A novel approach to ion–ion Langevin self-collisions in particle-in-cell modules applied to hybrid MHD codes

2017 Plasma Phys. Control. Fusion 59 054005

(http://iopscience.iop.org/0741-3335/59/5/054005)

IOP Publishing Plasma Physics and Controlled Fusion
Plasma Phys. Control. Fusion 59 (2017) 054005 (11pp) https://doi.org/10.1088/1361-6587/aa5f71

A novel approach to ion–ion Langevin self-collisions in particle-in-cell modules applied to hybrid MHD codes

T Nicolas[1], J-F Luciani[2], H Lütjens[2], X Garbet[3] and J Graves[1]

[1] EPFL, Swiss Plasma Center (SPC), CH-1015 Lausanne, Switzerland
[2] Centre de Physique Théorique, Ecole Polytechnique, CNRS, F-91128 Palaiseau Cedex, France
[3] CEA, IRFM, F-13108 Saint-Paul-Lez-Durance, France

E-mail: timothee.nicolas@epfl.ch

Received 15 November 2016, revised 27 January 2017
Accepted for publication 9 February 2017
Published 24 March 2017

Abstract

In order to have a better closure for magnetohydrodynamic (MHD) equations, a common approach is to obtain the ion fluid pressure tensor by directly computing the moments of an ion distribution function, obtained by a particle-in-cell solver of the Vlasov or Boltzmann equation. This is the so-called hybrid approach. Long MHD simulations are required for problems such as investigating the properties of the sawtooth cycle. In such long hybrid simulations, collisions are required to relax the distribution function after violent MHD events, and to obtain the self-consistent neoclassical transport. In this paper, we present a new approach to ion self-collisions, based on temperature- and velocity-shifted Maxwellian distributions. It is shown that the approach emulates the effect of the background reaction, without the need to explicitly implement it. Arbitrariness in the choice of the closest Maxwellian is removed. The model compares very well with binary collision Monte-Carlo simulations. The practical implementation as a Fokker–Planck module in a hybrid kinetic/MHD simulation code is discussed. This requires an additional manipulation in order to conserve energy and momentum.

Keywords: collisions, background reaction, hybrid kinetic/magnetohydrodynamics simulation

(Some figures may appear in colour only in the online journal)

1. Introduction

Although purely fluid models of the plasma dynamics are intrinsically limited by the closure problem, this issue can be partly overcome by coupling the fluid magnetohydrodynamic (MHD) dynamics to an ion kinetic solver. The moments of the ion distribution function can be used to close the momentum equation. This results in models such as that of [1, 2], where electrons evolve as a fluid, while the ions are evolved kinetically. The XTOR-K code [3] implements such a hybrid model: it consists of a two-fluid MHD solver based on the XTOR-2F code [4], self-consistently coupled to a 6D nonlinear kinetic particle-in-cell (PIC) full-*f* solver. The particle trajectories are integrated along their gyromotion by means of a Boris–Buneman scheme [5], thus, all finite Larmor radius effects and the associated nonlinearities are taken into account.

The hybrid approach allows efficient modelling of MHD instabilities while retaining important kinetic effects, such as the resonances between the bounce motion of suprathermal particles and the MHD modes. More specifically, it allows a study into the effect of suprathermal ions in burning plasmas, such as the alpha particles resulting from fusion reactions or the fast particles generated by the heating systems, on the MHD dynamics. The fast particles can modify the stability of preexisting instabilities, e.g. sawteeth [6] and Alfvén eigenmodes [7], or trigger new instabilities, e.g. the fishbone instability [8]. The detail of the full nonlinear dynamics can only be investigated using numerical tools. In [6], the sawtooth period τ_s is estimated to be in the range $50\,\mathrm{s} < \tau_s < 140\,\mathrm{s}$ for ITER ignited discharges. It is important to determine the sawtooth period in ITER with more precision because it is known that longer sawtooth

periods increase the risk that the sawtooth crash triggers a neoclassical tearing mode [9, 10], an instability detrimental to the overall plasma confinement.

On such long time scales, the effect of collisions becomes important. First, collisions are responsible for the neoclassical effects, which have an impact on the dynamics of tearing modes [11]. Second, collisions allow relaxation of the ion distribution function after a sawtooth crash, which may cause significant deviations with respect to a Maxwellian. Finally, collisions are also responsible for the thermalization of the alpha particles, which adopt a slowing down distribution. This paper discusses the implementation of an ion self-collision operator in the XTOR-K code.

There has been a lot of effort in developing models for simulating collisions in plasmas. Among numerous others, two popular methods applicable to long time simulations are the Takizuka–Abe binary collision method [12] (hereafter denoted by TA algorithm) and the δf with evolving background method of Brunner *et al* [13]. Both methods allow to model accurately the effect of the deviation δf with respect to the background Maxwellian distribution, while conserving energy and momentum. The δf method is not adapted to the XTOR-K case, since the latter evolves the total distribution function f. The TA method has the drawback that it is not well adapted to the parallel environment of XTOR-K. Indeed, in XTOR-K, the particle positions and velocities are initialized on each process and then remain on their initialization process until the end of the simulation. In other words, the parallelization is done by domain cloning, as opposed to domain decomposition. The TA algorithm implements binary collisions with particles in the neighborhood of a target particle, which requires to sort all the particles according to their positions and reaffect them to different processes, an operation that is very expensive in communications, CPU time and memory.

Instead, we have decided to take the simpler approach of Langevin collisions on a Maxwellian distribution. *A priori*, such an approach suffers from the fact that it is not momentum/energy conserving, and that it does not properly take into account the reaction on the background plasma (herafter simply denoted background reaction) from collisions with δf. In this paper, we show how to cure these flaws by properly choosing the Maxwellian, and how to implement the resulting algorithm in the parallel environment of XTOR-K. The paper is organized as follows. In section 2, the algorithm we propose to treat self-collisions is detailed in a local in space, three-dimensional in velocity space framework. It is then justified with a few examples in sections 3, 4 and 5, by comparison to analytical formulæ as well as to the TA algorithm. In section 6, the issue of the implementation of the Fokker–Planck module in the XTOR-K code is investigated. Section 7 discusses the possible pitfalls of the implementation. Finally, conclusions are drawn in section 8.

2. Self collision algorithm

The general kinetic equation for the distribution function of a species a with collisional Coulomb interaction is

$$\frac{\mathrm{d}f_a}{\mathrm{d}t} = \sum_b \mathcal{C}_{ab}[f_b, f_a], \tag{1}$$

where the sum is on all species b and the collision operator is the Landau operator given by [14, 15]

$$\mathcal{C}_{ab}[f_b, f_a] = \Gamma_{ab} \times \frac{\partial}{\partial \mathbf{v}_a} \cdot \int \overline{U}(\boldsymbol{u}) \cdot \left(\frac{f_b}{m_a} \frac{\partial f_a}{\partial \mathbf{v}_a} - \frac{f_a}{m_b} \frac{\partial f_b}{\partial \mathbf{v}_b} \right) \mathrm{d}^3\mathbf{v}_b, \tag{2}$$

where $\Gamma_{ab} = Z_a^2 Z_b^2 e^4 \Lambda / (8\pi\epsilon_0^2 m_a)$, Λ is the Coulomb logarithm, $\boldsymbol{u} = \mathbf{v}_a - \mathbf{v}_b$ and $\overline{U}(\boldsymbol{u}) = (1 - \boldsymbol{u}\boldsymbol{u}/u^2)/u$.

From now on, unless specified otherwise we consider only self-collisions and drop all species indices a and b. The Landau operator can be rewritten in the form of a Fokker–Planck operator as

$$C[f, f] = -\frac{\partial}{\partial \mathbf{v}} \cdot \left[\mathbf{F} f - \frac{1}{2} \frac{\partial}{\partial \mathbf{v}} \cdot (\overline{\mathbf{D}} f) \right]. \tag{3}$$

The friction $\mathbf{F}$ and diffusion $\overline{\mathbf{D}}$ are given by

$$\mathbf{F} = \frac{Z^4 e^4 \Lambda}{2\pi\epsilon_0^2 m^2} \frac{\partial h}{\partial \mathbf{v}}, \tag{4}$$

$$\overline{\mathbf{D}} = \frac{Z^4 e^4 \Lambda}{4\pi\epsilon_0^2 m^2} \frac{\partial^2 g}{\partial \mathbf{v} \partial \mathbf{v}}, \tag{5}$$

and h, g are the Rosenbluth potentials [16],

$$h(\mathbf{v}) = \int f(\mathbf{v}') \frac{1}{|\mathbf{v}' - \mathbf{v}|} \mathrm{d}^3\mathbf{v}', \tag{6}$$

$$g(\mathbf{v}) = \int f(\mathbf{v}') |\mathbf{v}' - \mathbf{v}| \mathrm{d}^3\mathbf{v}'. \tag{7}$$

