A methodology for the rigorous verification of Particle-in-Cell simulations

Fabio Riva,[1,*] Carrie F. Beadle,[1] and Paolo Ricci[1]

[1]*École Polytechnique Fédérale de Lausanne (EPFL), Swiss Plasma Center (SPC), CH-1015 Lausanne, Switzerland*

Abstract

A methodology to perform a rigorous verification of Particle-in-Cell (PIC) simulations is presented, both for assessing the correct implementation of the model equations (code verification), and evaluating the numerical uncertainty affecting the simulation results (solution verification). The proposed code verification methodology is a generalization of the procedure developed for plasma simulation codes based on finite difference schemes that was described by Riva *et al.* [Phys. Plasmas **21**, 062301 (2014)] and consists of an order-of-accuracy test using the method of manufactured solutions. The generalization of the methodology for PIC codes consists of accounting for numerical schemes intrinsically affected by statistical noise and providing a suitable measure of the distance between continuous, analytical distribution functions and finite samples of computational particles. The solution verification consists of quantifying both the statistical and discretization uncertainties. The statistical uncertainty is estimated by repeating the simulation with different pseudorandom number generator seeds. For the discretization uncertainty, the Richardson extrapolation is used to provide an approximation of the analytical solution and the grid convergence index is used as an estimate of the relative discretization uncertainty. The code verification methodology is successfully applied to a PIC code that numerically solves the one-dimensional, electrostatic, collisionless Vlasov-Poisson system. The solution verification methodology is applied to quantify the numerical uncertainty affecting the two-stream instability growth rate, which is numerically evaluated thanks to a PIC simulation.

*Electronic address: `fabio.riva@epfl.ch`

I. INTRODUCTION

Originally developed to simulate fluid flows in two dimensions [1], the Particle-in-Cell (PIC) algorithm [2–6] is now a valuable tool to solve the Vlasov-Maxwell system of equations. The PIC algorithm approximates the distribution function with a set of computational particles that are evolved in time according to Newton's laws, and computes self-consistently the electric and magnetic fields acting on the particles by solving the Maxwell's equations. While conceptually simple in their basic formulations, the development of PIC simulation methods has significantly increased their range of applicability, accuracy, and performance. Energy, momentum, and charge conserving algorithms have been developed [7–9], which, also within an implicit-time discretization [10–14], allowed progress from the solution of one-dimensional, electrostatic models to the simulation of complex and realistic three-dimensional electromagnetic systems. Thanks to PIC simulations, significant progress has been made in the understanding of fundamental plasma phenomena, such as collisionless shocks (see e.g. Refs. [15–17]), magnetic reconnection (see e.g. Refs. [18–20]), laser-plasma interactions (see e.g. Refs. [21–23]), and the plasma-wall transition (see e.g. Refs. [24, 25]). Despite the widespread use of PIC codes, the methodologies to rigorously assess the correct implementation of the PIC model into simulation codes (known as code verification) and estimate the numerical error affecting the simulation results (known as solution verification) are not well established. This is becoming a very crucial issue, since errors affecting PIC simulations, which are used to uncover complex plasma phenomena and, for example, predict performances of future nuclear facilities, can have far reaching consequences [26, 27]. The usual approaches to verify PIC simulations and evaluate the error affecting a simulation result are based on performing code-to-code comparisons (see e.g. Refs. [20, 28–32]). Simulations of very simple problems, such as the Landau damping of electron-plasma waves, for which an analytical solution is known, are also used. While valuable, these tests are not rigorous enough to ensure the correct implementation of the model in the simulation code or to estimate the numerical uncertainty affecting simulation results, due to difficulties such as understanding if differences in the numerical results are introduced by the finite resolution of the grid used for the discretization or by errors in the implementation of the codes [33]. On the other hand, analytical solutions in complex geometries or with complex collision operators do not generally exist. The goal of the present paper is to generalize

the rigorous code and solution verification methodologies developed for grid-based plasma simulation codes and presented in Ref. [34], to the verification of plasma simulation codes based on the PIC algorithm.

Order-of-accuracy tests allow the rigorous assessment of the correct implementation of grid-based numerical algorithms. In an order-of-accuracy test, the rate of convergence of the numerical solution to an exact solution is compared with the expected order of accuracy of the numerical scheme. If they agree, the code is verified [33]. Since the exact solution of the model equations is unknown in most cases, the method of manufactured solutions (MMS) was developed by the computational fluid dynamics community [35–38]. This method was recently ported to the plasma physics community [34], and it is now routinely used for the verification of grid-based plasma turbulence codes (see e.g. Refs. [39, 40]).

The correct implementation of the model equations in the simulation code does not imply that numerical results are free from numerical errors. Sources of numerical errors are: rounding off, finite statistical sampling (e.g. using a finite number of particles randomly distributed to represent a distribution function), termination of an iterative scheme with a non-vanishing residue, and finite grid resolution [33]. The evaluation of these errors is the objective of solution verification procedures. Since for grid-based algorithms the discretization error component usually dominates, a procedure for the solution verification was proposed based on the Richardson extrapolation [41, 42]. As a matter of fact, the Richardson extrapolation converges faster to the exact solution than the simulation results, and therefore it is used as higher order estimator of the exact solution in computing the discretization error. However, since the assumptions required to use the Richardson extrapolation as a solution estimator are difficult to satisfy, the Roache grid convergence index (GCI) was introduced as a relative numerical uncertainty estimate [43].

The present paper generalizes the methodology discussed in Ref. [34], providing a rigorous methodology for the code and solution verification of PIC simulations. For the code verification, we propose an order-of-accuracy test for PIC simulation codes, developing the MMS to account for numerical schemes intrinsically affected by statistical noise, and providing a measure of the distance between continuous, analytical distribution functions, and finite samples of computational particles. For the solution verification, we discuss how to estimate the statistical uncertainties affecting the numerical results and, using the Richardson extrapolation as a higher order estimator of the exact solution and the GCI as a relative

numerical uncertainty estimate, we provide an evaluation of the discretization error. Both methodologies are applied to a simple, one-dimensional, collisionless, electrostatic PIC simulation code, showing the peculiarities and the potential of the proposed procedures.
This paper is structured as follows. After the Introduction, in Sec. II we present the PIC algorithm used to numerically solve the plasma kinetic equations. Then, in Sec. III we illustrate the MMS, we explain how it is adapted to account for PIC codes, and we discuss several measures of the distance between continuous, analytical distribution functions and finite samples of computational particles. The solution verification methodology is presented in Sec. IV, where we describe how to estimate the statistical uncertainty and the discretitazion error affecting PIC simulation results. In Sec. V we present the simulation code that we verify with the MMS, and we apply the solution verification methodology to quantify the numerical uncertainty affecting the two-stream instability growth rate, which is evaluated from a PIC simulation. Our conclusions are reported in Sec. VI. Finally, Appendix A shows that the proposed distances between a continuous, analytical distribution functions, and a finite sample of computational particles are suitable for performing a PIC code verification with the MMS.

II. THE PARTICLE-IN-CELL METHOD

In the present paper we consider PIC codes used to numerically solve the Vlasov-Maxwell system of equations. The PIC algorithm represents the distribution function of plasma species as a set of computational particles (also known as superparticles or markers), whose position in the phase space is evolved according to Newton's laws. The forces acting on the particles are obtained by solving the Maxwell equations, having assigned to a numerical grid the charge and the current carried by the particles [2–6].
As the goal of the present paper is to introduce a rigorous methodology for the verification of PIC simulation codes, we consider the simplest kinetic model describing a one-dimensional, electrostatic, collisionless plasma in a periodic domain. The generalization to the collisional, electromagnetic, three-dimensional case does not present conceptual difficulties. The model

we consider is written

$$\frac{\partial f_\alpha}{\partial t} + v\frac{\partial f_\alpha}{\partial x} + \frac{q_\alpha}{m_\alpha}E\frac{\partial f_\alpha}{\partial v} = 0 \tag{1}$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}, \tag{2}$$

where $f_\alpha(x, v, t)$ is the distribution function for the α species ($\alpha = e$ for electrons and $\alpha = i$ for ions), q_α and m_α are the particle charge and mass, $\rho(x,t) = \sum_\alpha q_\alpha \int_{-\infty}^{+\infty} f_\alpha(x, v, t)dv$ is the total charge distribution and $E(x,t)$ is the electric field. As $m_i \gg m_e$, ions can be assumed at rest as a first approximation, with the ion plasma density $n_i = \int_{-\infty}^{+\infty} f_i dv$ constant in time and uniform along x. In the remainder of this paper we use this approximation and we consider only the evolution of the electron distribution function (we drop the α index).

