Computer Physics Communications 134 (2001) 58–77

Computer Physics Communications

www.elsevier.nl/locate/cpc

Gridless finite-size-particle plasma simulation

G. Vlad [a], S. Briguglio [a,*], G. Fogaccia [a], B. Di Martino [b]

[a] *Associazione Euratom-ENEA sulla Fusione, C.R. Frascati, C.P. 65, I-00044 Frascati, Rome, Italy*
[b] *Dip. Ingegneria dell'Informazione, Second University of Naples, Naples, Italy*

Received 19 April 1999; received in revised form 18 February 2000; accepted 11 July 2000

Abstract

The main features of gridless finite-size-particle (FSP) codes are discussed, from the point of view of the performances that can be obtained, with respect both to the spatial-resolution level and the efficiency of parallel particle simulations. It is shown that such codes are particularly suited for particle-decomposition parallelization on distributed-memory architectures, as they present a strong reduction, in comparison with particle-in-cell (PIC) codes, of the memory overhead related to storing the replicated fluctuating-field arrays. © 2001 Elsevier Science B.V. All rights reserved.

1. Introduction

One of the main issues of thermonuclear fusion research is represented by the investigation of small-scale electromagnetic fluctuations in magnetically confined plasmas and their connection with the high values of the transport coefficients observed in the tokamak experiments [1]. The full comprehension of such phenomena requires a self-consistent treatment of the interactions between electromagnetic waves and the plasma, which cannot be satisfactorily described in terms of few moments of the particle distribution function (i.e. by a fluid treatment). A full kinetic approach is instead requested, and significant results cannot be obtained, apart from the case of particular asymptotic regimes, on the basis of analytical studies. Numerical simulation is then needed to get insight in the plasma turbulent behaviour and, in particular, to determine the saturation mechanisms, the level of the saturated fluctuations and their effect on the plasma confinement.

Among the many existing numerical tools, particle simulation codes [2,3] seem to be the most suited one, as they directly study the time evolution of the particle distribution function and the mutual interaction between such a distribution and the fluctuating electromagnetic fields. Particle simulation indeed consists in performing self-consistent *particle pushing*: the phase-space coordinates of a particle population are evolved in the electromagnetic fields computed, at each time step, in terms of the contribution yielded by the particles themselves (e.g., pressure perturbation).

The most important class of particle simulation codes is that of the so-called *particle-in-cell* (PIC) codes, which compute the fluctuating fields and the required particle contribution only at the nodes of a spatial grid, then interpolating the fields at the (continuous) particle position in order to evolve particle coordinates.

* Corresponding author.
E-mail address: briguglio@frascati.enea.it (S. Briguglio).

0010-4655/01/$ – see front matter
PII: S0010-4655(00)00191-0

Because of the large ratio between the equilibrium scale lengths and the fluctuation ones (typically, several tens or more), high spatial resolution is required in such simulations (up to millions grid cells for three-dimensional PIC codes). Such a requirement, along with the need for ensuring an adequate description of the velocity-space dependence of the particle distribution function, makes the usage of large numbers of simulation particles (tens or hundreds millions) necessary. The limitation of the resources of present computational nodes makes the full exploitation of parallel computers unavoidable.

Two main techniques have been adopted in parallelizing PIC codes on distributed-memory architectures. The first approach is based on the so-called *domain decomposition* [4]: different portions of the physical domain and of the corresponding grid are assigned to different processors, together with the particles that reside on them. In this way the memory resources required to each computational node are reduced by a factor equal (in an average sense) to the number of processors. An almost linear scaling of the attainable physical-space resolution (more precisely, the maximum number of spatial cells) with the number of processors is then obtained. However, inter-processor communication is required when a particle migrates from one processor to another, and this can cause the parallel version of the code to be difficult to implement and/or inefficient; moreover, serious load-balancing problems can arise because of such a migration.

A complementary approach to domain decomposition is represented by *particle decomposition* [6]. It corresponds to replicating the whole spatial domain on each processor, while distributing the particle population. No particle migrates from one processor to another, because no particle meets, in its motion, the unphysical boundaries introduced by domain decomposition, and the communication among processors (needed to collect the different-processor particle contributions to the pressure perturbation) is both reduced and simplified. Moreover, load balancing is perfectly ensured during any simulation. On the opposite side, as far as PIC codes are concerned, the memory request associated to the grid quantities (equilibrium and fluctuating fields), which are replicated on each processor, gives rise to a bottle-neck on the scalability of physical resolution with processors: even in the limit of an infinite number of processors, in which each processor must treat a vanishingly small number of particles, the maximum resolution that can be reached is determined by the largest size of the (replicated) grid arrays that can be stored on each node. Such a constraint is so strongly penalizing that efforts in looking for alternative methods able to reduce the replicated-array memory offset are fully justified.

In many concrete situations, as a matter of fact, periodic spatial domains are considered, and the problem admits solution in terms of normal modes. This feature allows to solve the equations for the electromagnetic fields in Fourier space. Moreover, in order to get a deeper insight in the physical mechanisms underlying the observed phenomena, the analysis is often focused on the evolution of relatively few harmonics. This is the case of many studies in the field of controlled nuclear fusion research: the linear stability of Alfvén modes [7] in tokamaks in the presence of an alpha particle population produced by fusion reactions, for example, can be investigated by looking only at the presumably most unstable high mode-number modes, the evolution of each harmonic being independent of the others' one; their nonlinear saturation, in turn, comes out to be dominated by the strong wave-particle coupling, rather than by the weak mode-mode coupling, and can be analyzed restricting the spectrum to a limited number of harmonics.

Although these facts do not cause the resolution request to be relaxed (one is typically interested in the high mode-number portion of the complete fluctuation spectrum), they offer a different and more convenient path toward the target of an efficient parallelization and, hence, toward the high-resolution simulations: namely, the gridless *finite-size-particle* (FSP) approach. Instead of transforming the fields, from Fourier to real space, on a spatial grid, then interpolating such fields at each particle position to evolve particle coordinates, the gridless FSP algorithm directly transforms the fields at the particle position. Correspondingly, particle contributions to the pressure perturbation are just Fourier-transformed and summed together, instead of being collected at the nearest grid points. In such a way, the memory resources demanded by the field storage are greatly reduced.

Before the introduction of parallel computers, such a reduction was obscured by the overwhelming memory request associated to the storage of particle quantities. Moreover, the FSP method was heavily handicapped by the need of computing distinct Fourier transforms for each particle (rather than for each grid point). These reasons

motivate the prevalence, even for *few-harmonic* simulations, of the PIC approach. However, in the framework of particle-decomposition parallel simulation, the gridless FSP approach takes advantage from the distribution of the Fourier calculations among different processors and – above all – allows to remove, in fact, the bottle-neck in the scalability of the resolution with the number of processors. It is then worth to reconsider the comparison between the two methods. Aim of the present paper is to address this issue, with regard to the particle-decomposition parallelization on distributed-memory architectures of the numerical codes corresponding to the FSP and the PIC method.

Although, in principle, PIC codes could solve the field equations directly in the real space, using, e.g., finite difference methods, in the following we will assume, for the sake of simplicity, that both methods solve such equations in the Fourier space.

The paper is organized as follows. Section 2 presents the main features of particle simulation codes, while gridless FSP and PIC algorithms are compared in Section 3. Section 4 describes the most commonly used approaches (*domain* and *particle-decomposition*) for parallelization of PIC codes, whose respective merits and defects are briefly discussed, and examines the *particle decomposition* parallelization of a gridless FSP code. A comparison between results obtained by PIC and gridless FSP approaches for the Hybrid MHD-Gyrokinetic particle simulation Code [8] (HMGC) is reported in Section 5 and, with particular regard to the parallelization performances, in Section 6. Finally, a brief summary of the main ideas discussed in this paper is presented in Section 7.

2. Structure of particle simulation codes

The purpose of the particle simulation codes for the investigation of magnetically confined thermonuclear plasmas is to obtain a direct solution of the Vlasov equation (the collisionless version of the Boltzmann equation) for the particle distribution function evolving in the fluctuating electromagnetic fields, computed, in turn, by solving the selfconsistent Ampère–Maxwell set of equations. To this aim, the phase-space distribution function $F(t, Z)$ (with Z indicating the whole set of phase-space coordinates) is represented by its discretized form,

$$F(t, Z) \equiv \int \mathrm{d}Z'\, F(t, Z')\, \delta(Z - Z') \approx \sum_l \Delta_l\, F(t, Z_l) \delta(Z - Z_l), \tag{1}$$

where Δ_l is the volume element around the phase-space marker Z_l. It is immediate to show that such an expression of F satisfies Vlasov equation (here, the Einstein convention on the sum over repeated indices is adopted),

$$\frac{\mathrm{d}F}{\mathrm{d}t} \equiv \frac{\partial F}{\partial t} + \frac{\mathrm{d}Z^i}{\mathrm{d}t} \frac{\partial F}{\partial Z^i} = 0, \tag{2}$$

as far as each marker evolves in time according to the equations of motion for physical particles and the corresponding volume element $\Delta_l(t)$ evolves according to the equation

$$\frac{\mathrm{d}\Delta_l}{\mathrm{d}t} = \Delta_l(t) \left(\frac{\partial}{\partial Z^i} \frac{\mathrm{d}Z^i}{\mathrm{d}t} \right)_{t, Z_l(t)}. \tag{3}$$