The effect of the collision operator (3) on the distribution function can be computed with particles using a Langevin formulation rather than directly in the continuum. We adopt the following so-called Euler–Maruyama scheme,

$$\Delta\mathbf{v} = \mathbf{F}(\mathbf{v})\Delta t + \mathbf{d}(\mathbf{v}) \mathcal{N}_\mathrm{g} \sqrt{\Delta t} \tag{8}$$

with $\mathbf{d}\mathbf{d}^\mathrm{T} = \overline{\mathbf{D}}$ and $\mathcal{N}_\mathrm{g}$ a random number drawn from a Gaussian distribution with unit variance. Recall that a stochastic equation of the form $\mathrm{d}x = f(x)\mathrm{d}t + g(x)\mathrm{d}w(t)$, where $W(t) = \int_0^t \mathrm{d}w(t')$ designates the Wiener process, does not have any meaning *per se*. This is essentially because the Wiener process is not differentiable and advances by an infinity of infinitesimal jumps, hence one does not know whether $g(x)$ should be evaluated with the value of x before or after the jump (or a combination of the two). This ambiguity leads to the famous Itō/Stratonovich controversy, the solution to which is exposed with remarkable clarity in reference [17]. We use the Itō rule, which corresponds to the simple choice where $\mathbf{d}(\mathbf{v})$ is evaluated with the value of $\mathbf{v}$ before the kick. Such a choice leads to the evolution of the

distribution converging to equation (3) in the limit $\Delta t \to 0$ and $N \to \infty$ [17, 18].

The difficulty of evaluating $\mathbf{F}$ and $\overline{\mathbf{D}}$ lies in the fact that the Rosenbluth potentials cannot be computed analytically for arbitrary distributions. Since they are known for Maxwellian distributions, we restrict ourselves to the case where the distribution function is the sum of a shifted Maxwellian and a small perturbation:

$$f = f_{\mathrm{M}}(T, \mathbf{V}) + \delta f. \tag{9}$$

Note that we do not constrain the momentum and energy, carried by δf, to vanish. In this case the collision operator, which is bilinear in f, is linearized to give

$$\mathcal{C}[f, f] = \mathcal{C}[f_{\mathrm{M}}(T, \mathbf{V}), \delta f] + \mathcal{C}[\delta f, f_{\mathrm{M}}(T, \mathbf{V})], \tag{10}$$

since $\mathcal{C}[f_{\mathrm{M}}(T, \mathbf{V}), f_{\mathrm{M}}(T, \mathbf{V})] = 0$, and the remaining nonlinear term is neglected under the assumption $|\delta f / f_{\mathrm{M}}| \ll 1$.

The first term in equation (10) represents scattering of δf from a Maxwellian background while the second term represents the background reaction of δf on the Maxwellian, which is essential to ensure the properties of energy and momentum conservation. In other words, the first term represents the action of the Maxwellian on a test particle, and the second term ensures that all momentum and energy lost by the test particle is transferred to the bulk. The first term is easy to implement because the Rosenbluth potentials, hence $\mathbf{F}$ and $\overline{\mathbf{D}}$, are known for Maxwellian distributions. However only approximate forms are known for the background reaction [13, 19], and they are not straightforward to implement in a full-f algorithm.

The main point of the present work is that it is possible to conserve energy and momentum by implementing only the first term. However, instead of choosing $f_{\mathrm{M}}(T, \mathbf{V})$ as the background Maxwellian, we choose $f_{\mathrm{M}}(T + \delta T, \mathbf{V} + \delta \mathbf{V})$, where δT and $\delta \mathbf{V}$ are chosen so as to enforce energy conservation. In a full-f framework, the implemented collision operator becomes

$$\mathcal{C}[f, f] \simeq \mathcal{C}[f_{\mathrm{M}}(T + \delta T, \mathbf{V} + \delta \mathbf{V}), f]. \tag{11}$$

By expanding the shifted-temperature Maxwellian, $f_{\mathrm{M}}(T + \delta T, \mathbf{V} + \delta \mathbf{V}) \simeq f_{\mathrm{M}}(T, \mathbf{V}) + \delta f_{\mathrm{M}}$, we see that

$$\mathcal{C}[f, f] \simeq \mathcal{C}[f_{\mathrm{M}}(T, \mathbf{V}), \delta f] + \mathcal{C}[\delta f_{\mathrm{M}}, f_{\mathrm{M}}(T, \mathbf{V})], \tag{12}$$

where we have again neglected a nonlinear term. This expression is similar to equation (10) except that δf_{M} instead of δf is used in the background reaction term. Obviously, the two are different, otherwise we would have $f = f_{\mathrm{M}}(T + \delta T)$, which would mean that δf would vanish. However, despite the difference between δf and δf_{M}, we will use equation (11) for its simplicity of implementation as well as for its energy and momentum conservation properties. Before testing this approach in relevant situations, we now detail the method to obtain the effective temperature and velocity

$$T_{\mathrm{eff}} \equiv T + \delta T, \tag{13}$$

$$\mathbf{V}_{\mathrm{eff}} \equiv \mathbf{V} + \delta \mathbf{V}. \tag{14}$$

First, the quantities are normalized: temperature and density to their initial (or reference) values T_0, n_0, velocity to $\mathrm{v}_0 = \sqrt{2T_0/m}$, energy to T_0, time to $t_0 = 4\sqrt{2}\pi\epsilon_0^2 m^{1/2} T_0^{3/2} / (n_0 Z^4 e^4 \Lambda)$, $\mathbf{F}$ to v_0/t_0 and $\overline{\mathbf{D}}$ to v_0^2/t_0. The diffusion $\overline{\mathbf{D}}$ has the form $\overline{\mathbf{D}} = D_\perp(1 - \tilde{\mathbf{v}}\tilde{\mathbf{v}}/\tilde{\mathrm{v}}^2) + D_\parallel \tilde{\mathbf{v}}\tilde{\mathbf{v}}/\tilde{\mathrm{v}}^2$, whereas $\mathbf{F}(\tilde{\mathbf{v}}) = F(\tilde{\mathrm{v}})\tilde{\mathbf{v}}/\tilde{\mathrm{v}}$ with $\tilde{\mathbf{v}} \equiv \mathbf{v} - \mathbf{V}$, therefore equation (8) is rewritten

$$\Delta \mathbf{v} = F \frac{\tilde{\mathbf{v}}}{\tilde{\mathrm{v}}} \Delta t + \sqrt{D_\perp \Delta t}\, \mathcal{N}_{\mathrm{g}} \mathbf{e}_1 + \sqrt{D_\perp \Delta t}\, \mathcal{N}_{\mathrm{g}} \mathbf{e}_2 + \sqrt{D_\parallel \Delta t}\, \mathcal{N}_{\mathrm{g}} \mathbf{e}_3, \tag{15}$$

where the three $\mathcal{N}_{\mathrm{g}}$ represent three independent calls to the Gaussian, $\mathbf{e}_3 = \tilde{\mathbf{v}}/\tilde{\mathrm{v}}$ and $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ is orthonormal. Defining $X \equiv \tilde{\mathrm{v}}/\sqrt{T}$, the friction and diffusion coefficients for collisions on a Maxwellian with temperature T and velocity shift $\mathbf{V}$ read [16]

$$F(\mathbf{v}; T, \mathbf{V}) = -\frac{1}{T}\frac{\phi(X)}{X^2}, \tag{16}$$

$$D_\parallel(\mathbf{v}; T, \mathbf{V}) = \frac{1}{2\sqrt{T}}\frac{\phi(X)}{X^3}, \tag{17}$$

$$D_\perp(\mathbf{v}; T, \mathbf{V}) = \frac{1}{2\sqrt{T}}\left[\frac{\mathrm{erf}(X)}{X} - \frac{\phi(X)}{2X^3}\right], \tag{18}$$

$$\phi(x) \equiv \mathrm{erf}(x) - \frac{2}{\sqrt{\pi}} x \mathrm{e}^{-x^2}. \tag{19}$$

During a collision step, the velocity increment is $\Delta \mathbf{v} = F\tilde{\mathbf{v}}/\tilde{\mathrm{v}}\Delta t + \mathbf{Q}$, where $\mathbf{Q}$ represents the diffusion terms and has the properties $\langle \mathbf{Q} \rangle = 0$ and $\langle \mathbf{Q}^2 \rangle = (2D_\perp + D_\parallel)\Delta t$, the mean being understood in an ensemble average sense. Hence, the average energy variation $\langle \Delta \mathcal{E} \rangle \equiv \langle (\mathbf{v} + \Delta \mathbf{v})^2 - \mathbf{v}^2 \rangle$ is

$$\langle \Delta \mathcal{E} \rangle = \mathcal{G}(\mathbf{v}; T, \mathbf{V})\Delta t + F^2 \Delta t^2, \tag{20}$$

$$\mathcal{G}(\mathbf{v}; T, \mathbf{V}) \equiv 2\tilde{\mathrm{v}}F(\tilde{\mathrm{v}}) + 2D_\perp(\tilde{\mathrm{v}}) + D_\parallel(\tilde{\mathrm{v}}). \tag{21}$$

The quantity $\mathcal{G}(\mathbf{v}; T, \mathbf{V})$ is, in an ensemble average sense and to lowest order in Δt, the energy variation of a particle with velocity norm $\tilde{\mathrm{v}}$.