The PIC method solves numerically Eqs. (1)-(2) by performing the following steps. (i) At $t = 0$, N computational particles are randomly distributed in the phase space according to a distribution function $f_0(x, v)$, and a weight w_p is assigned to each particle, with $w_p = f(x_p, v_p, t = 0)/f_0(x_p, v_p)$ [if $f_0(x, v) = f(x, v, t = 0)$ all markers have the same weight]. (ii) The particle charge is assigned to a numerical grid with spacing Δx, to obtain the charge distribution at each grid point. (iii) Poisson's equation [Eq. (2)] is solved and the electric field E is computed on the grid. (iv) E is interpolated from the grid to the particle positions, to obtain the electric field E_p acting on each particle. (v) The equations of motion of the computational particles

$$\frac{dw_p}{dt} = 0 \qquad \frac{dx_p}{dt} = v_p \qquad \frac{dv_p}{dt} = \frac{q}{m}E_p, \tag{3}$$

are numerically integrated in time to $t = \Delta t$, with Δt the step of the time integration scheme. The distribution function is now known at $t = \Delta t$ and, following the steps (ii)-(v), the system is advanced until the final time of the simulation is reached.

Noting that the error associated with a statistical representation of the distribution function is expected to decrease as $N^{-1/2}$ [44, 45], the numerical error affecting quantities that result from a simulation such as f and E_p is

$$\epsilon = C_1\Delta x^\alpha + C_2\Delta t^\beta + C_3 N^{-1/2} + \text{higher order terms}. \tag{4}$$

where C_1, C_2, and C_3 are constants independent of Δx, Δt, and N; α denotes the order of accuracy of the spatial operators in the interpolation between particles and grid positions and in the solution of the Poisson equation; and β is the order of accuracy of the time

integration scheme.

To simplify the expression of the numerical error, it is useful to introduce the theoretical order of accuracy of the algorithm, p, and a parameter h representing the degree of refinement of the mesh and time step, and the number of markers in the system, defined as

$$h^p = \left(\frac{\Delta x}{\Delta x_0}\right)^{\alpha} = \left(\frac{\Delta t}{\Delta t_0}\right)^{\beta} = \left(\frac{N}{N_0}\right)^{-1/2}, \tag{5}$$

where Δx_0, Δt_0, and N_0 are reference parameters for a standard simulation. Consequently, from Eq. (4) we obtain

$$\epsilon_h = C_p h^p + \mathcal{O}\left(h^{p+1}\right), \tag{6}$$

where C_p is a constant independent of h. In the following, we consider $p = \alpha$, i.e. we define the theoretical order of accuracy of the algorithm as the order of accuracy of the spatial discretization scheme.

III. CODE VERIFICATION

Code verification is usually approached by [33]: (a) performing simple tests (e.g., energy conservation tests), (b) comparing simulation results with results from other codes (also known as code-to-code benchmark), (c) quantifying the numerical error with respect to the exact solution, (d) testing the convergence of the numerical solution to the exact solution, and (e) comparing the rate of convergence of the numerical solution to the expected order of the numerical scheme (order-of-accuracy tests). As the first two procedures [(a) and (b)] do not require an exact solution of the model equations, they are simpler to perform. Indeed, code-to-code comparison is routinely performed to verify numerical codes used in plasma physics, including PIC simulation codes [20, 28–32]. On the other hand, the other three approaches [(c)-(e)] are more rigorous, but they require an exact solution of the model. The order-of-accuracy test is the only code verification procedure able to ensure the correct implementation of the numerical scheme into a simulation code, and therefore the correct solution of the model equations [33].

Formally, an order of accuracy test for a PIC code can be stated as follows. Given the kinetic model M solved by the PIC code, we denote its exact solution as s [$M(s) = 0$] and its numerical discretization with degree of refinement h as M_h. Moreover, the numerical solution of M_h is denoted as s_h [$M(s_h) = 0$], and the numerical error affecting the simulation

results is defined as

$$\epsilon_h = \|s - s_h\|, \tag{7}$$

where $\|\cdot\|$ denotes a designated norm. By evaluating the two numerical solutions of M_h and M_{rh}, s_h and s_{rh}, where rh indicates coarsening the parameter h by a factor r, one can evaluate the observed order of accuracy, $\hat{p}$, using

$$\hat{p} = \frac{\ln\left(\epsilon_{rh}/\epsilon_h\right)}{\ln\left(r\right)}. \tag{8}$$

If $\hat{p}$ converges to p for $h \to 0$, i.e. when the numerical error is dominated by the lowest order term of its Taylor expansion (the so-called asymptotic regime), we can state that the PIC code is verified and the equations are correctly solved, with the order of accuracy expected for the numerical scheme.

A. The method of manufactured solutions

The evaluation of the numerical error ϵ_h, necessary to obtain $\hat{p}$, requires that s is known. Unfortunately, s is unknown in most cases, in particular for complex kinetic problems. The MMS was developed to overcome this issue, and approaches the problem as follows [35–38]: instead of solving M analytically, an arbitrary function s_M is imposed as a solution to the model (the so-called manufactured solution), and the model equations are modified to accommodate the imposed solution; the modified model is then solved numerically to compute the numerical error. More precisely, for a given model M, we choose an analytical function s_M and compute a source term, $S = M\left(s_M\right)$, which is subsequently subtracted from M to obtain a new analytical model G [$G = M - S$]. The analytical solution of G is s_M: $G\left(s_M\right) = M\left(s_M\right) - S = 0$. It is then straightforward to compute the discretization of G, $G_h = M_h - S$, which can be solved numerically to obtain $s_{M,h}$. Since the source term S is computed analytically, we do not add any new source of numerical error to the original numerical model, and the numerical error $\epsilon_h = \|s_M - s_{M,h}\|$ satisfies

$$\epsilon_h = C' h^p + \mathcal{O}\left(h^{p+1}\right), \tag{9}$$

where C' is a constant independent of h. By showing that $\hat{p} \to p$ for $h \to 0$, one verifies the simulation code.

We remark that the manufactured solution should satisfy the following requirements [33]:

(i) be sufficiently smooth and not singular, (ii) satisfy the code constraints (e.g. $f \geq 0$ and $f \to 0$ for $v \to \pm\infty$), (iii) be general enough to excite all terms present in the equations, and (iv) ensure that the different terms composing the equations are of the same order of magnitude so that no term dominates the others. Due to these constraints, the manufactured solutions are usually built as a combination of trigonometric and/or hyperbolic functions. We remark that the code verification is a purely mathematical issue and therefore the choice of s_M is independent of the physical solution of M.

B. Verification of a PIC simulation code using the method of manufactured solutions

The verification of a PIC simulation code with the MMS is not straightforward, as it implies the comparison of a continuous, analytical distribution function with a sample of computational particles affected by statistical noise. In this sub-section we propose a methodology to perform this comparison.

First, the manufactured solutions E_M and f_M are chosen, and the corresponding source terms to be added on the right-hand side of Eqs. (1)-(2) are computed according to

$$S_f(x,v,t) = \frac{\partial f_M}{\partial t} + v\frac{\partial f_M}{\partial x} + \frac{qE_M}{m}\frac{\partial f_M}{\partial v} \tag{10}$$

and

$$S_E(x,t) = \frac{\partial E_M}{\partial x} - \frac{\rho}{\epsilon_0}, \tag{11}$$

with $S_E = 0$ if E_M is chosen consistently with f_M. While adding S_E in the Poisson equation does not present any conceptual difficulty (see Ref. [34] for a detailed discussion of the verification of grid-based equations with the MMS), adding a source term to the Vlasov equation requires the evolution in time of the computational particle weights, w_p, and the modification of Eq. (3) accordingly [46]. More precisely, the particles are initially distributed with a pseudo-random number generator according to a chosen distribution function $f_0(x,v)$ and the initial weights are set as $w_p(0) = f_M\left[x_p(0), v_p(0), 0\right] / f_0\left[x_p(0), v_p(0)\right]$. The weights w_p are then evolved according to

$$\frac{dw_p}{dt} = \frac{S_f[x_p(t), v_p(t), t]}{f_0\left[x_p(0), v_p(0)\right]}. \tag{12}$$

We remark that, in the presence of a collision operator, the marker distribution is not conserved along particle trajectories and Eq. (12) should be generalized according to Ref. [46].

To avoid altering the convergence properties of the numerical scheme, Eq. (12) has to be integrated in time by using a numerical scheme with order of accuracy β or greater.

We now define the norms used to compute the numerical error affecting the simulation results. For the electric field, this does not present any particular difficulty, and we indicate the distance between the numerical and the manufactured solution as

$$\epsilon(E_p) = \max_t \max_{p=1,...,N} |E_p(t) - E_M[x_p(t), t]|. \tag{13}$$

On the other hand, the definition of the norm used to quantify the numerical error affecting f_M requires measurement of the distance between a continuous analytical distribution function and a set of N computational particles.