Such phase-space markers can then be interpreted as the phase-space coordinates of a set of N_{part} "particles", and $F(t, Z)$ can be approximated by

$$F(t, Z) \approx \sum_{l=1}^{N_{\mathrm{part}}} w_l(t) \delta\big(Z - Z_l(t)\big), \tag{4}$$

with the weight of the lth particle being defined as

$$w_l(t) \equiv \Delta_l(t) F\big(t, Z_l(t)\big).$$

The obvious goal of particle simulation is that of obtaining from the numerical plasma the same behaviour of the physical one. The main problem in reaching such a target is represented by the impossibility, with today numerical resources, of simulating a number of particles comparable with those typical of the real plasmas. This problem can be partially by-passed by considering each simulation particle as a cluster ("macroparticle") of very many physical particles; mutual interactions between macroparticles are retained, while those between particles inside the same cluster are neglected. By identifying the charge and the mass of each simulation particle with those of the whole cluster (but still imposing that such a simulation particle moves as a physical one), all the relevant parameters (Debye length, Larmor radius, etc.) of the simulation plasma coincide with the corresponding parameters of the physical plasma, notwithstanding that the simulation-particle density is much lower than the physical-particle one [9].

Further problems, however, arise, related to the following issue. Physical plasmas are characterized by the so called "plasma condition", i.e. by the presence of a large number of particles in a Debye sphere ($n_0\lambda_D^3 \gg 1$, with n_0 being the particle density and λ_D the Debye length). It is apparent that, once the equality between simulation and physical scales has been ensured, it holds in particular for λ_D, and it is very hard, for the simulation "plasma", to satisfy the plasma condition, because of the very low value, for such a system, of the density n_0. As a consequence, the set of numerical particles is, by itself, too collisional to be considered a plasma: short range interactions between particles dominate over the long range ones, and the physical behaviour is not properly reproduced.

In the typical case, in which the physical phenomena of interest are characterized by scale lengths much larger than λ_D, it is possible to regularize collisionality by introducing a cutoff on the small scales, which does not alternate the interesting dynamics. Two classical methods able to produce such a result are the FSP and the PIC ones. The FSP method [10] consists in smoothing the singular character of the particle contribution to the distribution function – Eq. (4) – i.e. in replacing the singular spatial densities, $\delta(\mathbf{x} - \mathbf{x}_l(t))$, by shaped regular (e.g., Gaussian) ones, $S(\mathbf{x} - \mathbf{x}_l(t))$. Short-range interactions are then cut off for mutual distance shorter than the width – L_s – of the function S, whilst the relevant long-range interactions are not significantly affected. This width can be interpreted as the typical size of the numerical particles (or, more appropriately, *charge clouds*). It is immediate to show [10] that such a FSP ensemble behaves as a plasma under the much more relaxed condition $n_0 L_s^3 \gg 1$ (with $L_s \gg \lambda_D$).

The PIC method consists in computing the electromagnetic fields only at the vertices of spatial-grid cells [11], then interpolating them at the (continuous) particle positions in order to perform particle pushing, and motivates the name – *particle-in-cell* – usually assigned to the related simulation techniques. The interpolation function introduces a smoothing of the short-range interactions analogous to the FSP one, and, if L_c is the typical spacing between grid points, the plasma condition is replaced by $n_0 L_c^3 \gg 1$. Note that the two different methods are expected to yield the same qualitative results if $L_c \approx L_s$, but they can result in very different computational requirements, as it will be shown in the next section.

3. Gridless FSP versus PIC algorithms

In order to enlighten the main differences between gridless FSP and PIC simulations, we examine, in this section, the essential features of particle pushing and pressure assignment in both types of simulation. For the sake of simplicity, we refer to a one-dimensional real space.

3.1. Force and pressure calculations

The electric force acting on a point charge placed at x_l can be written as

$$F_{q_l}(x_l) = q_l E(x_l) \equiv q_l \int_L \mathrm{d}x\, \delta(x - x_l) E(x) = \int_L \mathrm{d}x\, \rho(x) E(x), \tag{5}$$

with $\rho(x) \equiv q_l \delta(x - x_l)$ being the charge density associated to the point charge, $E(x)$ the electric field, and L the length of the periodic one-dimensional system. Introducing, instead of point charges, finite-size particles (charge clouds) with a regular shape $S(x)$, in order to smooth the short-distance interactions, the charge density becomes $\rho(x) \equiv q_l S(x - x_l)$. The force can then be written, in the FSP framework, as

$$\begin{aligned} F_{q_l}(x_l) &= q_l \int_L \mathrm{d}x\, S(x - x_l) E(x) \\ &= \frac{q_l}{L} \int_L \mathrm{d}x\, S(x - x_l) \sum_{n=-\infty}^{+\infty} \mathrm{e}^{\mathrm{i}kx} \widehat{E}(k) \\ &= \frac{q_l}{L} \sum_{n=-\infty}^{+\infty} \widehat{E}(k) \widehat{S}(-k) \mathrm{e}^{\mathrm{i}kx_l}, \end{aligned} \tag{6}$$

where $\widehat{E}(k)$ and $\widehat{S}(k)$ are the Fourier transforms of the (continuous-space) electric field and the shape function $S(x)$, respectively, and the wavevector k can assume only multiple values of $k_{\min} \equiv 2\pi/L$: $k = nk_{\min}$, with $n = 0, \pm 1, \pm 2, \ldots$.

After pushing particles, the Fourier components of the FSP particle pressure (needed to close the equations for the electromagnetic fields) are obtained as follows:

$$\begin{aligned} \hat{p}(k) &= \int_L \mathrm{d}x\, \mathrm{e}^{-\mathrm{i}kx} p(x) \\ &\propto \int_L \mathrm{d}x\, \mathrm{e}^{-\mathrm{i}kx} \int \mathrm{d}v\, v^2 \sum_{l=1}^{N_{\mathrm{part}}} \Delta_l(t)\, F(t, x_l, v_l) S(x - x_l) \delta(v - v_l) \\ &= \widehat{S}(k) \sum_{l=1}^{N_{\mathrm{part}}} v_l^2 \Delta_l(t)\, F(t, x_l, v_l) \mathrm{e}^{-\mathrm{i}kx_l}, \end{aligned} \tag{7}$$

with $-\infty < k < +\infty$ and $p(x)$ and $\hat{p}(k)$ being, respectively, the (continuous-space) particle pressure and its Fourier transform. Note that a finite-size $S(x)$ corresponds to a limited width of its Fourier transform in the k space and, hence, a smoothing of the short-distance (large $|k|$) interactions, which are otherwise exaggerated because of the relatively small number of simulation particles. If the equations for the fields are solved in the k space, the same effect can be obtained, more directly, by imposing an artificial cutoff on k, i.e. by considering a band-limited k space, $|k| < |k_{\mathrm{sup}}|$, with $|k_{\mathrm{sup}}| \ll 2\pi/\lambda_D$. If k_{phys} is the largest wave vector component one is, for physical reasons, interested to retain, it is sufficient to choose $|k_{\mathrm{sup}}| = |k_{\mathrm{phys}}|$. In such a case the expression for the force could be replaced by

$$F_{q_l}(x_l) = \frac{q_l}{L} \sum_{|n| \leqslant |k_{\mathrm{sup}}/k_{\min}|} \widehat{E}(k) \mathrm{e}^{\mathrm{i}kx_l}, \tag{8}$$

which corresponds to adopting, in Eq. (6), a step shape factor $\widehat{S}(k)$:

$$\widehat{S}(k) = \begin{cases} 1 & \text{if } |k| \leqslant |k_{\mathrm{sup}}|, \\ 0 & \text{if } |k| > |k_{\mathrm{sup}}|. \end{cases} \tag{9}$$

In the PIC approach, we introduce a spatial grid, X_j with $j = 1, \ldots N_{\mathrm{cell}}$. The force acting on the lth particle is then computed as

$$F_{q_l}(x_l) = q_l \Delta x \sum_{j=1}^{N_{\mathrm{cell}}} S_{\mathrm{int}}(X_j - x_l) E_j, \tag{10}$$

where $\Delta x \equiv L/N_{\text{cell}}$, $S_{\text{int}}(x)$ is the interpolation function, satisfying the condition

$$\Delta x \sum_{j=1}^{N_{\text{cell}}} S_{\text{int}}(X_j - x) = 1, \tag{11}$$

and E_j is the electric field at the jth grid point, which can be expressed in terms of its Fourier transform, $\overline{E}(k)$, according to

$$E_j = \frac{1}{L} \sum_{n=0}^{N_{\text{cell}}-1} \mathrm{e}^{\mathrm{i}kX_j} \overline{E}(k).$$