Similarly, the average momentum variation $\langle \Delta \mathbf{P} \rangle$ is

$$\langle \Delta \mathbf{P} \rangle = \mathcal{H}(\mathbf{v}; T, \mathbf{V})\Delta t, \tag{22}$$

$$\mathcal{H}(\mathbf{v}; T, \mathbf{V}) \equiv \frac{\tilde{\mathbf{v}}}{\tilde{\mathrm{v}}} F(\tilde{\mathrm{v}}). \tag{23}$$

For energy and momentum to be conserved, the average over the distribution function of $\mathcal{G}$ and $\mathcal{H}$ must vanish. Therefore, we define the effective temperature T_{eff} and the effective velocity shift $\mathbf{V}_{\mathrm{eff}}$ as the solution to the system $(\overline{\mathcal{G}}(T, \mathbf{V}), \overline{\mathcal{H}}(T, \mathbf{V})) = 0$, where $\overline{\mathcal{G}}(T, \mathbf{V}) \equiv \langle \mathcal{G}(\tilde{\mathbf{v}}; T, \mathbf{V}) \rangle_f$, $\overline{\mathcal{H}}(T, \mathbf{V}) \equiv \langle \mathcal{H}(\tilde{\mathbf{v}}; T, \mathbf{V}) \rangle_f$, the mean $\langle \cdot \rangle_f$ being now understood as an average on the distribution function f.

Numerically, T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ are found by means of a Newton algorithm, given that the Jacobian of the system depends on the partial derivatives of F, $D_\perp$ and $D_\parallel$ with respect to T and $\mathbf{V}$, which are known analytically. The expressions for $\mathcal{G}$, $\mathcal{H}$ and their partial derivatives are provided in appendix A. Because the Newton algorithm converges very rapidly, a limited amount of sums on the distribution are required at each time step. Since T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ do not change much between two time steps and a very good precision is not

required in its evaluation, the Newton iterations converge typically in one to three iterations.

The approach using $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$ has two advantages, in addition to conserving energy and momentum by definition. First it does not require any speculation on the form of the background reaction term, and bypasses the problem of its practical implementation. Second, it removes the arbitrariness in the definition of the closest Maxwellian function to compute the coefficients of the first term of equation (10). Indeed, in a full-f framework, one could decide to take $\delta f = f - f_{\rm M}(T, \mathbf{V})$, where T is given by the variance of the distribution and $\mathbf{V}$ is its average velocity, but this can lead to very large errors if there is a small number of very fast particles which make a small contribution to the number distribution function and to the momentum but a large contribution to the energy. Instead, one could take for the Maxwellian the distribution that minimizes the norm $\|f - f_{\rm M}(T, \mathbf{V})\|$, but then the definitions of T, $\mathbf{V}$ and δf depend on the choice of the norm. In our approach, this issue is removed. In section 5, it will be shown that in the aforementioned case where the distribution is the sum of a Maxwellian bulk and of fast particles with a large contribution to the energy, our algorithm finds an effective temperature which is very close to the temperature of the bulk.

Even when $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$ are used, there is still a variation in the total energy and momentum after the collision step, which is due to the fact that we neglected the Δt^2 error introduced by the friction, and to the PIC noise. The Δt^2 error is a systematic positive error and can be eliminated by applying a particle dependent correction δF to the friction, given by the condition $(F + \delta F)^2 \Delta t^2 + 2\delta F \tilde{\rm v} \Delta t = 0$. Therefore, the correction to the friction is

$$F \leftarrow -\frac{\tilde{\rm v}}{\Delta t}\left(1 - \sqrt{1 + \frac{2F\Delta t}{\tilde{\rm v}}}\right). \tag{24}$$

The sign of the solution to the quadratic equation is chosen so that the correction vanishes in the limit of $\Delta t = 0$. If Δt is too large, this correction may not be applicable for the particles with a small velocity $\tilde{\rm v}$, since $F < 0$. When this is the case, the correction is simply not applied to these particles, which does not affect the results significantly if the fraction of such particles is small. In the present work, we take $0.01 < \Delta t < 0.1$, sufficiently small so that the correction can always be applied. Finally, in order to ensure energy conservation exactly, and to correct any momentum generated in the process, we apply the following transformation to the velocity at the end of each time step:

$$\mathbf{v}' \leftarrow \langle \mathbf{v} \rangle_f + \sqrt{\frac{\mathcal{V}}{\mathcal{V}'}}(\mathbf{v}' - \langle \mathbf{v}' \rangle_{f'}), \tag{25}$$

where $\mathcal{V} = \langle \mathbf{v}^2 \rangle - \langle \mathbf{v} \rangle^2$ denotes the variance of the distribution, and primed (resp. unprimed) quantities refer to quantities taken after (resp. before) the collision step. This ensures exact energy and momentum conservation. Other models use this renormalization, see e.g. [21]. The difference with [21] is that the latter conserves energy and momentum without properly taking into account the background reaction, contrary to our approach. In addition since equation (25) is always applied, the usefulness of equation (24) to correct the friction is questionable. However, in some cases, the sytematic positive error introduced by the friction causes $\sqrt{\mathcal{V}/\mathcal{V}'}$ in equation (25) to be always smaller than 1, leading to a non-physical energy transfer between one part of the distribution and the other. This can lead to large errors (see section 5), so that the correction to the friction must be implemented as well.

The following three sections test the algorithm described above, hereafter denoted $T_{\rm eff}$ algorithm, in three different cases, by comparing it to analytic formulæ and to the Takizuka–Abe binary collision algorithm. Before turning to the results, we make a short comment on the relative numerical performance of the two methods. For the present study, our algorithm is roughly 4 times faster than the TA algorithm. However, none of the algorithm was optimized numerically, because the goal of the present paper is only to assess the intrinsic properties of the new algorithm. The TA algorithm can be made faster by adopting the statistics of Nanbu [22–25] in the velocity change in a binary collision. By considering the cumulative effects of several collisions, the statistics can be shown to depart from Gaussian statistics, and taking this into account, larger time steps can be used. However, as emphasized in the introduction, the TA algorithm would be orders of magnitude more expensive than ours in the parallel environment of XTOR-K, which uses domain cloning.

3. Case of a sum of two Maxwellian distributions with different temperatures

The rate of energy transfer between two Maxwellian distributions of species a and b with temperature T_a and T_b and no relative velocity is given in [20] (Chapter 5). In the case of species of same mass and charge, it is, in our normalization:

$$\frac{3}{2} n_b \frac{{\rm d}T_b}{{\rm d}t} = \frac{2 n_a n_b}{\sqrt{\pi}} \frac{T_a - T_b}{(T_a + T_b)^{3/2}}, \tag{26}$$

which can be integrated numerically with the constraint of energy conservation

$$n_a T_a + n_b T_b = (n_a T_a + n_b T_b)|_{t=0}. \tag{27}$$

To compare the numerical solutions of this equation with the results produced by the $T_{\rm eff}$ algorithm described in section 2, we initialize two Maxwellian distributions $f_a = f_{{\rm M}a}(T_a, 0)$ and $f_b = f_{{\rm M}b}(T_b, 0)$ with different temperatures and join them in a single distribution $f = f_a + f_b$. In principle, we could represent the different densities of the two distributions by attaching different weights to the particles. However, in view of the implementation in the XTOR-K code, where all particles have the same constant weight, the density ratio is actually equal to the ratio of the number of particles in each of the distributions. That is, if f_a is represented with N_a particles and f_b with N_b particles, then the densities n_a and n_b to be used in equations (26) and (27) are

given by $n_a/n_b = N_a/N_b$ and $n_a + n_b = 1$. The latter equality comes from our normalization.

A priori, the T_{eff} algorithm should work well only when one of the distributions, say f_b, represents a small perturbation to the other one, that is, if $N_b \ll N_a$ and/or if $|T_b - T_a| \ll T_a$. However, we will see that it works well even when it is not the case for some choice of parameters, so we do not make this assumption in the following.

In order to increase the reliability of our results, we also make the comparison with the TA algorithm [12], which reproduces equation (2) exactly in the limit of $\Delta t \to 0$, $N \to \infty$. In section 5 where analytical formulæ are not known, this is the most reliable way to test our algorithm, but even in the present case, it is useful because we do not expect the temperature curves to follow the solution of equations (26) and (27) exactly. Indeed, in the relaxation process, the distributions dynamically deviate from a Maxwellian distribution, so that the analytical formulæ no longer describe the temperature relaxation with high accuracy. Nonetheless, they should give the correct initial slope.

