results we have presented have assumed an adiabatic electron response. The δf code can be run with kinetic electrons by including electron as well as ion markers using the formalism we developed in the last section. The use of kinetic electrons is expensive in standard codes because the Courant stability condition $K_\parallel V_e \Delta\tau < 1$ mandates small time steps. Using the semi-implicit scheme presented in Section 1, however, we can violate the Courant condition with large $\Delta\tau$ chosen to resolve the phenomena of interest. But before we proceed with a description of the code results using electron as well as ion markers, we would like to note that when electron markers were included, the performance of the code was found to be degraded in four ways: (1) the fluctuations in ω_r and γ were larger; (2) the energy was not as precisely conserved; (3) more markers were required for accuracy and to avoid numerical instability; and (4) some cases with parameters for which the modes should be stable or damped are numerically unstable, having a small but finite γ. To illustrate this last point, for $\eta_i = \eta_e = 0$, $T_i/T_e = 1.0$, $k_i = 0.2$, $X_0 = 6.0$, and $k_y\rho_i = 0.8$ (with a realistic mass ratio), the gyrokinetic dispersion code yields $\omega_r^d = -0.482\ \omega_{*_i}$ and $\gamma^d = -0.108\ \omega_{*_i}$; but the δf particle code using 32,768 markers for each species is unstable with $\gamma \sim 0.01\ \omega_{*_i}$. This instability can be eliminated with a larger number of particles (131,072 for this case).

Despite the problems, we have been able to show that the linear δf particle code with kinetic electrons can produce results in agreement with the dispersion code for parameters for which the real frequency and growth rate are substantially different from those that result when the electron distribution is assumed to be adiabatic [20]. We now present data for runs with kinetic electrons. The parameters are: $\eta_i = 2.0$, $\eta_e = 1.0$, $k_y\rho_i = 0.75$, $k_i = 0.2$ (that is, $L_s/L_n = 10.0$), and $T_i/T_e = 1.0$. The dispersion code yields $\omega_r^d = 0.0893\ \omega_{*_i}$ and $\gamma = 0.0353\ \omega_{*_i}$ with full electron dynamics; with adiabatic electrons, $\omega_r = 0.0721\ \omega_{*_i}$ and $\gamma = 0.0504\ \omega_{*_i}$, respectively. Thus there is a substantial change brought about from the inclusion of kinetic electron dynamics. (There is not such a large difference for $\eta_e = \eta_i$.)

We use an especially large box with $X_0 = 16.0$ (the mode only extends to about $X = 4.0$) in order to demonstrate the sufficiency of the semi-implicit method for three-dimensional nonlinear runs for which we would use a large box with a number of rational surfaces. We make three runs, using 131,072, 65,536, and 32,768 markers per particle species, with 256 X values, 4 Gaussian $E_\perp$'s, and 4096 $V_\parallel$'s represented. The time step is 0.5. The value of $K_\parallel$ at the edge of the box X_0 is $K_\parallel = X_0 k_e = X_0 k_i \sqrt{m_i/m_e} = (42.8)(0.2)(16.0) = 137.$, so the Courant

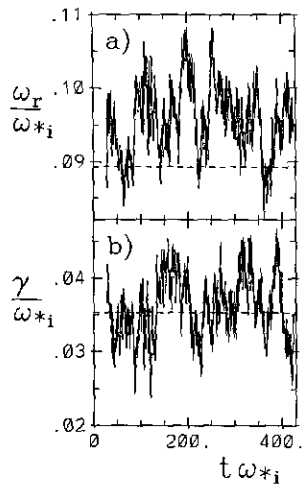

FIG. 4. Time dependent traces of (a) ω_r/ω_{*_i} and (b) γ/ω_{*_i} for a simulation of the η_i mode using kinetic (particle) electrons. The parameters are: $\eta_i = 2.0$, $\eta_e = 1.0$, $k_y\rho_i = 0.75$, $k_i = 0.2$ (that is, $L_n/L_n = 10.0$), and $T_i/T_e = 1.0$. The horizontal dashed lines are plotted at a vertical position corresponding to the gyrokinetic dispersion code values: (a) $\omega_r^d/\omega_{*_i} = 0.0721$; (b) $\gamma^d/\omega_{*_i} = 0.0504$.