Unfortunately the use of a grid risks to significantly affect the dynamics even in the domain of physical interest. Indeed, from Eq. (10) we can write

$$\begin{aligned} F_{q_l}(x_l) &= \frac{q_l}{N_{\text{cell}} L} \sum_{j=1}^{N_{\text{cell}}} \sum_{n=-\infty}^{+\infty} \mathrm{e}^{\mathrm{i}k(X_j - x_l)} \widehat{S}_{\text{int}}(k) \sum_{n'=0}^{N_{\text{cell}}-1} \mathrm{e}^{\mathrm{i}k'X_j} \overline{E}(k') \\ &= \frac{q_l}{L} \sum_{n=-\infty}^{+\infty} \mathrm{e}^{\mathrm{i}kx_l} \widehat{S}_{\text{int}}(-k) \overline{E}(k), \end{aligned} \tag{12}$$

with $\widehat{S}_{\text{int}}(k)$ being the Fourier transform of $S_{\text{int}}(x)$, and $k' \equiv n' k_{\min}$. Comparing this expression with Eq. (8), obtained in the FSP case, we see that two distinct problems arise. The first one is related to the differences between the Fourier transforms of the gridded electric field and the continuous-space one. Unlike $\widehat{E}(k)$, $\overline{E}(k)$ is a periodic function of k, with period $k_g \equiv 2\pi/\Delta x$. As wavevectors that differ for multiples of k_g cannot be distinguished from the grid point of view, $\overline{E}(k)$ gets contribution, for each of the wavevectors belonging to the first Brillouin zone, $|k| \leqslant \pi/\Delta x$, from all the aliases k_p, with $k_p \equiv k - pk_g$ and $p = 0, \pm 1, \pm 2, \ldots$. For example, even if the gridded electric field, E_j, were computed exactly in terms of the continuous-space field $E(x)$ as $E_j = E(X_j)$ (which, in practice, is not the case), the relationship between $\overline{E}(k)$ and $\widehat{E}(k)$ would be given by

$$\begin{aligned} \overline{E}(k) &\equiv \frac{L}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} E_j = \frac{L}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} E(X_j) \\ &\equiv \frac{1}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} \sum_{n'=-\infty}^{+\infty} \mathrm{e}^{\mathrm{i}k'x} \widehat{E}(k') = \sum_{p=-\infty}^{+\infty} \widehat{E}(k_p). \end{aligned}$$

An analogous coupling of the aliases through the grid can be found by examining the Fourier transform of the gridded particle pressure, obtained by assignment of particle contributions to the nearest grid points, as follows:

$$\begin{aligned} \overline{p}(k) &= \frac{L}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} p_j \\ &= \frac{L}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} \int_L \mathrm{d}x \, S_{\text{int}}(X_j - x) p(x) \\ &\propto \frac{L}{N_{\text{cell}}} \sum_{j=1}^{N_{\text{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} \int_L \mathrm{d}x \, S_{\text{int}}(X_j - x) \end{aligned}$$

$$\times \int \mathrm{d}v\, v^2 \sum_{l=1}^{N_{\mathrm{part}}} \Delta_l(t)\, F(t, x_l, v_l)\delta(x - x_l)\delta(v - v_l)$$

$$= \frac{L}{N_{\mathrm{cell}}} \sum_{j=1}^{N_{\mathrm{cell}}} \mathrm{e}^{-\mathrm{i}kX_j} \sum_{l=1}^{N_{\mathrm{part}}} v_l^2 \Delta_l(t)\, F(t, x_l, v_l) S_{\mathrm{int}}(X_j - x_l), \tag{13}$$

where $|k| \leqslant \pi/\Delta x$, p_j is the particle pressure at the jth grid point, and we have adopted an assignment function coincident with $S_{\mathrm{int}}(x)$ in order to preserve momentum conservation [3]. From Eq. (13) we see that the relationship between $\bar{p}(k)$ and $\hat{p}(k)$ is expressed by

$$\bar{p}(k) = \sum_{p=-\infty}^{+\infty} \widehat{S}_{\mathrm{int}}(k_p)\hat{p}(k_p).$$

The effect of aliases on $\bar{p}(k)$ (and, hence, on $\overline{E}(k)$) can be controlled up to some degree, by choosing $S_{\mathrm{int}}(x)$ in such a way to ensure a fast decrease of $\widehat{S}_{\mathrm{int}}(k)$ as $k\Delta x \to \infty$.

The second problem regards the factor $\widehat{S}_{\mathrm{int}}(-k)$ that appears in Eq. (12). Comparing such equation with Eq. (8), we see that, in order to avoid further distortions of the dynamics in the domain of physical interest, the condition

$$\left|1 - \widehat{S}_{\mathrm{int}}(k)\right| \ll 1 \tag{14}$$

has to be satisfied for each k such that $|k| < |k_{\mathrm{phys}}|$.

Both problems could be, in principle, overcome by (a) adopting an interpolation function $S_{\mathrm{int}}(x)$ having a step Fourier transform, like that of Eq. (9), with $k_{\mathrm{sup}} \equiv \pi/\Delta x$, in order to suppress alias effects and avoid distortions in the first Brillouin zone ($|k| < \pi/\Delta x$), and (b) taking $\Delta x \equiv \pi/k_{\mathrm{phys}}$ – i.e., $N_{\mathrm{cell}} = 2k_{\mathrm{phys}}/k_{\mathrm{min}}$ – in order to make such a zone coincident with the domain of physical interest. Unfortunately, such an interpolation function would be given by $S_{\mathrm{int}}(x) \equiv 1/L \sum_{|n| \leqslant N_{\mathrm{cell}}/2} \mathrm{e}^{-\mathrm{i}kx}$, which cannot be used, in practice, because the term $S_{\mathrm{int}}(X_j - x_l)$ in Eq. (10) would be different from zero for every grid point and its calculation would involve an expensive sum over the k space. The convenience of a PIC approach relies, instead, on involving, for a given particle, only a limited number of nearest grid points in the interpolation and adopting simple interpolation functions.

For realistic few-point interpolation functions, Eq. (9) is not verified. Eq. (11) implies [3] that

$$\widehat{S}_{\mathrm{int}}(0) = 1 \tag{15}$$

and

$$\widehat{S}_{\mathrm{int}}(pk_g) = 0, \tag{16}$$

for $p = \pm 1, \pm 2, \ldots$, but neither the flatness of $\widehat{S}_{\mathrm{int}}(k)$ in the first zone of Brillouin nor its vanishing outside that zone are recovered. This is the case, for example, of the linear interpolation function,

$$S_{\mathrm{int}}(x) = \begin{cases} \frac{1}{\Delta x}\left(1 - \frac{|x|}{\Delta x}\right) & \text{if } |x| \leqslant \Delta x, \\ 0 & \text{if } |x| > \Delta x, \end{cases} \tag{17}$$

with Fourier transform

$$\widehat{S}_{\mathrm{int}}(k) = \left[\frac{\sin(k\Delta x/2)}{k\Delta x/2}\right]^2. \tag{18}$$

Eqs. (15) and (16), along with the accuracy condition, Eq. (14), require, for the PIC case,

$$k_{\mathrm{phys}} \ll k_g. \tag{19}$$

Such a condition also guarantees that the alias effects poorly affects the physically relevant dynamics, because the aliases of modes with $|k| \leqslant |k_{\mathrm{phys}}|$ fall in region where $\widehat{S}_{\mathrm{int}}$ is close to zero.

Note that, if the Fast Fourier Transform (FFT) is adopted, we are forced, for a given spatial grid, to retain all the harmonics with $|k| \leqslant |k_{\rm sup}|$, with $k_{\rm sup} = N_{\rm cell}k_{\rm min}/2 = \pi/\Delta x \gg k_{\rm phys}/2$. Using simple Fourier Transform (FT) would allow to decouple the number of harmonics from the number of cells. Then, even using a grid as fine as required by Eq. (19), we could cutoff our Fourier space, when solving the equations for the field, at $k_{\rm sup} = k_{\rm phys}$, as done in obtaining Eq. (8). We will come back on this point in the following, when evaluating the computational loads associated to each method.

3.2. Computational loads for FSP and PIC simulations

Let us estimate the computational loads, in terms of memory and number of operations, associated to FSP and PIC methods. From Eqs. (6) and (7), we see that an important feature of the gridless FSP method is represented by the coupling between particle (l) and Fourier space (k) indices, both in the calculation of forces and pressure. This means that the sum over Fourier components must be performed for every single particle, when computing forces, and the sum over particles has to be repeated for each Fourier harmonic, when computing pressure. The number of operations per time step and the memory requirement are then approximately given by:

$$N_{\rm op}^{\rm FSP} \approx f(N_{\rm harm}) + n_{\rm FT}N_{\rm harm}N_{\rm part}, \tag{20}$$

$$M^{\rm FSP} \approx m_{\rm harm}N_{\rm harm} + m_{\rm part}N_{\rm part}. \tag{21}$$

Here $N_{\rm harm}$ is the number of Fourier harmonics retained in the simulation; $f(N_{\rm harm})$ is an increasing function of $N_{\rm harm}$, which corresponds to the number of operations required to update the electromagnetic fields in the Fourier space, and whose precise dependence is determined by the structure of the problem and by the algorithm used by the field solver; $n_{\rm FT}$ is the number of operations needed to compute each addendum in the sum required by the Fourier transform (and anti-transform). Moreover, $m_{\rm harm}$ and $m_{\rm part}$ are the amounts of memory needed to store, respectively, a single harmonic of the complete set of Fourier-space fields and the phase-space coordinates and weights for each particle.