We can compute the effective temperature analytically in the case of two Maxwellian distributions with different temperatures. It is given by equating the energy transfer of species a and b on the effective Maxwellian with temperature T_{eff} and density $n_a + n_b = 1$. For symmetry reasons, we can take $\mathbf{V}_{\text{eff}} = 0$ in this section. Therefore, T_{eff} is the solution to

$$n_b \frac{T_{\text{eff}} - T_b}{(T_{\text{eff}} + T_b)^{3/2}} + n_a \frac{T_{\text{eff}} - T_a}{(T_{\text{eff}} + T_a)^{3/2}} = 0. \tag{28}$$

When $n_b = 0$, $T_{\text{eff}} = T_a$, so we can expand T_{eff} in the parameter $\delta \equiv n_b/n_a$ as

$$T_{\text{eff}} = T_a + \delta T_1 + \delta^2 T_2 + \mathcal{O}(\delta^3). \tag{29}$$

By solving order by order and defining $\Delta T \equiv T_b - T_a$, $Y \equiv [1 + \Delta T/(2T_a)]^{-1}$, we find

$$T_1 = \Delta T Y^{3/2}, \tag{30}$$

$$T_2 = T_a(-3Y^4 + 11Y^3 - 11Y^2 + 3Y). \tag{31}$$

This formula works reasonably well up to $n_b \sim n_a$. Using this result, we can estimate the error in the energy transfer rate between the two distributions (in fact the first order of equation (29) is sufficient for this purpose). We have to compare the correct rate $n_a n_b (T_a - T_b)/(T_a + T_b)^{3/2}$ with the actual rate in the simulation, $n_b(n_a + n_b)(T_{\text{eff}} - T_b)/(T_{\text{eff}} + T_b)^{3/2}$. The ratio between the latter and the former is, to order 1 in δ,

$$1 + K(\Delta T)\delta, \tag{32}$$

$$K(\Delta T) = 1 - \frac{1 + \frac{3}{2}\frac{\Delta T}{2T_a + \Delta T}}{\left(1 + \frac{\Delta T}{2T_a}\right)^{3/2}} = \frac{15}{32}\left(\frac{\Delta T}{T_a}\right)^2 + \mathcal{O}(\Delta T^3) > 0. \tag{33}$$

Hence, the lowest order correction to the energy transfer rate is small, of order $\Delta T^2 (n_b/n_a)$, which makes the algorithm robust for reasonable values of ΔT. Heuristically, we can interpret this result as the fact that for larger densities n_b, T_{eff} becomes larger, reducing the effective ΔT seen by the f_b part of the distribution, but the density of the target distribution increases in the same time, which restores the correct energy transfer rate to order ΔT^2. From equation (33), we can predict that even for $n_b = n_a$, the Maxwellian energy transfer rate for small ΔT will be modelled well by our algorithm because the error is at second order in ΔT. However, it should break down for $\Delta T \gg T_a$, if the condition $n_b \ll n_a$ is not met.

The results of simulations of relaxation between two Maxwellian distributions can be seen in figure 1, for $n_b/n_a = 0.1$, $T_b/T_a = 2$ (a), $n_b/n_a = 0.5$, $T_b/T_a = 2$ (b), $n_b/n_a = 1$, $T_b/T_a = 2$ (c), $n_b/n_a = 0.1$, $T_b/T_a = 0.5$ (d), $n_b/n_a = 0.5$, $T_b/T_a = 0.5$ (e), $n_b/n_a = 1$, $T_b/T_a = 0.5$ (f). The value of N_a in all these simulations is $N_a = 10^5$. The two values of ΔT are chosen so that $K(\Delta T) \ll 1$. In order to improve readability, only the analytical result is plotted for f_a, since T_a can always be deduced from T_b and conservation of energy (which is exact in all the algorithms used in this paper). It is seen that our algorithm and the TA algorithm produce very similar results, almost undistinguishable. It can be verified that the discrepancy with the result of equations (26) and (27) is indeed due to the deviation from the Maxwellian distribution. It disappears if at each time step, the distributions are replaced with Maxwellian distributions with temperature given by the variance of the distribution.

We can also compare the deviation of the distribution with respect to the Maxwellian $\delta f_a = f_a - f_{\text{Ma}}(T_a)$ and $\delta f_b = f_b - f_{\text{Mb}}(T_b)$. Figure 2 shows the result for the distribution function in v_x, for $n_a = n_b$ and $T_b(0) = 2T_a(0)$ at $t/t_0 = 15$. It is seen that the deviation is virtually identical for the TA algorithm and the T_{eff} algorithm.

Finally we push the algorithm outside its domain of validity identified above by using $n_b = n_a$ and $\Delta T \gg T_a$, namely $T_b = 10T_a$. The result can be seen in figure 3. The energy transfer rate is too large at first, which is consistent with the positivity of K, see equation (33). However, on the long run, as ΔT becomes smaller and T_{eff} becomes closer to T_a, the algorithm behaves much better and once again reproduces the results of the TA algorithm.

We conclude that the T_{eff} algorithm is efficient at simulating the relaxation of two Maxwellian distributions, not only in obtaining the correct energy transfer rates, but also in reproducing the correct shape for δf. It is remarkable that the algorithm performs very well even if the condition $n_b \ll n_a$ or $\Delta T < T_a$ is not met.

4. Case of a sum of two Maxwellian distributions with different velocities

In this section we investigate the case where the f_a and f_b parts of the distribution have different average velocities. First, we take f_b to represent a cold beam with significant momentum: $T_b \ll T_a$, $|\langle \mathbf{v}_b \rangle_{f_b}| > \sqrt{T_a}$, $n_b < n_a$. This can be thought of as the interaction between the bulk plasma and particles from a neutral beam, just after ionization by the plasma. Again, we compare the results with the TA algorithm. Figure 4 shows

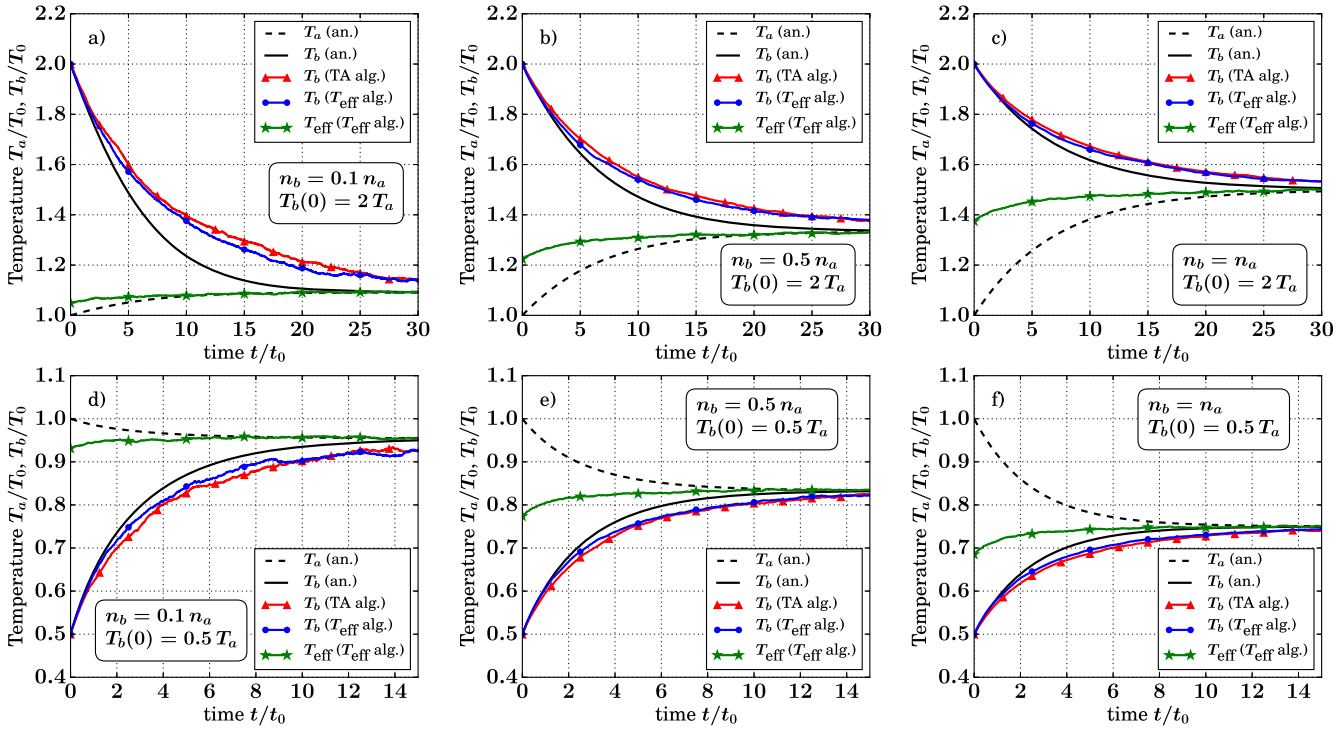

Figure 1. Comparison between the $T_{\rm eff}$ algorithm, the binary collision algorithm of [12], and the analytical formula of [20]. The parameters are as follows: (a) $n_b/n_a = 0.1$, $T_b/T_a = 2$, (b) $n_b/n_a = 0.5$, $T_b/T_a = 2$, (c) $n_b/n_a = 1$, $T_b/T_a = 2$, (d) $n_b/n_a = 0.1$, $T_b/T_a = 0.5$, (e) $n_b/n_a = 0.5$, $T_b/T_a = 0.5$, (f) $n_b/n_a = 1$, $T_b/T_a = 0.5$.