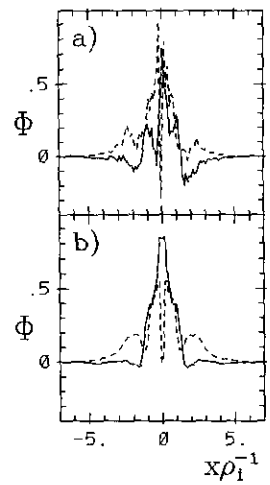

FIG. 5. Profile of Φ (arbitrary units) versus $X \equiv x\rho_i^{-1}$ for the parameters of Fig. 4: (a) the result from the δf particle code at a late time; (b) the result from the gyrokinetic dispersion code.

condition for the electrons, $\Delta\tau < 1/(K_\parallel V_{\parallel_e}) \Rightarrow 0.5 < 1/((137.)(1.0))$ is strongly violated. Despite this, the time step is sufficiently small to resolve the physics of interest since the value of $\Delta\tau$ is much smaller than the wave period $2\pi/\omega_r^d \approx 70.\ \omega_{*_i}^{-1}$ and the growth time $1/\gamma^d = 28.3\ \omega_{*_i}^{-1}$. Since the mode width is about one-fourth the box size, the effective number of markers (within the region of the mode) is about 32,768, 8192, and 4096, respectively. The values of ω_r and γ (averaged over one growth time) for the case of the run with 131,072 markers per particle species are displayed in Fig. 4 over a length of time equal to 14 growth times. In Table III, we list the values of $\overline{\omega}_r$, $\Delta\overline{\omega}_r/\omega_r^d$, $\Delta\tilde{\omega}_r/\omega_r^d$, $\overline{\gamma}$, $\Delta\overline{\gamma}/\gamma^d$, and $\Delta\tilde{\gamma}/\gamma^d$ for all three runs (with different numbers of particles). The simulation yields values of ω_r and γ which agree with the dispersion code values to about 10%, except for the case with 32,768 markers. For this small a number of markers, the code was numerically unstable as indicated by the error in ω_r and γ and the poor energy conservation. The profile of Φ in the $\hat{X}$-direction at a late time is displayed in Fig. 5a for the run with 131,072 markers per particles species, while the eigenfunction found from the dispersion code is shown in Fig. 5b. The value of Φ is plotted only between $x\rho_i$ equal to -7 and $+7$, although the simulation box extended from -16 to $+16$.

3. CONCLUSIONS

A particle code using the δf Algorithm produces accurate results with much less noise than a conventional particle code. Not only is the noise reduced, but the scaling of the noise, roughly N_α^{-1}, is more favorable. (On the other hand, although the number of particles required is fewer, one must be sure to have enough particles, for when the number of particles is reduced below the required number, the noise will increase very rapidly.)

The noise from the background distribution is absent in the δf Algorithm. This is particularly important for parameters arising in fusion confinement devices, where $\delta f/f_0 \sim \rho_i/L_n \sim 10^{-2} - 10^{-3}$. For such parameters, it is sufficient to use the lowest order gyrokinetic equation, which is valid in the limit $\rho_i/L_n \to 0$. Then ρ_i/L_n appears only as a scaling parameter and has no effect on the accuracy or expense of the simulation. With conventional particle code methods, however, decreasing ρ_i/L_n decreases the signal to noise ratio. Thus it is difficult to use a value of ρ_i/L_n (or strength of the equilibrium gradients) which is small enough to well approximate the $\rho_i/L_n \to 0$ limit, without using exorbitant numbers of particles. This is true even for parameters such that there is a strongly unstable mode. For parameters approaching marginal stability (which are typical of actual experiments and planned reactors, e.g., $\eta \sim 2$), con-

TABLE III

N_i	$\overline{\omega}_r/\omega_{*_i}$	$\Delta\overline{\omega}_r/\omega_r^d$	$\Delta\tilde{\omega}_r/\omega_r^d$	$\overline{\gamma}/\omega_{*_i}$	$\Delta\overline{\gamma}/\gamma^d$	$\Delta\tilde{\gamma}/\gamma^d$	$\lvert\Delta\mathscr{E}_{\omega}/\Delta\mathscr{E}_{\Phi}\rvert$	$\lvert\Delta\mathscr{E}/\Delta\mathscr{E}_{\Phi}\rvert$
131072	0.096	0.075	0.22	0.036	0.020	0.23	0.091	0.34
65536	0.085	−0.048	0.19	0.040	0.13	0.28	1.2	1.1
32768	−1.5	−18	3.4	0.06	0.7	0.6	2.9	2.8

ventional codes are wholly prohibitive. Thus in this most interesting case in which the plasma is close to marginal stability, the δf Algorithm has a tremendous advantage over conventional techniques. We note that the low noise advantages of the δf Algorithm are also present in nonlinear simulation [11]. We have also shown that it is possible to monitor the accuracy of the simulation using the energy terms $\mathscr{E}_{tot}$ and $\mathscr{E}$. Conservation of $\mathscr{E}_{tot}$ indicates that there is adequate space and time resolution, while conservation of $\mathscr{E}$ further indicates that there is an adequate number of markers to represent the continuous limit.