Note that, in the one-dimensional case here considered for simplicity, $N_{\rm harm} \leqslant k_{\rm phys}/k_{\rm min}$, the disequality holding in the strong sense in the ipothesis that, for simplified investigations, only few harmonics are retained (the larger k values being $k_{\rm phys}$). In the general multi-dimensional case, $N_{\rm harm}$ will be less or equal than the number of harmonics, $N_{\rm phys}$, contained in the multi-dimensional portion of the spectrum delimited by the maximum components of physical interest.

Regarding the PIC method, Eq. (10), which can be written as

$$F_{q_l}(x_l) = \frac{q_l}{N_{\rm cell}} \sum_{j=1}^{N_{\rm cell}} S_{\rm int}(X_j - x_l) \sum_{n=0}^{N_{\rm cell}-1} {\rm e}^{{\rm i}kX_j} \overline{E}(k),$$

and Eq. (13) show that, differently from the FSP case, particle and Fourier space indices are decoupled. The fields at the grid points may then be calculated only once per time step, regardless of the number of particles; in the same way, particle contributions to the grid-point pressure may be summed together only once, regardless of the number of Fourier modes. This requires a number of operations per time step and an amount of memory approximately given by

$$N_{\rm op}^{\rm PIC} \approx f\left(\frac{N_{\rm cell}}{2}\right) + n_{\rm FFT}N_{\rm cell}\log_2 N_{\rm cell} + n_{\rm int}N_{\rm part}, \tag{22}$$

$$M^{\rm PIC} \approx m_{\rm harm}\frac{N_{\rm cell}}{2} + m_{\rm cell}N_{\rm cell} + m_{\rm part}N_{\rm part}, \tag{23}$$

where $n_{\rm FFT}$ is a coefficient related to the FFT algorithm, $n_{\rm int}$ is the number of operations required for the interpolation/assignment between grid and particle positions, $m_{\rm cell}$ is the amount of memory needed to store the

real-space fields at each grid point, and the fact that the number of harmonics equals, within the FFT approach, $N_{\rm cell}/2$ has been used.

Two facts deserve to be noted. First, the PIC approach allows to adopt the FFT algorithm, with a very favorable dependence on the number of harmonics of the number of operations related to the computation of the fields on the grid. Second, because of the sake of accuracy, such number of harmonics is constrained, as stated in Section 3.1, to be much greater than $N_{\rm phys}$ (we refer hereafter to the general multi-dimensional case). Both these facts will assume relevance in comparing the respective merits of the FSP method and the PIC one.

In the framework of serial simulations, as far as $n_{\rm FT}N_{\rm harm} \gtrsim n_{\rm int}$, the gridless FSP method is much more expensive than the PIC one, without presenting any significant advantage in terms of lower memory requests. This can be easily seen by considering that in Eqs. (20)–(23) both computation and memory requests are dominated by the terms proportional to $N_{\rm part}$, as (a) the term related to the updating of the fields in the Fourier space is typically negligible with respect to the others, (b) $N_{\rm harm} \leqslant N_{\rm cell}$, and (c) for PIC codes, the number of particle per cell, $N_{\rm ppc} \equiv N_{\rm part}/N_{\rm cell}$, satisfies the condition $N_{\rm ppc} \approx n_0 L_c^3 \gg 1$. This motivates the large diffusion of PIC codes and the poor consideration in which gridless FSP simulation was taken in the past. A different trend can however be justified by the utilization of parallel computers. We will consider this point in detail at the end of the following section.

4. Parallelization of particle simulation codes

The need for high resolution and, consequently, for large number of particles has motivated a large effort, in the last years, in porting particle codes on parallel computers. In this paper we restrict our analysis to the case of distributed-memory architectures.

4.1. PIC codes

Let us start considering standard PIC codes. The typical structure of such codes can be summarized by the following three items for each time step:

(1) a field solver computes the fluctuating electromagnetic fields at the $N_{\rm cell}$ grid points of the spatial domain;
(2) phase-space coordinates and weights of the particles are evolved by using the fluctuating fields interpolated at each particle position;
(3) a certain momentum of the particle distribution function (e.g., pressure) is updated, on the grid, in order to close the field equations for the next time step, by distributing the contribution of each particle among the neighbor grid points.

Typical situations of interest correspond to a moderate number of relevant fields and particle variables, so that the plasma condition $N_{\rm ppc} \gg 1$ and Eqs. (22) and (23) cause the main purpose of the parallelization of PIC codes to be that of distributing the computational requests related to the particle population among several different processors.

The most widely explored approach to such a parallelization is represented by the so-called domain decomposition [4]: different portions of the spatial grid are assigned to different computational nodes, along with the particles therein contained. A large amount of the computational work corresponding to each of the three above-mentioned elementary items can be efficiently distributed among the processors. In this way, both the number of operations per time step and the memory space required to each computational node are, roughly speaking, reduced (in an average sense) by a factor equal to the number of processors, $n_{\rm proc}$. Neglecting corrections due to inter-processor communication, such quantities are indeed given by the following expressions (we assume to use parallel FFT routines):

$$N_{\rm op\ d.d.}^{\rm PIC} \approx f\left(\frac{N_{\rm cell}}{2}\right) + \frac{1}{n_{\rm proc}}\left(n_{\rm FFT}N_{\rm cell}\log_2 N_{\rm cell} + n_{\rm int}N_{\rm part}\right), \tag{24}$$

and

$$M_{\text{d.d.}}^{\text{PIC}} \approx m_{\text{harm}} \frac{N_{\text{cell}}}{2} + \frac{1}{n_{\text{proc}}} \left(m_{\text{cell}} N_{\text{cell}} + m_{\text{part}} N_{\text{part}} \right). \tag{25}$$

Inter-processor communication are however necessary, e.g., to transfer particle data from one computational node to another one when some particle moves from one portion of the grid to a different one; this particle migration can result in severe load-balancing problems and the above expressions, which state that such method is very efficient, can be, in practice, invalidated. In the limiting case, all the particles could indeed move to the grid portion assigned to a certain processor, leaving all the other processors under-utilized and recovering the original single-processor situation. In order to avoid such a possibility, a dynamical redistribution of grid and particle quantities is required. This can make the parallel implementation of a serial code very complicate, although satisfactory results have been recently obtained by using object-oriented programming [5].

The search for simple parallel programming has motivated an alternative approach to parallelization: the one based on particle decomposition [6]. It consists in statically distributing the particle population among processors, while replicating the data relative to grid quantities. Before updating the electromagnetic fields, at each time step, partial contributions to particle pressure coming from each portion of the population must be summed together. It is apparent that load balancing is automatically enforced, because no particle has to be transferred from one processor to another. As a consequence, the implementation of parallelization is, in principle, almost straightforward. On the opposite side, grid calculations do not take advantage, with regard both to the number of operations and the memory requests, from such a parallelization, because each processor has to handle the whole spatial domain. Neglecting the corrections due to the need for communicating partial-pressure data, the computational load on each node is indeed given, in this case, by

$$N_{\text{op p.d.}}^{\text{PIC}} \approx f\left(\frac{N_{\text{cell}}}{2} \right) + n_{\text{FFT}} N_{\text{cell}} \log_2 N_{\text{cell}} + n_{\text{int}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{26}$$

and

$$M_{\text{p.d.}}^{\text{PIC}} \approx m_{\text{harm}} \frac{N_{\text{cell}}}{2} + m_{\text{cell}} N_{\text{cell}} + m_{\text{part}} \frac{N_{\text{part}}}{n_{\text{proc}}}. \tag{27}$$

Particle decomposition comes out to be very efficient as far as the computational load related to the particle population dominates, for each processor, over the one related to the grid. This condition corresponds to require a large average number of particles per cell on each processor, and it is surely satisfied for moderately parallel machines. However, for $n_{\text{proc}} \gtrsim N_{\text{ppc}}$, it breaks down, and the computational effort of each processor is mainly devoted to the replicated calculation of fields on the grid.

Apart from efficiency problems, the implementation of the code on a very large number of processors does not allow one to reach high spatial resolution levels, due to the offset in the memory requests represented by the field data: the highest spatial resolution that can be reached in a simulation without requiring *memory paging* depends on the Random Access Memory (RAM) resources of the single computational node, whereas increasing the number of processors only allows to increase the average number of particle per cell and, hence, the velocity-space resolution. This limitation comes out to be even more stringent than the efficiency one. At a certain extent, such a constraint can be relaxed, at the expenses of computation loads and parallelization efficiency, by resorting to the simple FT algorithm instead of the FFT one. As stated in Section 3.1, this allows to decouple the number of Fourier harmonics retained, N_{harm}, from N_{cell}. It is then possible to cut the Fourier spectrum immediately above the wave numbers of physical interest (possibly retaining even a lesser number of harmonics, $N_{\text{harm}} \leqslant N_{\text{phys}}$), without renouncing to the spatial resolution that yields an accurate interpolation ($N_{\text{cell}} \gg N_{\text{phys}}$). In this case Eq. (27) would be replaced by

$$M_{\text{p.d. FT}}^{\text{PIC}} \approx m_{\text{harm}} N_{\text{harm}} + m_{\text{cell}} N_{\text{cell}} + m_{\text{part}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{28}$$

giving a reduction of the memory offset of up to a factor $(1 + m_{\rm harm}/2m_{\rm cell})$. At the same time, however, the approximate expression for the number of operations would become

$$N^{\rm PIC}_{\rm op\ p.d.\ FT} \approx f(N_{\rm harm}) + n_{\rm FT} N_{\rm harm} N_{\rm cell} + n_{\rm int} \frac{N_{\rm part}}{n_{\rm proc}}. \tag{29}$$

It can be seen that the replicated portion of the computation is larger in this case than in the FFT one, unless

$$n_{\rm FT} N_{\rm harm} \lesssim n_{\rm FFT} \log_2 N_{\rm cell}. \tag{30}$$

The last condition can be satisfied only if low values of $N_{\rm phys}$ and/or $N_{\rm harm}$ are considered. Note, however, that today limitations on single-node memory resources can impose, in practice, an upper limit on the maximum achievable $N_{\rm cell}$ and – hence – $N_{\rm phys}$ such that Eq. (30) is, in fact, satisfied.