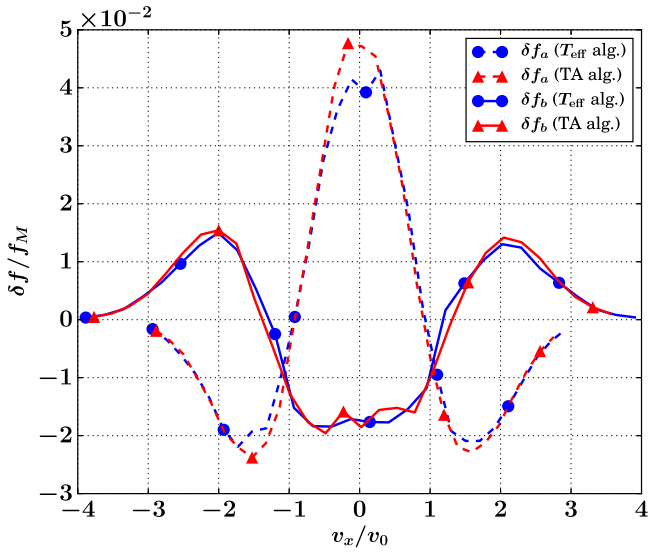

Figure 2. Deviation δf from the Maxwellian distribution for the $T_{\rm eff}$ algorithm compared to the TA algorithm.

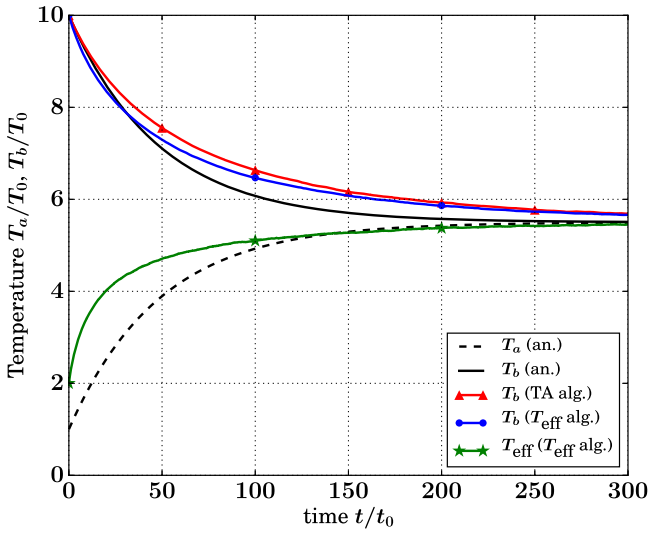

Figure 3. Comparison between the $T_{\rm eff}$ and the TA algorithms in the case $n_b = n_a$, $T_b = 10T_a$.

the data obtained with $n_b/n_a = 0.1$ and $n_b/n_a = 0.5$. In both cases the initial velocity difference is $V_{x,b} = 10v_0$.

In figures 4(a) and (b), the density of the beam is small. As expected, the beam directed energy is isotropized first, so that its temperature increases rapidly. Meanwhile, it transfers its energy and momentum to the Maxwellian bulk. Both $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$ follow closely the temperature and velocity values of the f_a part of the distribution. The small discrepancy with T_a and $\mathbf{V}_a$ allows for the conservation of energy and momentum. There is a noticeable difference in the maximum temperature reached by the beam between the two models, but the initial energy and momentum transfer rates are virtually equal. The time necessary to relax to a single Maxwellian distribution is almost the same, albeit a bit shorter for the $T_{\rm eff}$ algorithm.

However, in figures 4(c) and (d), where the beam density is comparable to the bulk density, the difference between the two models becomes significant. The initial energy and momentum transfer rates are no longer equal, and the effective temperature and velocity depart noticeably from their values for the f_a part of the distribution. Nonetheless, the

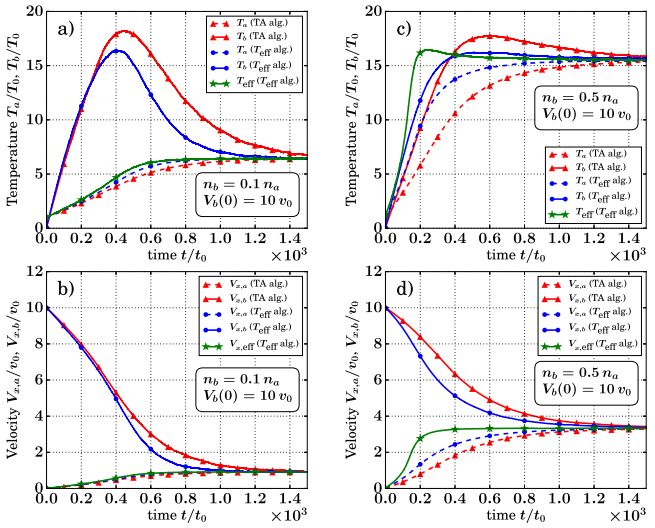

Figure 4. Comparison between the $T_{\rm eff}$ algorithm and the binary collision algorithm of [12]. Temperature evolution in (a), (c), velocity evolution in (b), (d). The parameters are as follows: (a) and (b): $n_b/n_a = 0.1$, (c) and (d): $n_b/n_a = 0.5$.

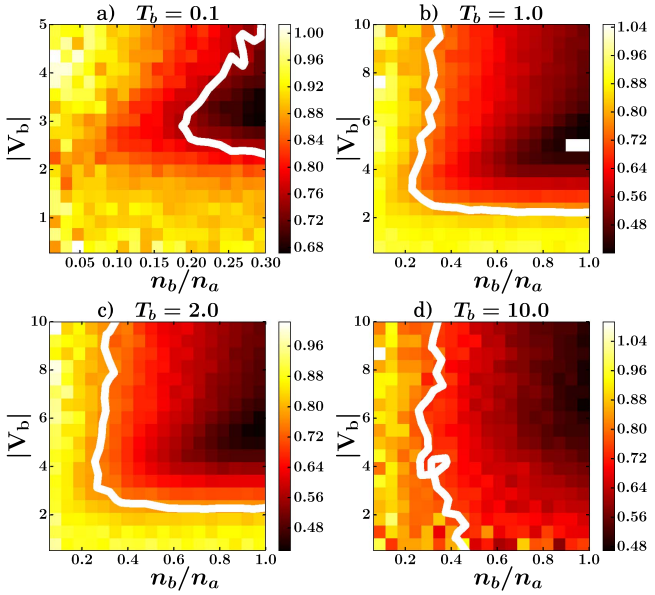

Figure 5. Ratio of $|d\mathbf{V}/dt|_{t=0}$ in the $T_{\rm eff}$ algorithm to the same quantity in the TA algorithm, for $T_b = T_a/10$ (a), $T_b = T_a$ (b), and $T_b = 2T_a$ (c).

overall time to relax to the Maxwellian distribution has the good order of magnitude.

In order to estimate the domain of validity of the algorithm, we compare the initial momentum transfer rates for different values of the density ratio and the velocity difference. Figure 5 displays the ratio between $|d\mathbf{V}/dt|_{t=0}$ in the TA algorithm to the same quantity for the $T_{\rm eff}$ algorithm, for $T_b = T_a/10$ (a), $T_b = T_a$ (b), $T_b = 2T_a$ (c) and $T_b = 10T_a$ (d). When data points are left blank in figure 5(b), it means that the Newton algorithm to find $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$ does not converge, due to the difficulty to find a good guess. Each data point is obtained by averaging the momentum transfer rate over 100 points to reduce the PIC noise. The contour drawn in white on the figure corresponds to a ratio of 0.75, that is, the $T_{\rm eff}$ algorithm overestimates the transfer rate by about one third. Except for $T_b \gg T_a$ (d), the region where the overestimation is less than one third is defined by $n_b < 0.2n_a$ or $|\mathbf{V}_b| < 2$. In other words, the density ratio can be large provided that the velocity shift is small, and the velocity shift can be large provided that the density ratio is small.

In practice, the most frequent applications verify the small density condition (e.g. beam injection) and/or involve subsonic flows $|V_b| \ll 1$ with small temperature discrepancies. Therefore, the $T_{\rm eff}$ algorithm can be used in such practical applications.