The comparison of a data set of N elements to a distribution function is a general mathematical issue that appears in many research fields [47, 48]. For a one-dimensional probability density function $g(x)$, a data set can be compared to g considering the cumulative distribution function (CDF) $G(x) = \int_{-\infty}^{x} g(x')dx'$ and the empirical distribution function (EDF) related to the data set $G_N(x) = \sum_{p=1}^{N} \theta(x - x_p)/N$, where x_p are the elements of the data set, $p = 1, ..., N$ is the particle index and $\theta(x)$ is the Heaviside step function [$\theta(x) = 0$ if $x < 0$, and $\theta(x) = 1$ otherwise]. Under the null hypothesis, i.e. $\{x_p\}_{p=1,..,N}$ is a set of N random realizations of the distribution function g, and in the limit $N \to \infty$, the distance $D_N = \sup_{x\in\mathbb{R}} |G(x) - G_N(x)|$ converges to 0 as $\mathcal{O}(N^{-1/2})$ [49], where the supremum is used rather than the maximum since $G_N(x)$ is a piecewise continuous function.

To generalize this result to $d > 1$ dimensions, Peacock developed a method, detailed in Ref. [50], which is used to evaluate the distance between a multidimensional distribution function and an observed sample of N elements. For a two-dimensional distribution function $f_M(x, v, t)$ and a data set of N elements $\{x_p(t), v_p(t)\}_{p=1,...,N}$ of equal weight, at a given time t (in the reminder of this section we drop the t dependence to simplify the notation), Peacock's methodology requires one to define the four CDFs

$$\begin{aligned} F^1(x,v) &= \frac{1}{n}\int_{-\infty}^{x}\int_{-\infty}^{v} f_M(x',v')dx'dv' \qquad & F^2(x,v) &= \frac{1}{n}\int_{x}^{+\infty}\int_{-\infty}^{v} f_M(x',v')dx'dv' \\ F^3(x,v) &= \frac{1}{n}\int_{x}^{+\infty}\int_{v}^{+\infty} f_M(x',v')dx'dv' \qquad & F^4(x,v) &= \frac{1}{n}\int_{-\infty}^{x}\int_{v}^{+\infty} f_M(x',v')dx'dv' \end{aligned} \tag{14}$$

and the four EDFs

$$\begin{aligned} F_N^1(x,v) &= \sum_{p=1}^{N} \frac{1}{N}\theta(x-x_p)\theta(v-v_p) \qquad F_N^2(x,v) = \sum_{p=1}^{N} \frac{1}{N}\theta(x_p-x)\theta(v-v_p) \\ F_N^3(x,v) &= \sum_{p=1}^{N} \frac{1}{N}\theta(x_p-x)\theta(v_p-v) \qquad F_N^4(x,v) = \sum_{p=1}^{N} \frac{1}{N}\theta(x-x_p)\theta(v_p-v), \end{aligned} \tag{15}$$

and compute the largest difference between F^i and F_N^i $(i = 1, 2, 3, 4)$,

$$d_i^P = \sup_{(x,v)\in\mathbb{R}^2} |F^i(x,v) - F_N^i(x,v)|. \tag{16}$$

The distance between $f_M(x,v)$ and $\{x_p, v_p\}_{p=1,...,N}$ is thus defined as

$$\epsilon_P\left(f_M\right) = \max\left(d_1^P, d_2^P, d_3^P, d_4^P\right). \tag{17}$$

Reference [50] shows empirically that $\epsilon_P\left(f_M\right)$ decreases as $\mathcal{O}(N^{-1/2})$, irrespective of the choice of f_M, if $\{x_p, v_p\}_{p=1,...,N}$ is a set of random realizations of f_M.

To verify a PIC simulation code with MMS, one has to account for arbitrary values of w_p, and the definition of the F_N^i, Eq. (15), should be modified as follows

$$\begin{aligned} F_N^1(x,v) &= \sum_{p=1}^{N} \hat{w}_p\theta(x-x_p)\theta(v-v_p) \qquad F_N^2(x,v) = \sum_{p=1}^{N} \hat{w}_p\theta(x_p-x)\theta(v-v_p) \\ F_N^3(x,v) &= \sum_{p=1}^{N} \hat{w}_p\theta(x_p-x)\theta(v_p-v) \qquad F_N^4(x,v) = \sum_{p=1}^{N} \hat{w}_p\theta(x-x_p)\theta(v_p-v), \end{aligned} \tag{18}$$

with $\hat{w}_p = w_p / \sum_{p=1}^{N} w_p$. We show empirically (see Appendix A) that, if one defines the EDFs according to Eq. (18), $\epsilon_P\left(f_M\right)$ still decreases as $N^{-1/2}$ for $N \to \infty$.

We remark that $\epsilon_P\left(f_M\right)$ is affected by statistical uncertainty due to the random initialization of the markers. Consequently, the observed order of accuracy $\hat{p}$ obtained using $\epsilon_h = \epsilon_P\left(f_M\right)$ in Eq. (8) is also affected by statistical uncertainty. To perform an order of accuracy test, it is therefore necessary to carry out a number, n_s, of simulations with different pseudorandom number generator seeds, and compute the numerical error $\epsilon_{h,i} = \epsilon_P\left(f_M\right)$ for each simulation, with $i = 1, ..., n_s$. Then, following the methodology discussed in Sec. IV A, it is possible to approximate the expected value of ϵ_h with

$$\epsilon_h \simeq \frac{1}{n_s}\sum_{i=1}^{n_s} \epsilon_{h,i} \tag{19}$$

and the corresponding statistical uncertanty with

$$\Delta\epsilon_h = 1.96\frac{\sigma_{\epsilon_h}}{\sqrt{n_s}}, \tag{20}$$

where $\sigma_{\epsilon_h} = \sqrt{\sum_{i=1}^{n_s}(\epsilon_h - \epsilon_{h,i})^2/(n_s-1)}$ is the standard deviation corresponding to the distribution of $\epsilon_{h,i}$. Finally, the expected value of $\hat{p}$ is computed combining Eq. (19) with Eq. (8), and the corresponding statistical uncertainty is obtained as

$$\Delta\hat{p} = \frac{1}{\ln(r)}\sqrt{\left(\frac{\Delta\epsilon_h}{\epsilon_h}\right)^2 + \left(\frac{\Delta\epsilon_{rh}}{\epsilon_{rh}}\right)^2}. \tag{21}$$

C. Reducing the computational cost of a PIC verification

The evaluation of $\epsilon_P\,(f_M)$ is computationally expensive for a data set with a large number of elements. In fact, since $F_N^i(x,v)$ is a discontinuous function, the classical methods applied to compute the maximum value of a continuous function are not suitable. Moreover, the local maxima of the difference $|F^i(x,v) - F_N^i(x,v)|$ are found at the N^2 points (x_{p_j}, v_{p_k}), with $p_j = 1, ..., N$ and $p_k = 1, ..., N$. Therefore, to compute d_i^P, one has to evaluate the limits

$$d_{j,k}^i = \lim_{x\to x_{p_j}^{\pm}} \lim_{v\to v_{p_k}^{\pm}} |F^i(x,v) - F_N^i(x,v)| \tag{22}$$

for all (x_{p_j}, v_{p_k}) and then impose $d_i^P = \max_{j,k} d_{j,k}^i$.

Reference [51] shows that the value of $F_N^i(x,v)$, where (x,v) is a general point of the phase-space, can be evaluated with a brute force algorithm, or partitioning the points (x_p, v_p) in a k-d tree or in a range-counting tree. It results that the complexity of computing d_i^P according to Eq. (16) is: $\mathcal{O}(N^3)$ with the brute force algorithm, $\mathcal{O}(N^{5/2})$ with a k-d tree partitioning, and $\mathcal{O}(N^2 \log N)$ with a range-counting tree partitioning. However, the memory used to partition the N points with a range-counting tree scales as $\mathcal{O}(N \log N)$, while it scales as $\mathcal{O}(N)$ for a k-d tree partitioning or brute-force algorithm. All the evaluations of the EDF used to obtain the results presented in this paper are performed using a k-d tree partitioning, which in our opinion is the best compromise between computational cost and memory needs. To decrease the computational cost of the computation of d_i^P, Fasano and Franceschini propose an alternative approach [52], which approximates d_i^P as

$$d_i^P \simeq d_i^{FF} = \max_{p=1,...,N} \lim_{x\to x_p^{\pm}} \lim_{v\to v_p^{\pm}} |F^i(x,v) - F_N^i(x,v)|, \tag{23}$$

where the F_N^i are evaluated according to Eq. (15). Reference [52] shows empirically that the value of $\epsilon_{FF}\,(f_M)$ decreases as $\mathcal{O}(N^{-1/2})$ if we define $\epsilon_{FF}(f_M) = \max\left(d_1^{FF}, d_2^{FF}, d_3^{FF}, d_4^{FF}\right)$, where $\{x_p, v_p\}_{p=1,...,N}$ is a set of random realizations of f_M. We show empirically (see App. A)

that, if one defines the EDFs according to Eq. (18), $\epsilon_{FF}(f_M)$ still decreases as $N^{-1/2}$ for $N \to \infty$. The computational cost of evaluating d_i^{FF} is reduced by a factor N with respect to d_i^P.