4.2. Gridless FSP codes

A much more sensible contribution to the relaxation of the memory constraint can be obtained by resorting to the gridless FSP simulation, so by-passing the introduction of a spatial grid. A particle-decomposition approach to the parallelization of a gridless FSP code would indeed bring to the following computational loads on each node:

$$N^{\rm FSP}_{\rm op\ p.d.} \approx f(N_{\rm harm}) + n_{\rm FT} N_{\rm harm} \frac{N_{\rm part}}{n_{\rm proc}}, \tag{31}$$

and

$$M^{\rm FSP}_{\rm p.d.} \approx m_{\rm harm} N_{\rm harm} + m_{\rm part} \frac{N_{\rm part}}{n_{\rm proc}}. \tag{32}$$

By comparing Eq. (32) with Eqs. (27) and (28), it can be seen that, both the reduction of the term proportional to $m_{\rm harm}$ (already obtained, in the PIC-FT framework) and the complete elimination of the term proportional to $m_{\rm cell}$ make the gridless FSP method much more convenient, in the framework of the particle-decomposition approach, than the PIC one.

Also concerning the parallelization efficiency (the ratio between the speed-up factor and the number of processors), the FSP tends to prevail on the PIC method, because, though the FSP method requires back and forth Fourier transforms being executed for each particle, these calculations are distributed among the processors, unlike the replicated PIC grid calculations. Large efficiency values can indeed be obtained insofar the first term on the right-hand side of Eq. (31) is negligible in comparison with those proportional to $N_{\rm part}$ – which can be the case, now, even for a large number of processors. At the same time, of course, the positive feature of an automatic load balancing, typical of the particle-decomposition parallelization, is preserved.

Regarding the absolute CPU-time performance, a comparison between Eqs. (26) and (31) shows that the FSP code reaches better results than the PIC one for

$$N_{\rm harm} \lesssim \frac{n_{\rm FFT}}{n_{\rm FT}} \frac{n_{\rm proc}}{N_{\rm ppc}} \log_2 N_{\rm cell}, \tag{33}$$

which gives the concept of "few harmonics" a quantitative specification. Here we are neglecting the f terms in both equations and assuming that $n_{\rm int} \ll n_{\rm FT} N_{\rm harm}$. Note that in the FSP case neither $N_{\rm cell}$ nor $N_{\rm ppc}$ have, by themselves, any meaning. However, we can interpret $N_{\rm cell}$ as the number of cells that would be necessary to accurately describe, in the PIC framework, the same modes we investigate by the gridless code. In the following, we will then also refer to $N_{\rm ppc}$ as to an average number of particles "per cell", looking at $N_{\rm part}$ as the product of certain levels of spatial and velocity-space resolution (measured by $N_{\rm cell}$ and $N_{\rm ppc}$, respectively).

If the simple FT algorithm were adopted, the above condition would be replaced, after comparing Eqs. (29) and (31), by the following one:

$$n_{\rm proc} \gtrsim N_{\rm ppc}. \tag{34}$$

Note, finally, that different conclusions would be reached if the parallelization of the PIC code were based on a dynamically load-balanced domain decomposition. In such a case, assuming that the efforts needed by the domain remapping can be neglected, Eqs. (31) and (32) has to be compared with Eqs. (24) and (25), respectively. The FSP code prevails on the PIC one with respect to the memory requirements (as $N_{\rm harm} \leqslant N_{\rm phys} \ll N_{\rm cell}$). On the opposite, the PIC CPU-time performances comes out to be much better than the FSP ones, as far as the computation is dominated by the distributed portion. This is generally true, unless the condition

$$n_{\rm proc} > \frac{n_{\rm FT} N_{\rm harm} N_{\rm part}}{f(N_{\rm harm})} \tag{35}$$

is satisfied, which however makes the parallel codes very inefficient.

5. The FSP Hybrid MHD-Gyrokinetic Code

In this section we present the results of the implementation of a particle-decomposition parallel gridless FSP version of the Hybrid MHD-Gyrokinetic Code (HMGC), a code for the investigation of Alfvénic turbulence in tokamak plasmas, originally implemented in a PIC version [8]. The parallelization of the PIC version of HMGC has been presented in Ref. [6]. The PIC code solves the coupled sets of reduced MHD equations [12,13] for the fluctuating electromagnetic fields and gyrokinetic equations of motion [14] for collision-free energetic ions. The physical domain is represented by a three-dimensional toroidal grid, for which quasi-cylindrical coordinates are adopted (see Fig. 1): the minor-radius coordinate, r, and the poloidal and toroidal angles, ϑ and φ, respectively. The field solver uses finite differences in the minor radius direction and Fourier expansion in the poloidal and toroidal directions. Field values at each particle position are obtained by trilinear interpolation of the fields at the vertices of the cell the particle belongs to. The interpolation function is then given by

$$S(\mathbf{x}) = S_r(r) S_\vartheta(\vartheta) S_\varphi(\varphi), \tag{36}$$

with

$$S_\zeta(\zeta) \propto 1 - \frac{|\zeta|}{\Delta\zeta}, \tag{37}$$

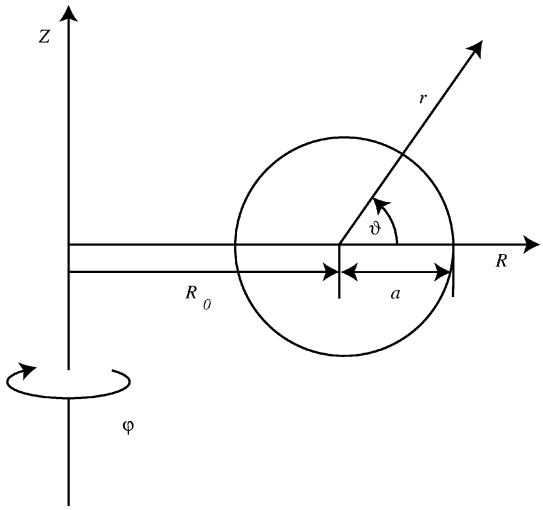

Fig. 1. Toroidal coordinate system (r, ϑ, φ) for a tokamak plasma equilibrium with major radius R_0 and minor radius a.

and $\zeta \equiv r, \vartheta, \varphi$. The same functional form is adopted for the weight function in order to distribute the pressure contribution of each particle among the vertices of the cell.

The FSP version of the code eliminates the $N_\vartheta \times N_\varphi$-point grid in the ϑ and φ directions, and introduces finite sizes for particles along the same directions. The corresponding Fourier variables are $k_\vartheta \equiv m/r$ and $k_\varphi \equiv n/R$, where m and n are respectively the poloidal and toroidal number, and R is the major-radius coordinate of the torus. The N_r-point radial grid is instead maintained (so that this FSP code is not a "gridless" one, in an absolute sense), because the domain is not periodic in that direction.

The expression for the force acting on the lth particle is obtained, in the FSP HMGC case, by a combination of the two approaches described in Section 3:

$$\mathbf{F}_{q_l}(r_l, \vartheta_l, \varphi_l) = q_l \sum_j S_r(r_l - r_j) \sum_{m,n} \widehat{\mathbf{E}}_{j;m,n} \widehat{S}_\vartheta(-m) \widehat{S}_\varphi(-n) \mathrm{e}^{\mathrm{i}(m\vartheta_l + n\varphi_l)}, \tag{38}$$

with $\widehat{\mathbf{E}}_{j;m,n}$ being the (m,n) Fourier component of the electric field, calculated (by the field solver) at the jth radial grid point, and

$$\widehat{S}_\vartheta(m) = \frac{\sin^2(m\Delta\vartheta/2)}{(m\Delta\vartheta/2)^2}, \tag{39}$$

$$\widehat{S}_\varphi(n) = \frac{\sin^2(n\Delta\varphi/2)}{(n\Delta\varphi/2)^2}. \tag{40}$$

Here we maintain the shape factors, although they could be suppressed, as it was explained before, by introducing an explicit cutoff in the Fourier space. The specific choice of such factors makes the FSP version of the code exactly corresponding to the PIC one.