5. Case of alpha particles

In this section, we investigate the slowing-down properties of alpha particles, which represents an important situation in view of the simulation of burning plasmas. It is well-known that alpha particles deposit most of their energies on electrons. However, as the alpha particles are slowed down, this becomes less true. Let us study the ratios of the self-collision slowing-down frequency of an alpha particle $\nu_{\rm s}^{\alpha/\alpha}$ (on the thermalized Helium ash) to the slowing-down frequency on the main bulk ion $\nu_{\rm s}^{\alpha/{\rm i}}$ and to the slowing-down frequency of the electrons $\nu_{\rm s}^{\alpha/{\rm e}}$ (see [26]). Assuming a plasma containing 10% of Helium, (the maximum value allowed in ITER to prevent fuel dilution [27]), one obtains:

$$\frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm e}}} = \frac{2}{1+\frac{m_\alpha}{m_{\rm e}}}\frac{\phi(v_\alpha/v_{{\rm th},\alpha})}{\phi(v_\alpha/v_{{\rm th,e}})}\frac{Z_\alpha^2 n_\alpha}{n_{\rm e}}, \tag{34}$$

$$\frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm i}}} = \frac{2}{1+\frac{m_\alpha}{m_{\rm i}}}\frac{\phi(v_\alpha/v_{{\rm th},\alpha})}{\phi(v_\alpha/v_{{\rm th,i}})}\frac{Z_\alpha^2 n_\alpha}{n_{\rm i}}, \tag{35}$$

where $v_{{\rm th},\beta} \equiv \sqrt{2T_\beta/m_\beta}$ is the thermal velocity of the species β, and $Z_\alpha = 2$ is the Helium charge number. For an ion at 3.5 MeV and $T_{\rm i} = T_{\rm e} = T_\alpha = 10$ keV, the ratios are approximately equal to

$$\frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm e}}} \simeq 0.01, \qquad \frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm i}}} \simeq 0.3 \tag{36}$$

and most of the collisions lead to heating the electrons, as is well-known. After the ions have slowed down to $5 \times v_{{\rm th},\alpha}$, the ratios are

$$\frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm e}}} \simeq 0.7 \qquad \frac{\nu_{\rm s}^{\alpha/\alpha}}{\nu_{\rm s}^{\alpha/{\rm i}}} \simeq 0.3, \tag{37}$$

so that self-collisions, although still smaller than collisions on main ions and on electrons, contribute to the slowing-down.

The point here is not that self-collisions have a large impact on the slowing-down of alpha particles, but rather, that it is better to check that the $T_{\rm eff}$ collision algorithm does not introduce spurious effects when the distribution has a large tail of fast particles. Indeed, the algorithm is applied in the same way regardless of the shape of the distribution function.

The distribution is initialized with $N_a = 10^5$ particles of a Maxwellian distribution at rest and with temperature $T_a = 1$,

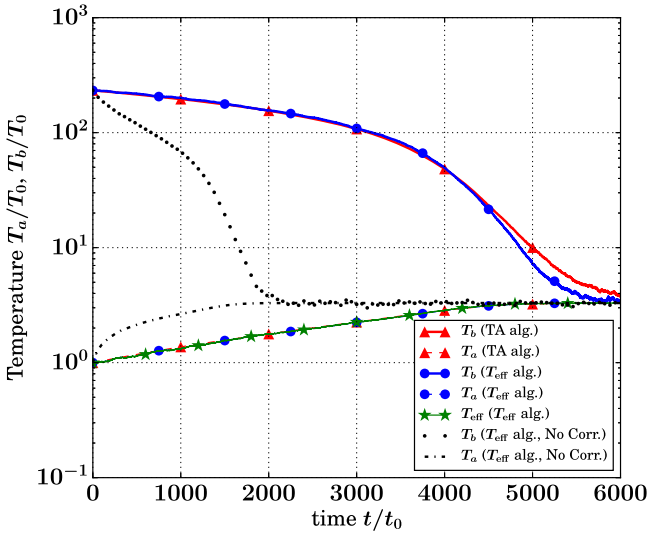

Figure 6. Relaxation of fast alpha particles on a cold bulk.

and an energy Dirac delta of $N_b = 10^3$ particles with isotropic velocity distribution at $\mathrm{v} = \sqrt{350}$. This corresponds to 1% of alpha particles at 3.5 MeV in a 10 keV plasma. With such parameters, the total energy of the distribution is significantly larger than the energy of the bulk f_a, despite the smallness of the fast particles density.

The comparison between the $T_{\rm eff}$ and TA algorithms is shown in figure 6. The time step is $\Delta t = 0.1$ for the $T_{\rm eff}$ algorithm and $\Delta t = 0.5$ for the TA algorithm. The temperature of the bulk T_a, of the alpha particles (more precisely the quantity $2/3\langle \mathrm{v}^2\rangle_{f_b}$) and the effective temperature $T_{\rm eff}$ are plotted. The dotted and dash-dotted curves represent the results of the $T_{\rm eff}$ algorithm when the correction to the friction, equation (24), is not applied. As anticipated in section 2, if only equation (25) is used, energy and momentum are conserved, but the price to pay is a large error in the energy transfer rate between the fast particles and the bulk. This error vanishes in the limit $\Delta t \to 0$, but it would require prohibitively small time steps. Instead, implementing equation (24) restores the correct transfer rates, and the time evolution of the distribution becomes virtually identical to the one of the TA algorithm.

Also remarkable is the fact that the effective temperature, $T_{\rm eff}$, follows closely T_a, the temperature of the bulk, during the whole evolution. That way, the relevant physical process is simulated: the fast particles, which are weakly collisional, and in particular almost do not interact with each other, mainly see the slow part of the distribution, which corresponds to f_a with temperature T_a. The small difference between $T_{\rm eff}$ and T_a ensures the conservation of energy, that is, that all the energy lost by the fast particles is transferred to the bulk.

6. Implementation in the XTOR-K code

We intend to implement the $T_{\rm eff}$ algorithm in the XTOR-K code. This requires to go a step further, from the 3D model detailed above to the full 6D implementation. There are three issues here. The first is parallelization. The second is that the algorithm involves sums on particles which are not located at the same point in space. The last issue is conservation of energy and momentum in the XTOR-K sense.

Regarding the first issue, parallelization means that the particles in a cell of the mesh do not *a priori* belong to the same process. As mentioned in the introduction, this is a serious issue in the case of the TA algorithm, because it means that the particles must be sorted and reaffected to the processes according to their positions, an operation that is very expensive in communications, CPU time and memory. However, the $T_{\rm eff}$ algorithm only involves sums over the particles, which can always be computed separately on each process, and then summed using message passing interface (MPI) routines such as `MPI_Reduce` or `MPI_Allreduce`. Thus, this first issue is easy to solve using the standard tools of parallel computing.

Regarding the second issue of non-locality, it is a matter of trade-off between the PIC noise and the approximation made by using particles at different locations to define the parameters of the effective Maxwellian distribution. If they are defined on a very small spatial scale, they are better localized, but the number of particles may not be large enough to obtain a sufficiently small PIC noise. Recall that the PIC noise is proportional to $1/\sqrt{N}$, where N is the number of samples. On the contrary, if we define them on a too large spatial scale, the PIC noise is efficiently reduced, but the spatial variations are no longer modelled accurately. A reasonable trade-off is to compute $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$ on the same (R, Z, φ) mesh that is used to compute the fluid moments of the kinetic particles. The cells of this mesh contain typically a few thousands to tens of thousands of particles, which is sufficient to obtain an acceptably small PIC noise. Also, as we have seen, in most cases of interest, $T_{\rm eff}$ is close to the temperature of the bulk of the distribution, so its characteristic scale length is the same as that of the bulk temperature and it varies smoothly on the mesh if the latter does. Note that in addition to $T_{\rm eff}$ and $\mathbf{V}_{\rm eff}$, an other parameter defines the Maxwellian distribution: its density $n_{\rm eff}$. The density $n_{\rm eff}$ is taken to be simply proportional to the number of particles in the cell.

The last issue of energy and momentum conservation arises because despite the exact energy and momentum conservation properties of the $T_{\rm eff}$ algorithm in the three dimensional velocity space, the fluid energy and momentum are not conserved in the XTOR-K sense. Indeed, the momentum and energy at a grid point (R_α, Z_β) in the (R, Z) space, and for a toroidal mode number[4] n are given by

$$\mathbf{j}_{\alpha\beta,n} = \sum_i W_{i,\alpha\beta,n}\mathbf{v}_i, \tag{38}$$

$$\mathcal{E}_{\alpha\beta,n} = \sum_i W_{i,\alpha\beta,n}\mathbf{v}_i^2, \tag{39}$$

where the sum is on all particles and the weight functions $W_{i,\alpha\beta,n}$, which depend on the positions of the particles,

[4] The moments are Fourier discretized in the toroidal direction.

quantify how much each particles is contributing to the moment in the cell. After a collision, the particle velocities are changed without their positions being affected, and one sees that there is no reason for $\mathbf{j}_{\alpha\beta,n}$ and $\mathcal{E}_{\alpha\beta,n}$ to be equal after the collision step to their value before collision. In order to conserve energy and momentum in this sense, it is possible to transform the velocities in the same spirit as equation (25), according to the following transformation:

$$\mathbf{v}_i' \leftarrow \lambda_i \mathbf{v}_i' + \boldsymbol{\mu}_i, \tag{40}$$

where primed quantities still denote the quantities after a collision step. In the 6D PIC code, equation (40) replaces equation (25). The quantities λ_i and $\boldsymbol{\mu}_i$ are linearly interpolated at the position (R, Z, φ) of the particles from the coefficients $\lambda_{\alpha\beta} = \sum_n \lambda_{\alpha\beta,n} \mathrm{e}^{-\mathrm{i}n\varphi}$ and $\boldsymbol{\mu}_{\alpha\beta} = \sum_n \boldsymbol{\mu}_{\alpha\beta,n} \mathrm{e}^{-\mathrm{i}n\varphi}$, so as to reduce the number of coefficients to compute and store.