As the computational cost of evaluating d_i^{FF} remains very demanding for high values of N, we discuss here an alternative method used to approximate d_i^P. Instead of maximizing $|F^i(x,v) - F_N^i(x,v)|$ over all points (x_p, v_p), as done according to the Fasano and Franceschini approach, one can generate M random points (x_j, v_j), with $j = 1, ..., M$, and approximate d_i^P with

$$d_i^P \simeq d_i^{MC} = \max_{j=1,...,M} |F^i(x_j, v_j) - F_N^i(x_j, v_j)|. \tag{24}$$

This approximation is a true equality in the limit $M \to \infty$, and corresponds to evaluating Eq. (16) with the Monte-Carlo method. We can therefore compute the distance between $f_M(x,v)$ and $\{x_p, v_p\}_{p=1,...,N}$ as

$$\epsilon_{MC}(f_M) = \max\left(d_1^{MC}, d_2^{MC}, d_3^{MC}, d_4^{MC}\right). \tag{25}$$

The evaluation of d_i^{MC} is computationally N^2/M times less demanding than d_i^P and N/M times less demanding than d_i^{FF}.

To further reduce the computational cost of performing a PIC code verification, in the rest of the present paper we also investigate the comparison of $F^i(x,v)$ with $F_N^i(x,v)$ only at $x = \pm\infty$ and $v = \pm\infty$, i.e. evaluating the supremum of the difference $|F^i(x,v) - F_N^i(x,v)|$ only over the boundaries of the phase-space domain. More precisely, we define the two errors

$$\epsilon_x(f_M) = \sup_{x\in\mathbb{R}} \left| \int_{-\infty}^{x} \left[\int_{-\infty}^{+\infty} f_M(x', v) dv \right] dx' - \sum_{p=1}^{N} \hat{w}_p \theta(x - x_p) \right| \tag{26}$$

$$\epsilon_v(f_M) = \sup_{v\in\mathbb{R}} \left| \int_{-\infty}^{v} \left[\int_{-\infty}^{+\infty} f_M(x, v') dx \right] dv' - \sum_{p=1}^{N} \hat{w}_p \theta(v - v_p) \right| \tag{27}$$

and assess whether they decrease according to the order of accuracy expected for the numerical scheme.

IV. SOLUTION VERIFICATION

Even if a model is correctly implemented in a simulation code, numerical errors always affect the simulation results [33]. Estimating the amplitude of these errors is a crucial issue,

not only to ensure the reliability of the numerical results, but also to quantify the uncertainty of the simulations when performing a rigorous validation of the physical model with experimental results. The evaluation of the numerical error affecting the simulation results is the objective of the solution verification procedure.

The simulation results are affected by round-off, iterative, statistical sampling, and discretization errors [33]. Round-off errors are due to the finite number of digits that computers use when representing numerical values. Assuming that all the computations are performed in double precision, round-off errors are usually negligible with respect to the other sources of errors (we assume that this is the case in the remainder). Iterative errors are due to the use of iterative numerical schemes terminated with a finite residue. This source of error can be reduced by increasing the number of iterations and it is neglected here. Statistical sampling errors, due to the random initialization of the markers, and discretization errors, due to the use of a finite grid resolution, a finite time step, and a finite number of computational particles, cannot be neglected in general. Moreover, as the analysis of the simulation results is generally performed using post-processing tools (e.g. the linear growth rate of an instability is usually obtained with an exponential fit), the solution verification procedure should also quantify the numerical uncertainty introduced by these tools. The sum of statistical sampling errors, discretization errors, and uncertainties introduced by post-processing tools constitute the numerical uncertainty affecting the simulation results.

A. Statistical error

The numerical results obtained with a PIC simulation code are always affected by statistical uncertainty. In fact, the distribution functions are approximated with a finite number of computational particles, which are initialized randomly according to a defined initial distribution function, by using a pseudorandom number generator. Moreover, PIC simulation codes often make use of operators based on pseudorandom number generators (e.g. when a collision term is added to the Vlasov equation). This introduces another source of statistical uncertainty.

To estimate the statistical uncertainty affecting X_h, where X_h is a point-by-point solution value or a solution functional evaluated from a simulation with discretization parameter h, we proceed as follows. We repeat the simulation n_s times with the same h, but changing

the pseudorandom number generator seed, and we define

$$\bar{X}_h = \frac{1}{n_s} \sum_{i=1}^{n_s} X_{h,i}, \tag{28}$$

where $X_{h,i}$ is the i^{th} evaluation of X_h and $i = 1, ..., n_s$. Assuming that the $X_{h,i}$ are randomly distributed from an unknown probability distribution with unknown but finite mean $\mu_{X,h}$ and variance $\sigma^2_{X,h}$, then $\bar{X}_h \to \mu_{X,h}$ for $n_s \to \infty$. Moreover, according to the central limit theorem, the distribution of $\bar{X}_h$ converges to the normal distribution with mean $\mu_{X,h}$ and variance $\sigma^2_{X,h}/n_s$ for $n_s \to \infty$. Therefore, for $n_s \to \infty$,

$$\mu_{X,h} - \frac{1.96\sigma_{X,h}}{\sqrt{n_s}} \leq \bar{X}_h \leq \mu_{X,h} + \frac{1.96\sigma_{X,h}}{\sqrt{n_s}} \tag{29}$$

with probability equal to 0.95. As a consequence, $\bar{X}_h$ can be used as an estimator of X_h, and we compute the uncertainty on this value as

$$\Delta X_h^{stat} = \frac{1.96\sigma_{X,h}}{\sqrt{n_s}}. \tag{30}$$

We remark that the unknown $\sigma^2_{X,h}$ can be estimated according to

$$\sigma^2_{X,h} = \frac{1}{n_s - 1} \sum_{i=1}^{n_s} \left(X_{h,i} - \bar{X}_h \right)^2. \tag{31}$$

Equations (28) and (30) provide a rigorous estimate of X_h and of its statistical uncertainty. However, due to the high computational cost of PIC simulations, n_s is typically low. To still have a realistic estimate of the statistical uncertainty, one can run n_s simulations with a smaller number of particles, $N' < N$, and evaluate the corresponding variance, $(\sigma_{X,h'})^2$, according to Eq. (31). Then, assuming that the statistical uncertainty is proportional to $N^{-1/2}$, the statistical error for a single simulation carried out with N particles can be estimated as

$$\Delta X_h^{stat} = 1.96\sigma_{X,h'} \sqrt{\frac{N'}{N}}. \tag{32}$$

B. Discretization error

Since PIC codes make use of discretized spatial grids, finite integration time steps, and a finite number of markers, their results are always affected by discretization errors. A rigorous methodology for the evaluation of the discretization error is detailed in Ref. [34],

and its main features are summarized here. Defining the Richardson extrapolation [41, 42] as

$$\hat{X} = \bar{X}_h + \frac{\bar{X}_h - \bar{X}_{rh}}{r^p - 1}, \tag{33}$$

then $|X - \hat{X}| = \mathcal{O}(h^{p+1})$, where X is the exact solution of the physical model and p is the order of accuracy of the numerical scheme (i.e., $\hat{X}$ converges to X faster than X_h for $h \to 0$). Consequently, we can use $\hat{X}$ as higher order estimator of X and approximate the discretization error as

$$\Delta X_h^{disc} \simeq \left\| \bar{X}_h - \hat{X} \right\| = \left\| \frac{\bar{X}_{rh} - \bar{X}_h}{r^p - 1} \right\|, \tag{34}$$

and the relative discretization error (RDE) as

$$RDE = \frac{\bar{X}_h - X}{X} \simeq \frac{\bar{X}_h - \hat{X}}{\hat{X}} = \frac{\bar{X}_{rh} - \bar{X}_h}{\bar{X}_h r^p - \bar{X}_{rh}}. \tag{35}$$

We remark that, for $\hat{X}$ to be a reasonable estimate of X, several assumptions should be satisfied [33]. First, the Richardson extrapolation method requires that the degree of mesh refinement can be represented solely by the parameter h. Second, the simulations used to evaluate $\hat{X}$ should be in the asymptotic regime, that is $p \simeq \hat{p}$, where

$$\hat{p} = \frac{\ln\left[\left(\bar{X}_{r^2h} - \bar{X}_{rh}\right) / \left(\bar{X}_{rh} - \bar{X}_h\right)\right]}{\ln(r)}. \tag{36}$$

This may result in computationally very expensive simulations, due to the potential need for very fine meshes. Third, it is required that the solutions are smooth enough and do not present singularities and/or discontinuities. More precisely, to allow the expansion of the numerical error in powers of the parameter h, the derivatives of the analytical solution should exist and be continuous. Finally, we note that we do not have any guarantee that the Richardson extrapolated solution will meet the same governing equations satisfied by either the numerical solution or the analytical solution; consequently we use this extrapolation for the computation of the numerical error only.