Analogously, the (m,n) Fourier component of the particle pressure at the same radial grid point is given by

$$\hat{p}_{j;m,n} \propto \widehat{S}_\vartheta(m) \widehat{S}_\varphi(n) \sum_{l=1}^{N_{\mathrm{part}}} S_r(r_l - r_j) v_l^2 \Delta_l(t)\, F(t, \mathbf{x}_l, \mathbf{v}_l) \mathrm{e}^{-\mathrm{i}(m\vartheta_l + n\varphi_l)}. \tag{41}$$

The Alfvén modes, which constitute the main object of the investigation performed by HMGC, are typically characterized [7], for a given value of the toroidal number n, by the coupling of several poloidal harmonics, with sharper radial structure the higher the number n is, and poloidal numbers such that

$$0 \leqslant m \leqslant nq(a) + \tfrac{1}{2}. \tag{42}$$

Here $q(r) \equiv r B_\varphi / R_0 B_\vartheta$ is the so-called *safety factor*, a and R_0 are the minor and the major radius of the torus, respectively, B_φ is the toroidal magnetic field component and B_ϑ is the poloidal one. As a consequence, both poloidal and radial resolution requests (N_ϑ and N_r, respectively) linearly increase with increasing n, as it happens – of course – for the toroidal resolution, (N_ϕ).

In Ref. [15] the PIC version of the code has been applied to the investigation of the linear stability and the nonlinear saturation of Alfvén modes in tokamaks, in the presence of an energetic-ion population. The main results obtained in that analysis can be summarized as follows: as the energetic-ion pressure increases above a certain threshold, depending on the toroidal number n, the fast-growing Energetic Particle Mode (EPM) becomes unstable [16]. The EPM saturates because of a sudden displacement of a large part of the energetic particles along the minor radius. Such a displacement can, in principle, greatly enhance the energetic-particle losses, and makes the determination of the threshold for EPM destabilization an important issue in the theoretical research on reactor-relevant plasmas.

Fig. 2 shows the growth rate of the EPM's obtained by HMGC simulations at different values of n and β_H, with β_H being the ratio between the energetic-particle pressure and the magnetic one. The case of a plasma with aspect ratio $R_0/a = 10$ and energetic-particle Larmor radius ρ_H such that $\rho_H/a = 0.01$ is considered here. A density profile of the form $\exp[-(r^2/L_n^2)^{\alpha_n}]$ has been assumed for the energetic ions, with $a^2/L_n^2 = 2$ and $\alpha_n = 2$. The

results obtained by the PIC and the FSP versions of HMGC are compared. Slight quantitative differences between the results obtained by the two methods can be traced back to the different algorithms adopted, but a substantial agreement is found between the two approaches. Fig. 3 shows the output of nonlinear PIC (a) and FSP (b) HMGC simulations, for $n = 1$, $\beta_H = 0.04$ and the other parameters as in Fig. 2. The behaviour of the total (magnetic plus kinetic) fluctuating energy associated to different poloidal harmonics versus the time t is reported, along with the contour plot of the energetic-particle line-density in the (t, r) plane: saturation occurs in coincidence with a sudden displacement of the source of instability represented by the energetic-particle population. Also in this respect the two methods yield almost coincident results.

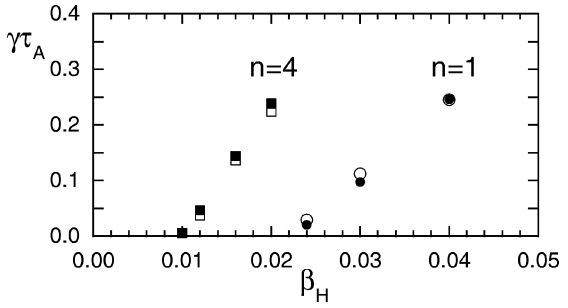

Fig. 2. Growth rate of Energetic Particle Modes, normalized to the inverse of the Alfvén time τ_A, obtained, at different values of n and β_H, by PIC (full symbols) and FSP (empty symbols) HMGC simulations.

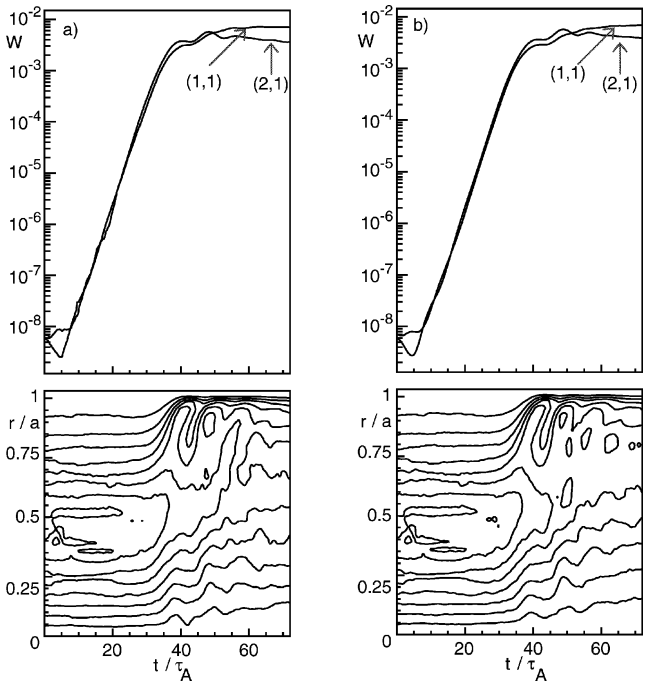

Fig. 3. Total fluctuating energy associated to different (m, n) harmonics versus time (upper) and contour plot of the energetic-particle line-density in the (t, r) plane (lower), for nonlinear PIC (a) and FSP (b) HMGC simulations. Here $\beta_H = 0.04$.

6. Parallelization results

The FSP HMGC has been parallelized, according to the same strategy (namely, the *particle-decomposition* one) adopted for the corresponding PIC version of the code [6], within the High Performance Fortran (HPF) framework [17], and tested on a 16-node IBM SP parallel system, by using the IBM *xlhpf* compiler [18] (an optimized native compiler for IBM SP systems). Some obvious differences exist between the two parallel codes, related, for example, to the fact that, as explained in Sections 3 and 4 in the FSP case the Fourier components of the pressure are directly calculated as a sum over the particle population, rather than Fourier transforming the corresponding sum computed at the spatial grid points. [1]

Although all the arguments discussed in the previous sections maintain their qualitative validity, it is worth specifying the expressions given in Eqs. (31) and (32) for such particular case [19]:

$$N_{\text{op p.d.}}^{\text{FSP}} \approx N_r \hat{f}(N_{\text{harm}}) + n_{\text{FT}} N_{\text{harm}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{43}$$

$$M_{\text{p.d.}}^{\text{FSP}} \approx \widehat{m}_{\text{harm}} N_r N_{\text{harm}} + m_{\text{part}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{44}$$

where $\hat{f}$ and $\widehat{m}_{\text{harm}}$ are, respectively, the number of operations needed to update the electromagnetic field or the memory needed to store a single m, n-harmonic of such field at each point of the radial grid. In principle, memory resources of each computational node could still be saturated by the offset associated to the first term in Eq. (44). Anyway, such an offset is in fact negligible, as its mode-number dependence, for a number of retained harmonics proportional to n, is quadratic. In the PIC case, if the FFT approach were adopted, the offset would be yielded by both the first two terms on the right-hand side of Eq. (27) and would scale as $N_r N_\vartheta N_\varphi$ (i.e. as n^3). As stated in Section 4.1, such an offset can be reduced by a significant factor, by adopting a less efficient FT approach, which in fact depresses the first of the two terms. Here we will consider only such FT approach to PIC simulation, in order to avoid limiting too much the range of n in which the comparison between the PIC and FSP methods can be performed. The expressions given in Eqs. (29) and (28), respectively, become, for the particular case considered here:

$$N_{\text{op p.d.FT}}^{\text{PIC}} \approx N_r \hat{f}(N_{\text{harm}}) + n_{\text{FT}} N_{\text{harm}} N_r N_\vartheta N_\varphi + n_{\text{int}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{45}$$

$$M_{\text{p.d.FT}}^{\text{PIC}} \approx \widehat{m}_{\text{harm}} N_r N_{\text{harm}} + m_{\text{cell}} N_r N_\vartheta N_\varphi + m_{\text{part}} \frac{N_{\text{part}}}{n_{\text{proc}}}. \tag{46}$$

We will treat the more efficient (but more memory-demanding) FFT approach in the Appendix.

With the level of precision adopted in our simulations to describe the spatial structure of the modes, the memory (in Megabytes) needed to store the field arrays for the FSP and the PIC version of HMGC, respectively, scales with the toroidal number n according to the following expressions:

$$M_{\text{field}}^{\text{FSP}} \approx 0.023 n^2, \tag{47}$$

and

$$M_{\text{field}}^{\text{PIC}} \approx 0.37 n^3. \tag{48}$$

Here we consider linear simulations with $q(0) = 1.1$, $q(a) = 1.9$ and modes characterized by a single toroidal number, and those poloidal numbers that satisfy the coupling condition, Eq. (42). It can be appreciated that the PIC approach is in fact penalized both by the n dependence and the proportionality coefficient. As such arrays

[1] The auxiliary array, introduced in Ref. [6] with an extra dimension related to the number of processors, represents, in this case, the partial contribution (yielded by those particles that reside on a certain node) to the Fourier transform of the pressure, rather than to the pressure itself.