The coefficients $\lambda_{\alpha\beta,n}$ and $\boldsymbol{\mu}_{\alpha\beta,n}$ are determined using the constraints $\mathbf{j}'_{\alpha\beta,n} = \mathbf{j}_{\alpha\beta,n}$ and $\mathcal{E}'_{\alpha\beta,n} = \mathcal{E}_{\alpha\beta,n}$. This results in a coupled nonlinear system, which we will now write explicitly. The linear interpolation gives the nth Fourier component of λ_i as

$$\begin{aligned}\lambda_{i,n} = &\sum_{\alpha\beta} (u_{i,\alpha\beta} t_{i,\alpha\beta} \lambda_{\alpha+1,\beta+1,n} \\ &+ (1 - u_{i,\alpha\beta}) t_{i,\alpha\beta} \lambda_{\alpha+1,\beta,n} \\ &+ u_{i,\alpha\beta} (1 - t_{i,\alpha\beta}) \lambda_{\alpha\beta+1,n} \\ &+ (1 - u_{i,\alpha\beta})(1 - t_{i,\alpha\beta}) \lambda_{\alpha\beta,n}) \\ \equiv &\sum_{\alpha\beta} I_{i,\alpha\beta} \lambda_{\alpha\beta,n},\end{aligned} \tag{41}$$

where $u_{i,\alpha\beta}$ (resp. $t_{i,\alpha\beta}$) is the coefficient of the linear interpolation in the R (resp. Z) direction (see figure 7). The coefficients $u_{i,\alpha\beta}$ and $t_{i,\alpha\beta}$ both vanish if the point (R_i, Z_i) is not inside the square (R_α, Z_β), $(R_{\alpha+1}, Z_\beta)$, $(R_{\alpha+1}, Z_{\beta+1})$, $(R_\alpha, Z_{\beta+1})$. For every particle indexed by i, only four coefficients $I_{i,\alpha\beta}$ are nonzero, and $\sum_{\alpha\beta} I_{i,\alpha\beta} = 1$.

From now on, we do not write the mode number index n, in order to obtain a lighter notation. It is understood that the system of equations we will obtain for the λ and $\boldsymbol{\mu}$ coefficients must be solved for every value of n. Using the transformed expression equation (40) for the new velocities, we obtain the following set of equations for the coefficients $\lambda_{\alpha\beta}$ and $\boldsymbol{\mu}_{\alpha\beta}$:

$$\sum_{\alpha'\beta'} (\lambda_{\alpha'\beta'} \mathcal{A}^J_{\alpha'\beta';\alpha\beta} + \boldsymbol{\mu}_{\alpha'\beta'} \mathcal{B}^J_{\alpha'\beta';\alpha\beta}) = \mathbf{j}_{\alpha\beta}, \tag{42}$$

$$\begin{aligned}\sum_{\alpha'\beta',\alpha''\beta''} &(\lambda_{\alpha'\beta'} \lambda_{\alpha''\beta''} \mathcal{A}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} \\ &+ 2\lambda_{\alpha''\beta''} \boldsymbol{\mu}_{\alpha'\beta'} \cdot \mathcal{B}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} \\ &+ \boldsymbol{\mu}_{\alpha'\beta'} \cdot \boldsymbol{\mu}_{\alpha''\beta''} \mathcal{C}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta}) = \mathcal{E}_{\alpha\beta}\end{aligned} \tag{43}$$

with

$$\mathcal{A}^J_{\alpha'\beta';\alpha\beta} = \sum_i \mathbf{v}_i' W_{i,\alpha\beta,n} I_{i,\alpha'\beta'}, \tag{44}$$

$$\mathcal{B}^J_{\alpha'\beta';\alpha\beta} = \sum_i W_{i,\alpha\beta,n} I_{i,\alpha'\beta'}, \tag{45}$$

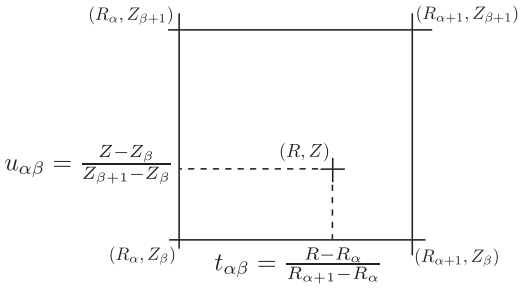

Figure 7. Definition of the coefficients for the linear interpolation.

$$\mathcal{A}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} = \sum_i \mathbf{v}_i'^2 W_{i,\alpha\beta,n} I_{i,\alpha'\beta'} I_{i,\alpha''\beta''}, \tag{46}$$

$$\mathcal{B}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} = \sum_i \mathbf{v}_i' W_{i,\alpha\beta,n} I_{i,\alpha'\beta'} I_{i,\alpha''\beta''}, \tag{47}$$

$$\mathcal{C}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} = \sum_i W_{i,\alpha\beta,n} I_{i,\alpha'\beta'} I_{i,\alpha''\beta''} \tag{48}$$

and we used the symmetry relation $\mathcal{B}^{\mathcal{E}}_{\alpha'\beta',\alpha''\beta'';\alpha\beta} = \mathcal{B}^{\mathcal{E}}_{\alpha''\beta'',\alpha'\beta';\alpha,\beta}$. Equations (42) and (43) are a coupled nonlinear system, which can be solved using standard iterative techniques, such as the Newton–Krylov method, which is already used in the solver of the fluid part of XTOR-K [4]. The initial guess $\lambda = 1$ and $\boldsymbol{\mu} = 0$ can be used in the iterative method. If some cells are not populated enough, they may cause the whole system to be singular. When this is the case, it is sufficient to remove the corresponding nodes from the system and to set $\lambda = 1$ on these nodes. As in the first part of the algorithm, the computation of T_{eff} and $\mathbf{V}_{\mathrm{eff}}$, determining the coefficients of the nonlinear system only requires `MPI_reduce` type communication, because they are obtained as a sum over all the particles in a cell (equations (44)–(48)).

Finally, we sum up the implementation of the T_{eff} algorithm in the XTOR-K case, including the communications between processes:

(i) On each process, the sums $\bar{\mathcal{G}}$, $\bar{\mathcal{H}}$, and their derivatives with respect to T and $\mathbf{V}$, are computed in every cell of the (R,Z) mesh for every toroidal mode n.
(ii) A call to `MPI_Reduce` sums the values of all the processes, and the values obtained are used at each iteration of the Newton process. At the end of this process, the values of n_{eff}, T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ are known at the center of the cells of the (R,Z) mesh for every toroidal mode n.
(iii) The values of n_{eff}, T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ are broadcasted to all the processes.
(iv) The friction and diffusion coefficients are computed, according to the value of n_{eff}, T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ at the location of each particle, obtained from n_{eff}, T_{eff} and $\mathbf{V}_{\mathrm{eff}}$ at the center of the cells of the mesh by interpolation. The friction is corrected according to equation (24). However exact conservation of momentum and energy is not ensured using equation (25) at this stage.

(v) For each value of the mode number n, the coefficients of the nonlinear system (42) and (43), $\mathcal{A}^J$, $\mathcal{B}^J$, $\mathcal{A}^{\mathcal{E}}$, $\mathcal{B}^{\mathcal{E}}$ and $\mathcal{C}^{\mathcal{E}}$, are computed on each process.
(vi) The total coefficients are obtained via `MPI_reduce`.
(vii) The nonlinear system is solved in parallel using iterative Newton–Krylov technique. The coefficients $\lambda_{\alpha\beta}$ and $\boldsymbol{\mu}_{\alpha\beta}$ are obtained at every node of the (R, Z) grid and broadcasted to all the processes.
(viii) Transformation equation (40) is applied to each particle using the local λ_i and $\boldsymbol{\mu}_i$, linearly interpolated from the grid quantities $\lambda_{\alpha\beta}$ and $\boldsymbol{\mu}_{\alpha\beta}$.

7. Discussion

To the authors' knowledge, the present approach has never been proposed, let alone used in a parallel full-f PIC code. Consequently, there remains a large number of unknowns regarding its applicability to the XTOR-K code. The present section lists the questions that will have to be clarified upon implementing the method in the code.