Since it may be demanding to satisfy the requirement of being in the asymptotic regime, Ref. [43] introduces the GCI, defined as

$$GCI = \frac{F_s}{r^{\tilde{p}} - 1} \left| \frac{\bar{X}_{rh} - \bar{X}_h}{\bar{X}_h} \right|, \tag{37}$$

that represents another estimate of the relative discretization error affecting the simulation results. The GCI is obtained by approximating in Eq. (35) $\bar{X}_h r^p - \bar{X}_{rh} \simeq (r^{\tilde{p}} - 1)\, \bar{X}_h$. The

parameters F_s and $\tilde{p}$ ensure that the GCI is larger than the numerical discretization error in 95% of the cases, and are defined as follows: if the difference between p and $\hat{p}$ is less than 10%, the simulations are assumed to be in the asymptotic regime and $F_s = 1.25$ and $\tilde{p} = p$. If the difference between p and $\hat{p}$ is larger than 10%, a more conservative factor of safety, $F_s = 3$, is used and $\tilde{p} = \min[\max(0.5, \hat{p}), p]$. If $\hat{p}$ is not evaluated (for example, if only two solutions are available), $F_s = 3$ and $\tilde{p} = p$. We remark that there is still an ongoing discussion in the verification community about the generality of these estimates. We finally note that the presented procedure can be applied not only to point-by-point solution values, but also to solution functionals.

V. APPLICATION OF THE VERIFICATION METHODOLOGY TO A PIC SIMULATION CODE

In this section we apply the code and solution verification methodologies previously discussed to a simple PIC simulation code. We discuss the chosen manufactured solutions and we verify the correct implementation of the physical model into the simulation code by showing that $\hat{p} \to p$ for $h \to 0$. As an application of the solution verification methodology, we consider the simulation of two counterstreaming beams of electrons, and we discuss how to evaluate the linear growth rate of the resulting two-stream instability and the numerical uncertainty affecting this value.

A. The PIC simulation code

The PIC simulation code considered numerically solves Eqs. (1)-(2) on a periodic spatial domain that extends from $x = 0$ to $x = L$. It uses a numerical grid $x_i = i\Delta x$ to discretize the x coordinate, with $\Delta x = L/M$ the grid spacing ($i = 0, ..., M-1$ and M the number of grid points), and a time step Δt for the integration of the equations of motion. The charge of the particles is assigned to the grid using a first-order weighting scheme, known as cloud-in-cell (CIC) scheme [3], i.e. $\rho(x_i, t) = \sum_{p=1}^{N} qI[x_i - x_p(t)]w_p(t)$, with I the interpolation function given by

$$I(x) = \begin{cases} 0 & \text{if } |x| > \Delta x \\ -\frac{|x|}{\Delta x} + 1 & \text{if } |x| \leq \Delta x. \end{cases} \tag{38}$$

Poisson's equation $\partial_x^2\phi(x,t) = -\rho/\epsilon_0$ is solved by using a second order centered finite difference scheme and imposing the boundary condition $\phi(x=0)=0$. The electric field E_p is computed according to $E(x,t) = -\partial_x\phi(x,t)$ by using a second order centered finite difference scheme and interpolating from the grid onto the particle positions using again the CIC scheme. Finally, the equations of motion, Eq. (3), are integrated in time with a second order Leapfrog scheme. This numerical scheme is second order in Δx and Δt, i.e. $\alpha = \beta = 2$.
In the code, all quantities are normalized to (tilde denotes a physical quantity in SI units): $x = \tilde{x}/\tilde{\lambda}_D$, $t = \tilde{t}\tilde{\omega}_{pe}$, where $\tilde{\lambda}_D = \sqrt{\frac{\epsilon_0 \tilde{T}_{e0}}{\tilde{n}_0 e^2}}$ is the Debye length and $\tilde{\omega}_{pe} = \sqrt{\frac{\tilde{n}_0 e^2}{\epsilon_0 m_e}}$ is the plasma frequency, with $\tilde{n}_0$ and $\tilde{T}_{e0}$ a reference density and electron temperature, respectively. The simulation code is written in Fortran 90 and parallelized using a domain cloning approach, implemented within an hybrid Message Passing Interface (MPI) and OpenMP environment.

B. A practical example of PIC code verification

To apply the code verification methodology previously discussed, we choose the following manufactured solutions

$$E_M(x,t) = 2\pi k_x L \sin(\pi t)\sin\left(k_x \frac{2\pi}{L} x\right) \tag{39}$$

$$f_M(x,v,t) = f_x(x,t) f_v(v), \tag{40}$$

where

$$f_v(v) = \frac{2}{\sqrt{\pi}} v^2 e^{-v^2} \tag{41}$$

and we make use of two functions for $f_x(x,t)$,

$$f_{x1}(x,t) = \left[1 - \sin(\pi t)\cos(2\pi x/L)\right]/L \tag{42}$$

$$f_{x2}(x,t) = \frac{1}{L} + \frac{1}{2}\left[1-\cos(\pi t)\right]\left\{\frac{\lambda\left[(L-x)^{-2} + x^{-2}\right]}{2\cosh^2\left[\lambda(L-x)^{-1} - \lambda x^{-1}\right]} - \frac{1}{L}\right\}, \tag{43}$$

to ensure empirically that the results discussed are valid for different choices of f_M. We denote $f_1 = f_{x1} f_v$ and $f_2 = f_{x2} f_v$. We remark that the manufactured solution satisfies the requirements listed in Sec. III A. In particular, the parameters k_x, L, and λ allow us to calibrate the numerical error, so that the magnitude of the different terms in the Vlasov-Poisson system are of the same order of magnitude (we use $L = 2$ and $k_x = 2$ for f_1, and $k_x = 5$, $L = 5$, and $\lambda = 20$ for f_2, and for all the simulations we evolve the

computational particles for 2 time units). We note that $f_v(v)$ is chosen different from a Maxwellian distribution in v to ensure that a numerical solution does not converge to f_M because of numerical dissipation. Finally, we note that the computational particles are initially distributed according to the probability distribution function $f_0(v) = e^{-|v|}/(2L)$ and the initial weights are computed as $w_p(0) = f_M[x_p(0), v_p(0), 0]/f_0[v_p(0)]$.

For the verification of the PIC code, we refine at the same time the grid size and the time step, while increasing the number of particles. Defining $h = \Delta x/\Delta x_0 = \Delta t/\Delta t_0 = (N/N_0)^{-1/4}$, we perform 5 sets of simulations, with respectively $h = 1, 2, 4, 8, 16$, for both f_1 and f_2. We perform simulations with $h = 8, 16$ thousands of times, simulations with $h = 4$ hundreds of times, and simulations with $h = 1, 2$ a few times, and for each value of h we compute the average of $\epsilon_{FF}\,(f_M)$ and $\epsilon(E_p)$ and the corresponding uncertainty. The observed order of accuracy $\hat{p}$ and the corresponding uncertainty are computed applying Eqs. (8) and (21), respectively. We note that, while $\epsilon(E_p)$ is computed considering all the time steps of the simulations, $\epsilon_{FF}\,(f_M)$ is estimated at $t = 2$ due to the high computational cost of its evaluation.

The results obtained from these simulations are represented in Fig. 1. Both $\epsilon_{FF}\,(f_M)$ and

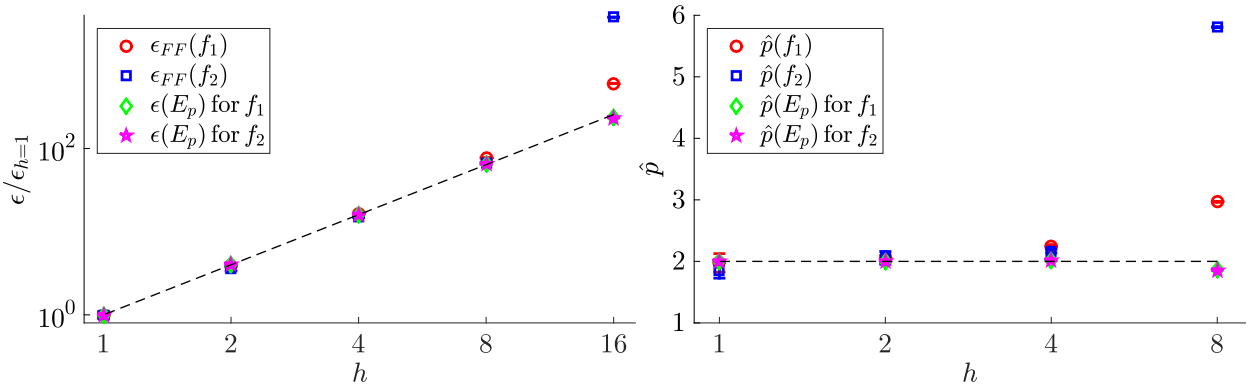

FIG. 1: Values of $\epsilon_{FF}\,(f_M)$ and $\epsilon(E_p)$ averaged over the performed set of simulations (left) and corresponding $\hat{p}$ (right) for the two distribution functions f_1 and f_2, and for $h = 1, 2, 4, 8, 16$. Each error is normalized to its value at $h = 1$, and the statistical uncertainties are represented with errorbars. The dashed lines represent h^2 (left) and $\hat{p} = 2$ (right).