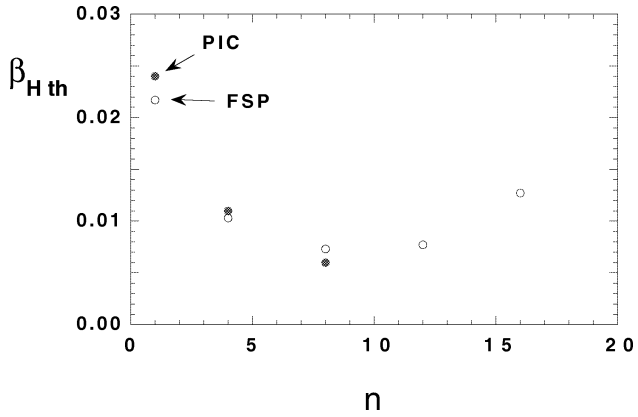

Fig. 4. Threshold β_H values for the linear EPM destabilization at different values of the toroidal number of the mode. Results obtained by FSP (empty symbols) and PIC (full symbols) simulations are compared.

are replicated on the different processors, the saturation of memory space (and the need for *memory paging*) is reached, in the PIC case, at lower values of n and, thus, at lower space resolution. The whole memory needed to store the distributed arrays associated to particle quantities (corresponding to the last term in Eqs. (46) and (44)) scales instead, for both versions of HMGC, according to the following expression:

$$M_{\text{part}} \approx 0.23 N_{\text{ppc}} n^3, \tag{49}$$

with the cubic dependence on n being associated to the "number of cells", $N_r N_\vartheta N_\varphi \equiv N_{\text{part}}/N_{\text{ppc}}$.

In Fig. 4, the β_H threshold for the linear EPM destabilization is reported versus the toroidal number of the mode n. Results obtained by FSP (empty symbols) and PIC (full symbols) simulations are compared. The relevant point to be noted here is the fact that, for a given size of the computational node RAM, parallel FSP simulations allow to push the investigation up to higher values of n. The importance of this possibility in order to get a sufficient insight in the physical behaviour of the system of interest is apparent from Fig. 4, which shows that the decrease of the β_H threshold with increasing n is restricted to values $n \lesssim 10$, so indicating the existence of a maximum β_H for safe tokamak operations. Such a feature could not be appreciated in a previous study [15] based on the PIC approach. The efficiency η, defined as the speed-up factor divided by the number of processors, is plotted, in Fig. 5, versus $n_{\text{proc}}/N_{\text{ppc}}$. Simulations retaining modes with a single toroidal number n are considered here. The results obtained by FSP HMGC are compared with those obtained by PIC HMGC for several cases, corresponding to different choices of n. It is possible to note that, for each value of n, the efficiency essentially maintains its ideal value ($\eta \approx 1$) up to a certain threshold value in $n_{\text{proc}}/N_{\text{ppc}}$. Such a value comes out to be higher in the FSP case than in the PIC one, as expected, because of the lesser amount of replicated calculations that characterizes the FSP code.[2] Note that the PIC results would be less penalized if the FFT algorithm were adopted, instead of the less-memory-demanding FT one.

The most relevant features, here, is represented by the fact that, while for the PIC code the $n_{\text{proc}}/N_{\text{ppc}}$ threshold value exhibits a negligible dependence on n, for the FSP one such a dependence is positive and much stronger. This fact can be understood by considering that the threshold value is approximately obtained by imposing that the replicated calculations exceed the distributed ones. For the PIC case, from Eq. (46), such a condition can be written as

$$N_r \hat{f}(N_{\text{harm}}) + n_{\text{FT}} N_{\text{harm}} N_r N_\vartheta N_\varphi \gtrsim n_{\text{int}} \frac{N_{\text{part}}}{n_{\text{proc}}}. \tag{50}$$

[2] For the largest simulations, superlinear results are obtained, which can be possibly traced back to memory and/or cache effects and compiler options.

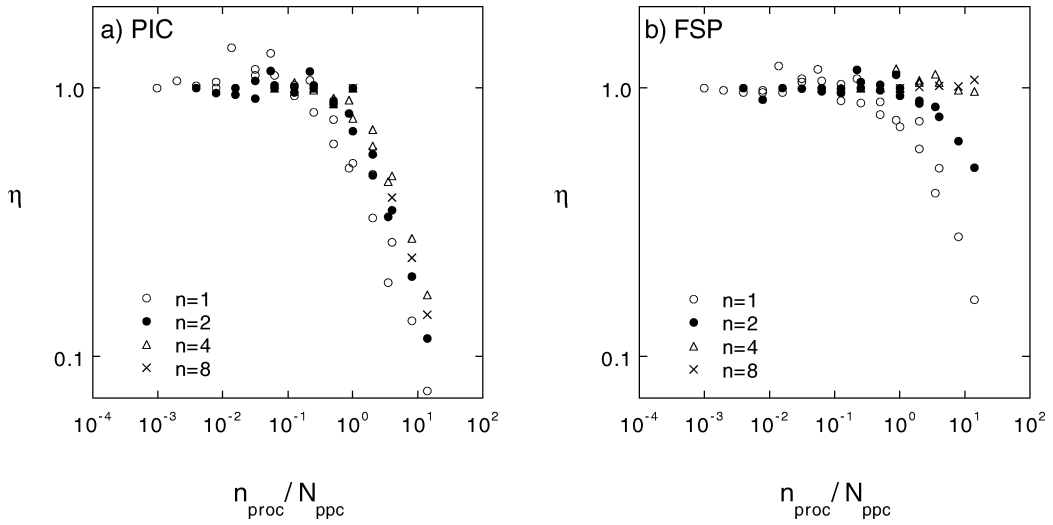

Fig. 5. Efficiency η, defined as the speed-up factor divided by the number of processors, versus $n_{\mathrm{proc}}/N_{\mathrm{ppc}}$. Simulations retaining modes with a single toroidal number are considered. The results obtained by PIC HMGC (a) are compared with those obtained by FSP HMGC (b), for several different choices of n.

Taking into account the dependence on n of each term (see Section 5), this yields, for the threshold, the following approximate dependence:

$$T_{\mathrm{PIC}} \equiv \left.\frac{n_{\mathrm{proc}}}{N_{\mathrm{ppc}}}\right|_{\mathrm{th}}^{\mathrm{PIC}} \propto \frac{n^3}{n\hat{f}(\alpha_1 n) + \alpha_{\mathrm{FT}} n^4}, \tag{51}$$

where α_1 is defined by $N_{\mathrm{harm}} \equiv \alpha_1 n$, and α_{FT} is a coefficient depending on n_{FT} and other parameters related to the specific simulations considered.

For the FSP code, the corresponding condition is instead obtained from Eq. (43) and reads

$$N_r \hat{f}(N_{\mathrm{harm}}) \gtrsim n_{\mathrm{FT}} N_{\mathrm{harm}} \frac{N_{\mathrm{part}}}{n_{\mathrm{proc}}}. \tag{52}$$

The threshold is then given by

$$T_{\mathrm{FSP}} \equiv \left.\frac{n_{\mathrm{proc}}}{N_{\mathrm{ppc}}}\right|_{\mathrm{th}}^{\mathrm{FSP}} \propto \frac{n^3}{\hat{f}(\alpha_1 n)}. \tag{53}$$

From Eqs. (51) and (53), we get, for the ratio between the above threshold values,

$$\frac{T_{\mathrm{FSP}}}{T_{\mathrm{PIC}}} \propto n\left[1 + \frac{\alpha_{\mathrm{FT}} n^3}{\hat{f}(\alpha_1 n)}\right]. \tag{54}$$

We see that the advantages of the FSP method in terms of parallelization efficiency become more apparent with increasing n. In the Appendix, it is shown that such conclusion would maintain its qualitative validity even if the FFT algorithm were used in the PIC approach. We can then conclude that the particle-decomposition parallelization of FSP codes is potentially efficient even for massively parallel architectures, not only for simulations characterized by high resolution in the velocity space (high N_{ppc} values), but also – different from the same parallelization of PIC codes – for simulations with high resolution in the real space (high n values).