First, it is worth emphasizing that only a few cases for the shape of the distribution function have been studied here. They were chosen for their relevance (heating by a fast beam or by the alpha particles) and for the ease in initializing the distribution function with Maxwellian distributions. The results are found to be very good in these cases: the conditions for the model to reproduce the correct energy and momentum transfer rates, as well as the correct δf, are not very restrictive. Even when the conditions are not met, and the transfer rates are wrong by a large amount, the overall relaxation times are roughly correct (see figures 4 (c) and (d)). However, this does not guarantee that all possible shapes for the distribution function will give satisfying results provided that the density of δf is small. In particular, one of the main objectives of the collision module in the XTOR-K code is to reproduce the ion-driven neoclassical transport self-consistently. Neoclassical transport results from the collisional friction between passing and trapped particles. Passing particles can have a large parallel momentum, unlike trapped particles, which are constrained to have a slow precessional motion in the toroidal direction. This induces a discontinuity of the distribution function at the trapped/passing boundary in the $(v_\parallel, v_\perp)$ plane, which is relaxed by collisions. The specific shape of the distribution function is responsible for the detailed properties of the neoclassical transport, and the present study does not assure that neoclassical transport will be modelled correctly. Nonetheless, the good results obtained here allow for some optimism. Also, analytical formulæ exist in a lot of cases, so it will be possible to assess the quality of the results quantitatively when the collisions are implemented in the XTOR-K code.

Even before turning to neoclassical transport, the present study does not say anything about classical transport, which takes place accross a uniform magnetic field in the presence of density and temperature gradients. The benchmarking of the algorithm versus classical and neoclassical transport theory is left as future work.

Last but not least, we have not discussed in this paper the issue of multi-species collisions. The XTOR-K code is meant to evolve kinetically several different species of ions, while electrons are always kept in the fluid part of the code. Collisions on electrons can readily be handled in the following way: their velocity and temperature being available from the fluid, one merely needs to give the ions Langevin kicks for the corresponding electron Maxwellian, without the slightest consideration of energy and momentum conservation in a first step; in a second step, the energy and momentum lost by the ions is computed and entirely given to the electrons. The electron distribution function is supposed to remain Maxwellian all the time. If there are multiple kinetic ion species, a similar approach could be adopted, where for each species the Maxwellian considered is the effective Maxwellian found in the ion-collision step. Under certain conditions, such as on the density and/or mass discrepancies, it may even be possible to generalize the T_{eff} algorithm to multispecies collisions, so that all species collide on the same effective Maxwellian distribution. This is left as future work.

8. Conclusion

In this paper, we have shown that it is possible to write an energy and momentum conserving self-collision algorithm by requiring the particles to collide on a single Maxwellian distribution, which is shifted in temperature and velocity. These shifts are what allows to model the background reaction satisfactorily. In all the cases analyzed, including the interaction between shifted or non-shifted Maxwellian distributions with different temperatures, and a distribution with a tail of fast ions, the algorithm compares very well with the binary collision algorithm of [12], which is exact in the limit $N \to \infty, \Delta t \to 0$. Contrary to the latter, our algorithm can be implemented in a 6D domain cloning PIC code, without requiring to sort the particles belonging to different processes at each collision time step. Only calls to `MPI_Reduce` type routines with the operator sum are required in order to get all the quantities required by the algorithm. Compared to the local three-dimensional in velocity space algorithm, there is an additional computational price to pay to recover energy and momentum conservation in the six-dimensional case, which is the resolution of a large nonlinear system. We do not expect this price to be prohibitive.

Acknowledgments

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grand agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Appendix A. Expressions for $\mathcal{G}$, $\mathcal{H}$ and their derivatives

We detail here the expressions of $\mathcal{G}$ and $\mathcal{H}$, as well as their partial derivatives with respect to T and $\mathbf{V}$. The quantities $\bar{\mathcal{G}}$ and $\bar{\mathcal{H}}$, as well as the Jacobian of the system, are obtained by applying the operator $\langle\cdot\rangle_f$ on the respective quantities, in order to obtain the average over the distribution. We recall the notation $X = |\mathbf{v} - \mathbf{V}|/\sqrt{T}$.

The explicit expressions for $\mathcal{G}$ and $\mathcal{H}$ are:

$$\mathcal{G} = \frac{\frac{4}{\sqrt{\pi}} X \mathrm{e}^{-X^2} - \mathrm{erf}(X)}{\sqrt{T} X}, \tag{A.1}$$

$$\mathcal{H} = -(\mathbf{v} - \mathbf{V}) \frac{\phi(X)}{X^3 T^{3/2}}. \tag{A.2}$$

The derivatives of $\mathcal{G}$ are given by

$$\frac{\partial \mathcal{G}}{\partial T} = \frac{(4X^2 - 1)\mathrm{e}^{-X^2}}{\sqrt{\pi} T^{3/2}}, \tag{A.3}$$

$$\frac{\partial \mathcal{G}}{\partial \mathbf{V}} = \frac{8(\mathbf{v} - \mathbf{V})}{\sqrt{\pi} T^{3/2}} \left(\left(1 + \frac{1}{4X^2} \right) \mathrm{e}^{-X^2} - \frac{\sqrt{\pi}\, \mathrm{erf}(X)}{8X^3} \right). \tag{A.4}$$

The temperature derivative of $\mathcal{H}$ is given by

$$\frac{\partial \mathcal{H}}{\partial T} = \frac{2(\mathbf{v} - \mathbf{V})\mathrm{e}^{-X^2}}{\sqrt{\pi} T^{5/2}}. \tag{A.5}$$

The diagonal terms are given by

$$\frac{\partial \mathcal{H}_i}{\partial V_i} = -\left[\phi(X) \left(\frac{3(\mathrm{v}_i - V_i)^2}{X^2 T} - 1 \right) - \frac{4}{\sqrt{\pi}} (\mathrm{v}_i - V_i)^2 \frac{X}{T} \mathrm{e}^{-X^2} \right] \frac{1}{X^3 T^{3/2}}. \tag{A.6}$$

Finally, the off-diagonal terms, $\partial \mathcal{H}_i / \partial V_j$ with $i \neq j$, are given by

$$\frac{\partial \mathcal{H}_i}{\partial V_j} = \frac{(\mathrm{v}_i - V_i)(\mathrm{v}_j - V_j)}{\sqrt{\pi} X^5 T^{5/2}} \left[\frac{4}{\sqrt{\pi}} X^3 \mathrm{e}^{-X^2} - 3\phi(X) \right]. \tag{A.7}$$

References

[1] Cheng C 1991 *J. Geophys. Res.* **96** 21159
[2] Cheng C *et al* 1999 *J. Geophys. Res.* **104** 413
[3] Lütjens H *et al* 2015 *16th European Fusion Theory Conf. (Lisbon, Portugal)*
[4] Lütjens H *et al* 2010 *J. Comput. Phys.* **229** 8130
[5] Buneman O 1967 Time-reversible difference procedures *J. Comp. Phys.* **1** 517–35
[6] Porcelli F *et al* 1996 *Plasma Phys. Control. Fusion* **38** 2163
[7] Duong H *et al* 1993 *Nucl. Fusion* **33** 749
[8] Coppi B *et al* 1990 *Phys. Fluids* B **2** 927
[9] Sauter O *et al* 2002 *Phys. Rev. Lett.* **88** 105001
[10] Chapman I *et al* 2010 *Nucl. Fusion* **50** 102001
[11] Maget P *et al* 2014 *Phys. Plasmas* **21** 062504
[12] Takizuka T *et al* 1977 *J. Comput. Phys.* **25** 205
[13] Brunner S *et al* 1999 *Phys. Plasmas* **6** 4504
[14] Landau L 1936 *Phys. Z. Sowjetunion* **10** 154
[15] Braginskii S 1965 *Rev. Plasma Phys.* **1** 205
[16] Rosenbluth M N *et al* 1957 *Phys. Rev.* **107** 1
[17] Van Kampen N 1981 *J. Stat. Phys.* **24** 175
[18] Gardiner C W *et al* 1985 *Handbook of Stochastic Methods* (Berlin: Springer)
[19] Lin Z *et al* 1995 *Phys. Plasmas* **2** 2975
[20] Hazeltine R D *et al* 2003 *Plasma confinement* (Mineola, New York: Dover)
[21] Bergmann A 1998 *Control. Plasma Phys.* **38** 231
[22] Nanbu K 1997 *Phys. Rev.* E **55** 4642
[23] Bobylev A *et al* 2000 *Phys. Rev.* E **61** 4576
[24] Wang C *et al* 2008 *J. Comput. Phys.* **227** 4308
[25] Nanbu K 2000 *IEEE Trans. Plasma Sci.* **28** 971
[26] Huba J D and Plasma N R L 2007 *NRL Plasma Formulary* (Washington, DC: Naval Research Laboratory)
[27] Loarte A *et al* 2007 *Nucl. Fusion* **47** S203