$\epsilon(E_p)$ clearly decrease for $h \to 0$. Moreover, the observed order of accuracy $\hat{p}$ converges to 2 when decreasing h, proving that the PIC algorithm is correctly implemented in the code, and the equations are verified.

As a further proof of the capabilities of the code verification methodology illustrated herein, we perform the same verification with a zero-order weighting scheme (the so-called nearest-grid-point scheme, or NGP) when interpolating the electric field. This corresponds to use an interpolation function defined as

$$I(x) = \begin{cases} 1 & \text{if } |x| \leq \frac{\Delta x}{2} \\ 0 & \text{if } |x| > \frac{\Delta x}{2} \end{cases} \tag{44}$$

when interpolating the electric field from the grid onto the marker positions. Since the accuracy of the numerical scheme is reduced, the error affecting the results is expected to satisfy

$$\epsilon = C'' h + \mathcal{O}(h^2), \tag{45}$$

where C'' is a constant independent of h. The results are presented in Fig. 2 (only f_1 is con-

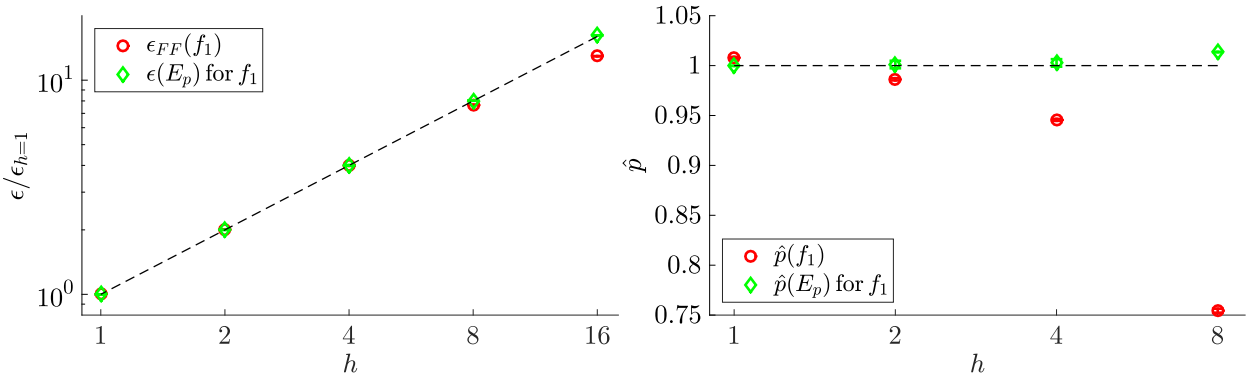

FIG. 2: Values of $\epsilon_{FF}(f_1)$ and $\epsilon(E_p)$ averaged over the performed set of simulations (left) and corresponding $\hat{p}$ (right) for $h = 1, 2, 4, 8, 16$. Each error is normalized to its value at $h = 1$, and the statistical uncertainties are represented with errorbars. The dashed lines represent h (left) and $\hat{p} = 1$ (right).

sidered for this test). The code verification methodology is able to identify this change in the numerical scheme. In fact, while both $\epsilon_{FF}(f_1)$ and $\epsilon(E_p)$ decreases as $h \to 0$, the observed order of accuracy converges to 1. Therefore, the proposed code verification methodology not only ensures that the numerical solution converges to the exact solution, but it also correctly identifies the convergence rate.

To investigate the applicability of the distance $\epsilon_{MC}(f_M)$ for the verification of PIC simulation codes, we consider the same set of simulations presented in Figs. 1 and 2, and we

evaluate the difference between $f_M(x, v, t)$ and the sample of computational particles according to Eq. (25) at $t = 2$, for $M = 10^6$. The results thus obtained are shown in Fig. 3. We

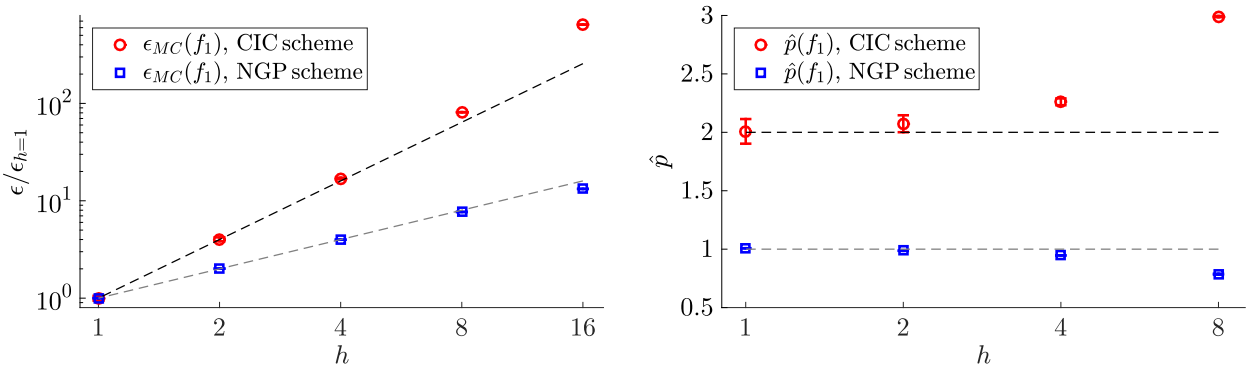

FIG. 3: Values of $\epsilon_{MC}(f_1)$ averaged over the performed set of simulations (left) and corresponding $\hat{p}$ (right) for $h = 1, 2, 4, 8, 16$, interpolating the electric field onto the marker positions both with the CIC (red) and the NGP (blue) interpolation schemes. Each error is normalized to its value at $h = 1$, and the statistical uncertainties are represented with errorbars. The dashed lines represent h and h^2 (left) and $\hat{p} = 1$ and $\hat{p} = 2$ (right).

observe that the error decreases for $h \to 0$ as expected both for the CIC and NGP schemes, with $\hat{p} \to 2$ for the CIC scheme, and $\hat{p} \to 1$ for the NGP scheme. Moreover, we note that the errors computed according to the Fasano and Franceschini method (see Figs. 1 and 2) and according to Eq. (25) are very similar. This means that the norm ϵ_{MC} is suitable for the verification of PIC simulation codes.

Finally, we consider the same set of simulations presented in Figs. 1 and 2 and we evaluate $\epsilon_x(f_M)$ and $\epsilon_v(f_M)$. The results thus obtained are presented in Figs. 4 and 5 for the CIC and NGP weighting schemes, respectively. We observe that the error decreases for $h \to 0$ as expected for both the weighting schemes, with $\hat{p} \to 2$ for the CIC scheme, and $\hat{p} \to 1$ for the NGP scheme. Therefore, also the norms defined in Eqs. (26)-(27) are suitable for the verification of PIC simulation codes with the MMS. We note that, as the computational cost of evaluating $\epsilon_x(f_M)$ and $\epsilon_v(f_M)$ is considerably decreased with respect to $\epsilon_{FF}(f_M)$, this evaluation is performed for all $t = 0.08j$, with $j = 0, ..., 25$, and the maximum between the resulting values is computed. We finally remark that this last approach is easily generalized to a d-dimensional distribution function, without increasing significantly the computational cost of performing the order-of-accuracy test.