When the linear δf code is run using electron markers as well as ion markers, there is increased noise and less exact energy conservation. Nevertheless, the code produces results in agreement with a gyrokinetic dispersion code; it allows one to simulate cases in which the linear behavior is substantially different from that which results from the adiabatic approximation. A more serious problem is that weak numerical instabilities have been found in runs using parameters which should produce a stable or damped mode. These instabilities can be eliminated by using a large number of markers. It is not known whether such numerical instabilities would be a problem in a nonlinear simulation.

The semi-implicit method we have developed for the evolution of Φ allows us to use a time step $\Delta\tau$ larger than that which violates the Courant condition. Thus, the time step can be chosen just small enough to resolve the modes of interest.

The advantages of the δf Algorithm, the ability to run with a relatively small number of particles with low noise, and the unrestricted time step, should make it possible to simulate turbulent plasmas in three-dimensional geometry using realistic parameters (at least with kinetic ions, and perhaps with electrons as well).

ACKNOWLEDGMENTS

This research was supported by the U.S. Dept. of Energy Contract DE-FG05-80ET-53088. R.D. was also partially supported by NASA Grant NAGW-1652.

REFERENCES

1. P. J. Catto, *Plasma Phys.* **20,** 719 (1978).
2. E. A. Frieman and L. Chen, *Phys. Fluids* **25,** 502 (1982).
3. P. L. Similon, Ph.D. thesis, Princeton University, October 1981 (unpublished).
4. W. W. Lee, *Phys. Fluids* **26,** 556 (1983).
5. W. W. Lee, *J. Comput. Phys.* **72,** 243 (1987).
6. R. D. Sydora, *APS Div. Plasma Phys., 1989 Meeting,* Bulletin **34,** p. 2138, 1989 (unpublished).
7. T. Tajima and F. W. Perkins, *Sherwood Theory Meeting, 1983,* Paper 2P9 (unpublished).
8. Z. Mikic and E. C. Morse, *Phys. Fluids* **30,** 2806 (1987).
9. A. M. Dimits and W. W. Lee, *Proceedings, 12th Conference on the Numerical Simulation of Plasmas, San Francisco, California, 1987,* Paper PW23; *J. Comput. Phys.* **107,** 309 (1993).
10. M. Kotschenreuther, *APS Div. Plasma Phys., 1988 Meeting,* Bulletin **33,** p. 2107, 1988 (unpublished).
11. M. Kotschenreuther *et al., Plasma Physics and Controlled Nuclear Fusion Research 1990, Thirteenth Conf. Proceedings, Washington, DC,* Vol. 2, p. 351, 1990.
12. S. E. Parker and W. W. Lee, *Phys. Fluids B* **5,** 77 (1992).
13. G. Hu and J. A. Krommes, *Phys. Plasmas* **1,** 863 (1994).
14. T. P. Armstrong, R. C. Harding, G. Knorr, and D. Montgomery, "Solution of Vlasov's Equation by Transform Methods," in *Methods in Computational Physics* (Academic Press, New York, 1970), p. 30.
15. P. C. Liewer, *Nucl. Fusion* **25,** 543 (1985).
16. G. DiPeso, E. C. Morse, and R. W. Ziolkowski, *J. Comput. Phys.* **96,** 325 (1991).
17. D. C. Barnes, R. A. Nebel, and C. E. Seyler, *APS Div. Plasma Phys., 1992 Meeting,* Bulletin **37,** p. 1557, 1992 (unpublished).
18. M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions* (Dover, New York, 1972), p. 916.
19. T. Tajima, private communication.
20. R. Linsker, *Phys. Fluids* **24,** 1485 (1981).
21. T. Tajima, *Computational Plasma Physics: With Applications to Fusion and Astrophysics* (Addison–Wesley, Redwood City, CA, 1989).
22. W. M. Tang, G. Rewoldt, and L. Chen, *Phys. Fluids* **29,** 3715 (1986).
23. G. Schmidt, *Physics of High Temperature Plasmas* (Academic Press, New York, 1979).