Fig. 6 shows the Central Processor Unit (CPU) *user time* required by each simulation time step versus the number of processors. Results obtained by FSP (empty symbols) and PIC-FT (full symbols) simulations with fixed

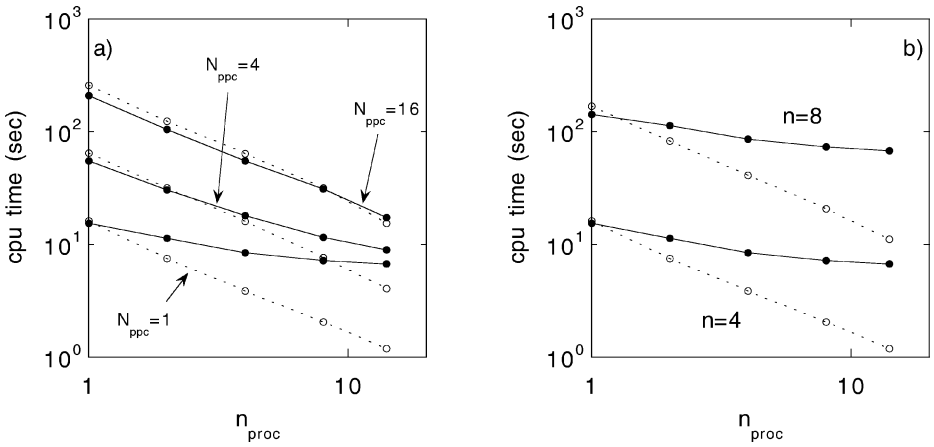

Fig. 6. CPU *user time* per time step versus n_{proc}. Results obtained by FSP (empty symbols) and PIC (full symbols) simulations with fixed mode number ($n = 4$) and different values of N_{ppc} (a), or fixed velocity-space resolution ($N_{\mathrm{ppc}} = 1$) and different values of n (b) are compared.

mode number ($n = 4$) and different values of N_{ppc} (a), or fixed velocity-space resolution ($N_{\mathrm{ppc}} = 1$) and different values of n (b) are compared. It can be seen that the FSP approach becomes more advantageous when the number of processors exceeds a certain threshold value, approximately equal to N_{ppc} and essentially independent on n, in agreement with Eq. (34). Different behaviours would be found if PIC-FFT simulations were taken into account (cf. Eq. (33)).

7. Conclusions

In the present paper we have reconsidered the suitability of the gridless finite-size-particle approach to plasma physics particle simulation. The comparison of this method with the more usual particle-in-cell method, referred to a specific particle simulation code (the HMGC one), exhibits a very good agreement, if the size of the simulation particles in the FSP method is of the same order of the spacing of grid points in the PIC one.

We have shown that, in the framework of the *particle decomposition* approach, the gridless finite-size-particle method allows to minimize the heavy memory request of high spatial resolution parallel simulation. In this respect and from the point of view of parallel efficiency such a method is more convenient than the usual particle-in-cell one, because it reduces both the size of the replicated arrays and the amount of replicated calculations assigned to each computational node. On the other hand, all the features that make the *particle decomposition* approach to parallelization of PIC codes more convenient than the *domain decomposition* one are preserved in the FSP case: perfect load balancing, simplified and reduced inter-processor communication, easy implementation of the parallel version of originally serial codes.

The FSP method can be advantageous also concerning the absolute CPU-time performances, as far as the number of processors exceeds a certain threshold. This is due to the fact that, although the FSP method requires much heavier calculations for each particle (the electromagnetic fields are transformed back from the Fourier space directly at the particle position), such calculations are distributed among processors. If the FT algorithm is adopted, in order to save memory space, in the PIC simulations, the threshold is essentially given by the average number of particle per cell, N_{ppc}, and is easily overcome, even in moderately parallel simulations. If the more efficient FFT algorithm is instead adopted, the threshold also depends on the number of harmonics retained in the simulation and, more weakly, on the required spatial resolution. In concrete situations, the number of processors can easily come out to be below this threshold if the full set of harmonics corresponding to a given spatial resolution is retained.

If, however, for the sake of simplifying the investigation, only a limited number of harmonics is retained, without reducing the resolution requirement, the threshold value sensibly decreases, and the FSP method can prevail also over the FFT-PIC one. In practice, the resort to such simplified *few-harmonic* simulations is often forced by the limitations on the memory available on each computational node.

Appendix A. The PIC HMGC code in the framework of FFT approach

In order to extend as much as possible the comparison between FSP and PIC methods, we have considered, in this paper, a version of the PIC HMGC simulation code which uses the FT algorithm, instead of the more efficient FFT one, in order to minimize the memory requirements for a given spatial resolution level.

As noted in the text, if the FFT algorithm were used, some of the conclusions achieved with regard to the FSP vs. PIC comparison would have to be revised, at least from a quantitative point of view. In particular, referring to the specific code (HMGC) considered here, from Eqs. (26) and (27) we would obtain, for the number of operations and the memory requirement of a FFT-PIC simulation, instead of Eqs. (46) and (46), the following expressions:

$$N_{\text{op p.d.}}^{\text{PIC}} \approx N_r f\left(\tfrac{1}{2}N_\vartheta N_\varphi\right) + n_{\text{FFT}} N_r N_\vartheta N_\varphi \log_2(N_\vartheta N_\varphi) + n_{\text{int}} \frac{N_{\text{part}}}{n_{\text{proc}}}, \tag{A.1}$$

$$M_{\text{p.d.}}^{\text{PIC}} \approx \tfrac{1}{2}\widehat{m}_{\text{harm}} N_r N_\vartheta N_\varphi + m_{\text{cell}} N_r N_\vartheta N_\varphi + m_{\text{part}} \frac{N_{\text{part}}}{n_{\text{proc}}}. \tag{A.2}$$

The threshold for ideal efficiency of such simulations would come out, from Eq. (A.2), to be given by

$$T_{\text{PIC}} \propto \frac{n^3}{n\hat{f}(\alpha_2 n^2) + \alpha_{\text{FFT}} n^3 \log_2 n}, \tag{A.3}$$

with a dependence on n not much more favorable than that obtained, in Eq. (51), within the FT approach. Here α_2 is defined by $\frac{1}{2}N_\vartheta N_\varphi \equiv \alpha_2 n^2$, and α_{FFT} depends, analogously to α_{FT}, on n_{FT} and other parameters specific of the considered simulations. The ratio between the FSP threshold and the PIC one would then scale as

$$\frac{T_{\text{FSP}}}{T_{\text{PIC}}} \propto n \frac{\hat{f}(\alpha_2 n^2) + \alpha_{\text{FFT}} n^2 \log_2 n}{\hat{f}(\alpha_1 n)}, \tag{A.4}$$

to be compared with Eq. (54).

References

[1] J. Wesson, Tokamaks, Clarendon Press, Oxford, 1997.
[2] R.W. Hockney, J.W. Eastwood, Computer Simulation Using Particles, McGraw-Hill, New York, 1981.
[3] C.K. Birdsall, A.B. Langdon, Plasma Physics via Computer Simulation, McGraw-Hill, New York, 1985.
[4] P.C. Liewer, V.K. Decyk, A general concurrent algorithm for plasma particle-in-cell codes, J. Comput. Phys. 85 (1989) 302–322.
[5] A. Pukhov, Three-dimensional electromagnetic relativistic particle-in-cell code VLPL (Virtual Laser Plasma Lab), J. Plasma Phys. 61 (1999) 425–433.
[6] B. Di Martino, S. Briguglio, G. Vlad, P. Sguazzero, Parallel plasma simulation in high performance Fortran, in: High Performance Computing and Networking, Springer, Berlin, 1998, pp. 203–212.
[7] L. Chen, F. Zonca, Theory of shear Alfvén waves in toroidal plasmas, Phys. Scripta T 60 (1995) 81–90.
[8] S. Briguglio, G. Vlad, F. Zonca, C. Kar, Hybrid magnetohydrodynamic-gyrokinetic simulation of toroidal Alfvén modes, Phys. Plasmas 2 (1995) 3711–3723.
[9] A. Sestero, Basic interactions in real plasmas and in the plasmas of numerical experiments, Il Nuovo Cimento B 9 (1972) 222–232.
[10] A.B. Langdon, C.K. Birdsall, Theory of plasma simulation using finite-size particles, Phys. Fluids 13 (1970) 2115–2122.
[11] R.W. Hockney, Computer simulation of anomalous plasma diffusion and numerical solution of Poisson's equation, Phys. Fluids 9 (1966) 1826–1835.

[12] H.R. Strauss, Nonlinear, three-dimensional magnetohydrodynamics of noncircular tokamaks, Phys. Fluids 20 (1977) 134–140.
[13] R. Izzo, D.A. Monticello, W. Park, J. Manickam, H.R. Strauss, R. Grimm, K. McGuire, Effects of toroidicity on resistive tearing modes, Phys. Fluids 26 (1983) 2240–2246.
[14] W.W. Lee, Gyrokinetic approach in particle simulations, Phys. Fluids 26 (1983) 556–562.
[15] S. Briguglio, F. Zonca, G. Vlad, Hybrid magnetohydrodynamic-particle simulation of linear and nonlinear evolution of Alfvén modes in tokamaks, Phys. Plasmas 5 (1998) 3287–3301.
[16] L. Chen, Theory of magnetohydrodynamic instabilities excited by energetic particles in tokamaks, Phys. Plasmas 1 (1994) 1519–1522.
[17] H. Richardson, High performance Fortran: history, overview and current developments, Tech. Rep. TMC-261, Thinking Machines Corporation, 1996.
[18] M. Gupta, S. Midkiff, E. Schonberg, V. Seshadri, D. Shields, K.Y. Wang, W.M. Ching, T. Ngo, A HPF Compiler for the IBM SP2, in: SuperComputing'95, ACM, 1995.
[19] G. Vlad, C. Kar, F. Zonca, F. Romanelli, Phys. Plasmas 2 (1995) 418.