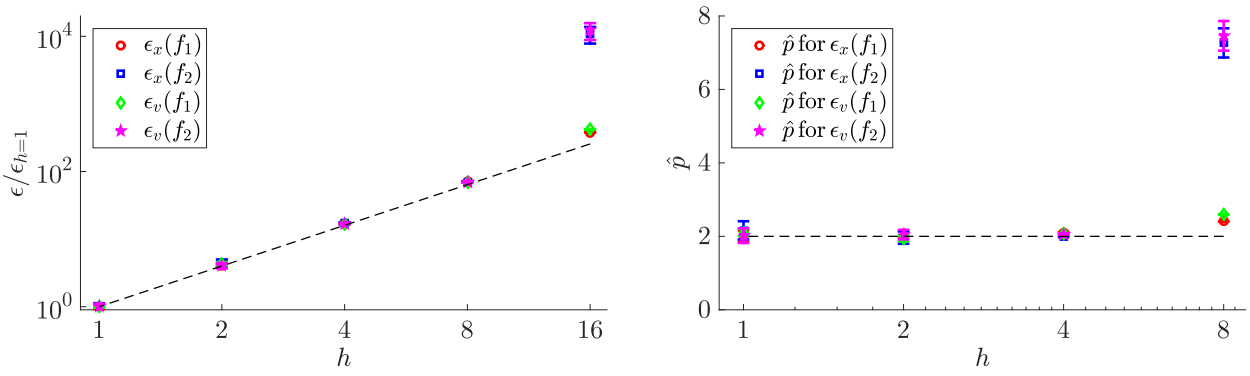

FIG. 4: Values of $\epsilon_x(f_M)$ and $\epsilon_v(f_M)$ averaged over the performed set of simulations (left) and corresponding $\hat{p}$ (right) for the two distribution functions f_1 and f_2, and for $h = 1, 2, 4, 8, 16$. Each error is normalized to its value at $h = 1$, and the statistical uncertainties are represented with errorbars. The dashed lines represent h^2 (left) and $\hat{p} = 2$ (right). The electric field is interpolated from the grid onto the particle positions using the CIC scheme.

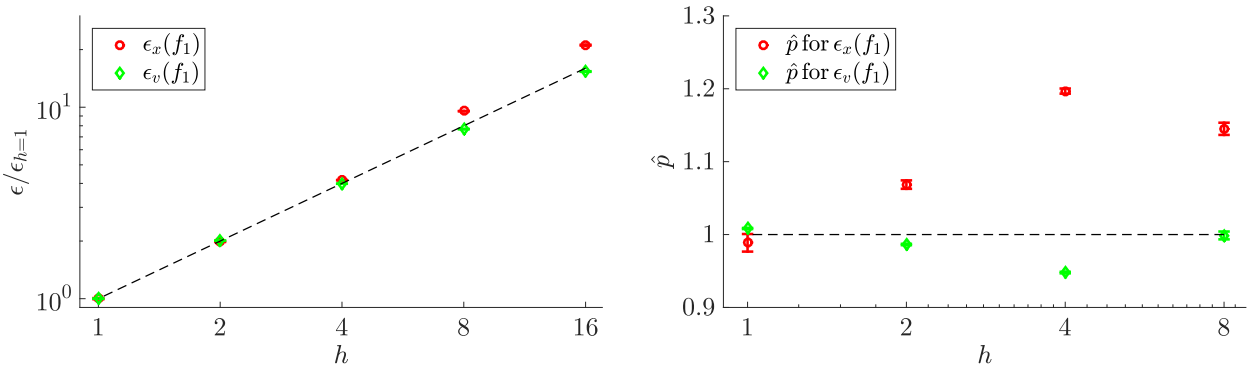

FIG. 5: Values of $\epsilon_x(f_M)$ and $\epsilon_v(f_M)$ averaged over the performed set of simulations (left) and corresponding $\hat{p}$ (right) for the two distribution functions f_1 and f_2 and for $h = 1, 2, 4, 8, 16$. Each error is normalized to its value at $h = 1$, and the statistical uncertainties are represented with errorbars. The dashed lines represent h (left) and $\hat{p} = 1$ (right). The electric field is interpolated from the grid onto the particle positions using the NGP scheme.

C. A practical example of PIC solution verification

In order to illustrate a practical example of application of the solution verification methodology, we consider here the two-stream instability. This textbook plasma instability is ideally studied by using PIC simulations.

Consider the distribution function $f(x, v) = f_x(x)f_v(v)$, where $f_x(x) = 1/L$ and $f_v(v) =$

$[\delta(v-v_0)+\delta(v+v_0)]/2$, with $\delta(v)$ the Dirac function. The dispersion relation associated with small amplitude perturbations is

$$D(\omega,k)=1-\frac{1}{2}\left[\frac{1}{(\omega-kv_0)^2}+\frac{1}{(\omega+kv_0)^2}\right]. \tag{46}$$

Since for $0<k<1/v_0$ the ω solution of $D(\omega,k)=0$ is complex, the system is affected by an instability called two-stream instability. As a consequence, if the system is perturbed, small amplitude modes can grow exponentially, before saturating due to nonlinear effects. The fastest growing mode, with growth rate $\gamma_{\max}=1/\sqrt{8}$, is obtained for $k_{\max}=\sqrt{3/(8v_0^2)}$. To numerically compute the linear growth rate of the two-stream instability, we proceed as follows. First, we initialize our PIC simulations according to a distribution function $f=[f_x+A\cos(k_{\max}x)]\,f_v$, where $A\ll 1$ is used to seed the perturbation. Second, we compute the Fourier transform of $\phi(x,t)$ along x, thus obtaining $\tilde{\phi}(k,t)$. Third, we identify the time interval during which the mode $\tilde{\phi}(k_{\max},t)$ grows exponentially. Finally, over the identified time interval, we fit the amplitude of the mode, $|\tilde{\phi}(k_{\max},t)|$, with an exponential curve to evaluate $\gamma_{\max}$.

We now apply the solution verification methodology discussed in Sec. IV to rigorously estimate $\gamma_{\max}$ and its numerical uncertainty $\Delta\gamma_{\max}$. We perform three sets of ten simulations for $h=1,2,4$, with different pseudorandom number generator seed, with $L=2\pi$, $v_0=0.2$, $\Delta x_0=L/128$, $\Delta t_0=1/16$, and $N_0=2.048\cdot 10^9$, for which we expect $k_{\max}\simeq 3$ and $\gamma_{\max}\simeq 0.353$. The time evolution of $|\tilde{\phi}(k_{\max},t)|$ is shown in Fig. 6 (left panel) for $h=4$. After an initial transient, the mode grows until $t\simeq 18$, before saturating because of non-

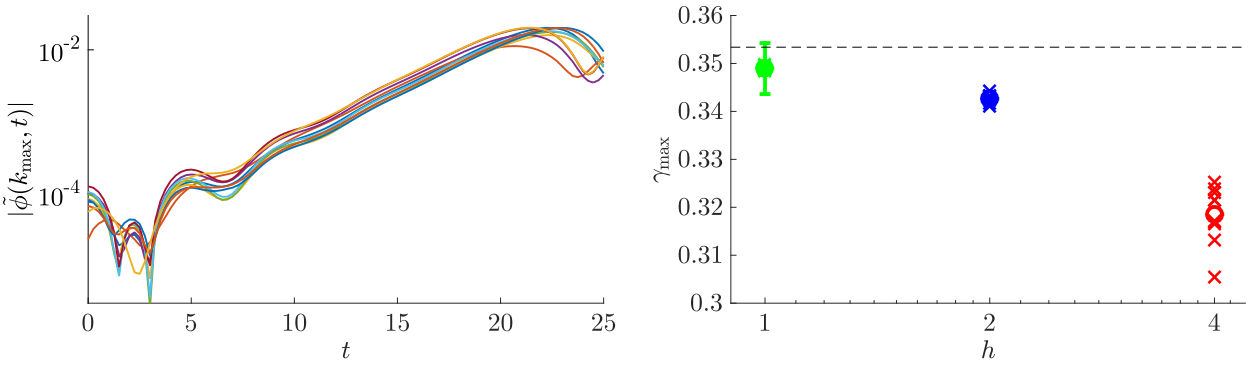

FIG. 6: Time evolution of $|\tilde{\phi}(k_{\max},t)|$ for $h=4$ (left) and growth rates of the two-stream instability for $h=1,2,4$ (right). The circles in the right panel represent $\bar{\gamma}_{\max,h}$, while the errorbars represent $\Delta\gamma_{\max}$. The dashed line represents the expected value of $\gamma_{\max}$

linear effects. We exponentially fit each profile in the time interval $15 \leq t \leq 17$ to obtain the growth rates $\gamma_{\max}$ that we plot in Fig. 6 (right panel, red crosses). The same process is repeated for $h = 1$ (green crosses) and $h = 2$ (blue crosses). It is noticeable that the spreading of the growth rates is smaller at smaller h (i.e. larger N).

To compute $\Delta\gamma_{\max}$ we estimate separately the uncertainty introduced by the post-processing (i.e., the exponential fit), $\Delta\gamma^{fit}_{\max}$, the statistical uncertainty, $\Delta\gamma^{stat}_{\max}$, and the discretization error, $\Delta\gamma^{disc}_{\max}$. First, the uncertainty introduced by the exponential fit is $\Delta\gamma^{fit}_{\max,1} \simeq 0.003$, that is the confidence interval of the fit for the simulations with $h = 1$. Second, applying Eqs. (28), we compute the values of $\bar{\gamma}_{\max,h}$ averaged over the set of simulations for $h = 1, 2, 4$, which are represented in Fig. 6 (right panel) as circles. We note that, as discussed in Sec. IV A, the variance of $\gamma_{\max,1}$ can be estimated as $\sigma_{\gamma,1} \approx \sigma_{\gamma,4}/16 = 3.7 \cdot 10^{-4}$. This is close to $\sigma_{\gamma,1} = 3.9 \cdot 10^{-4}$, obtained with Eq. (31). We therefore obtain $\Delta\gamma^{stat}_{\max,1} = 0.0002$.

Third, the discretization error is obtained by applying the methodology discussed in Sec. IV B. In particular, using the three estimates of $\gamma_{\max,h}$ obtained by averaging over the 10 simulations, which are $\bar{\gamma}_{\max,h} = 0.349, 0.343, 0.318$ for $h = 1, 2, 4$, respectively, we compute the Richardson extrapolation $\bar{\gamma}_{\max} = \bar{\gamma}_{\max,1} + (\bar{\gamma}_{\max,1} - \bar{\gamma}_{\max,2})/3 = 0.351$ according to Eq. (33). We also compute the observed order of accuracy according to Eq. (36), obtaining $\hat{p} = 1.96$, thus ensuring that the Richardson extrapolation is a reasonable estimate of the exact solution. The discretization error is thus computed according to Eq. (34), obtaining $\Delta\gamma^{disc}_{\max,1} = 0.002$.

Finally, $\Delta\gamma_{\max}$ is obtained by summing up the uncertainty introduced by the exponential fit, the statistical uncertainty, and the discretization error, $\Delta\gamma_{\max} = \Delta\gamma^{fit}_{\max,1} + \Delta\gamma^{stat}_{\max,1} + \Delta\gamma^{disc}_{\max,1}$, resulting $\Delta\gamma_{\max} \simeq 0.005$. Comparing the value of $\bar{\gamma}_{\max,1} = 0.349$ with the expected value $\gamma_{\max} \simeq 0.353$, it results that the numerical evaluation of $\gamma_{\max}$ is consistent with the exact solution within the numerical uncertainty.

VI. CONCLUSIONS

In the present paper a methodology to rigorously verify PIC simulations is proposed, generalizing the procedures for finite difference codes presented in Ref. [34] to PIC algorithms. The main differences between the verification of grid-based and PIC simulation codes are discussed, and a methodology to overcome the emerging difficulties is illustrated.

To rigorously assess the correct implementation of PIC algorithms into simulation codes, an order-of-accuracy test based on the MMS is proposed, accounting for numerical schemes intrinsically affected by statistical noise, and providing a measure of the distance between continuous, analytical distribution functions, and finite samples of computational particles. In particular, the value of ϵ_h is estimated averaging over several simulations carried out with different pseudorandom number generator seeds, and the statistical uncertainty affecting ϵ_h and $\hat{p}$ is quantified. Then, the distances defined in Refs. [50, 52] are generalized to account for time-evolving marker weights, proving empirically that ϵ_P and ϵ_{FF} still decrease as $N^{-1/2}$ for $N \to \infty$ when $w_p \neq 1$. Moreover, since the proposed norms are extremely demanding in terms of computational resources when large number of computational particles are considered, the value of d_i^P is approximated with a Monte-Carlo approach and ϵ_{MC} is used in verifying the PIC simulation code, allowing to considerably decrease the computational cost of a PIC code verification. Finally, the norms ϵ_x and ϵ_v are introduced, showing that it is possible to consider independently each coordinate of the phase-space when performing a PIC code verification. The latter approach is easily generalized to phase-space in more dimensions, without increasing the computational cost considerably.

To estimate the numerical uncertainty affecting the simulation results, a rigorous methodology is proposed. In particular, the uncertainty introduced by using a finite grid, a finite time step, and a finite number of markers to perform a PIC simulation is quantified by introducing the Richardson extrapolation to approximate the exact solution of the model, and thus estimating the discretization error affecting the simulation results. Moreover, the statistical uncertainty is quantified by repeating the simulation with different pseudorandom number generator seeds. Finally, the numerical uncertainty affecting the simulation results is computed by summing up the different contributions.

The application of the proposed procedures to a one-dimensional, electrostatic, collisionless PIC simulation code allowed us to investigate the peculiarities of the verification methodology, showing how to perform a rigorous PIC code verification. We also quantify the numerical uncertainty affecting the estimate of the two-stream instability growth rate. The verification methodology discussed in this paper can be easily generalized to more complex geometries and more realistic systems, providing the basis to perform a rigorous verification of complex PIC simulations.

Acknowledgments

The authors gratefully acknowledge useful discussions with S. Brunner and L. Villard. The simulations presented herein were carried out in part using the HELIOS supercomputer system at Computational Simulation Centre of International Fusion Energy Research Center (IFERC-CSC), Aomori, Japan, under the Broader Approach collaboration between Euratom and Japan, implemented by Fusion for Energy and JAEA; and in part at the Swiss National Supercomputing Center (CSCS) under Projects ID s549. This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Fond National Suisse de la Recherche scientifique and from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

APPENDIX A: VERIFICATION OF DIFFERENT APPROACHES TO EVALUATING THE DIFFERENCE BETWEEN AN ANALYTICAL DISTRIBUTION FUNCTION AND A FINITE SAMPLE OF PARTICLES

In this appendix we empirically show that $\epsilon_P(f_M)$, $\epsilon_{FF}(f_M)$, and $\epsilon_{MC}(f_M)$ decrease as $N^{-1/2}$ for $N \to \infty$ if $\{x_p, v_p\}_{p=1,...,N}$ is a set of random realizations of f_M, even if $w_p \neq 1$. We first generate N points (x_p, v_p) according to $f_0(v) = e^{-|v|}/(2L)$, with $p = 1, ..., N$, and we set $w_p = f_1(x_p, v_p, 0)/f_0(v_p)$. Then, we compute $\epsilon_P(f_1)$, $\epsilon_{FF}(f_1)$, and $\epsilon_{MC}(f_1)$ at $t = 0$ [for $\epsilon_{MC}(f_1)$, $M = 10^6$]. We apply this procedure for different N, and, for each N, we repeat the process a number of times, changing the pseudorandom number generator seed. We compute the averaged value of $\epsilon_P(f_1)$, $\epsilon_{FF}(f_1)$, and $\epsilon_{MC}(f_1)$ according to Eq. (28), and the corresponding statistical uncertainties according to Eq. (30). Finally, defining $h = 1/N$, we estimate $\hat{p}$ and its statistical uncertainty by applying Eqs. (8) and (21). The results thus obtained are presented in Fig 7. We observe that the distance between $f_1(x, v, 0)$ and the data sets decreases as $N^{-1/2}$ for $N \to \infty$, with a similar value, for all the three norms.

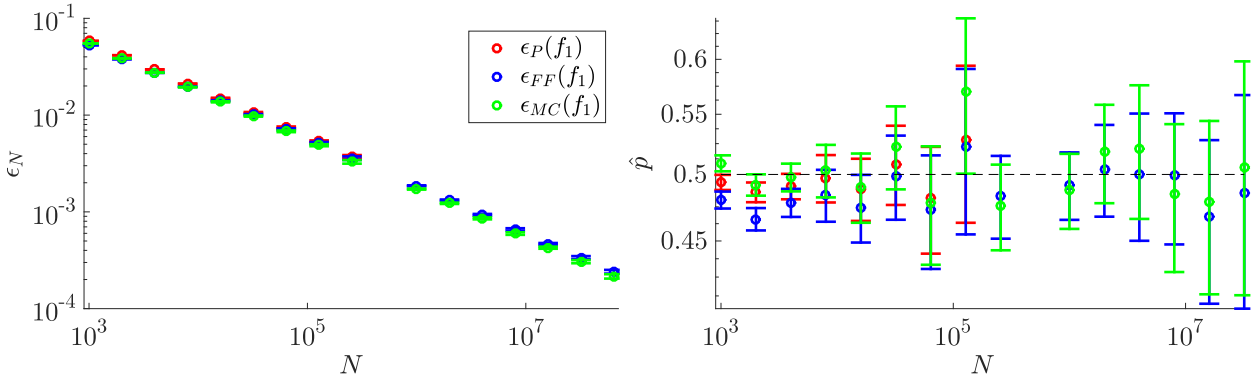

FIG. 7: Values of $\epsilon_P(f_1)$, $\epsilon_{FF}(f_1)$, and $\epsilon_{MC}(f_1)$ averaged over the performed set of simulations (left) and corresponding values of $\hat{p}$ (right) for $h = 1/N$. The errorbars represent the statistical uncertainty affecting the results.

---

[1] F.H. Harlow. The particle-in-cell computing method for fluid dynamics. *Methods for Comput. Phys.*, 3:319–343, 1964.

[2] John Dawson. One-Dimensional Plasma Model. *Physics of Fluids*, 5(4):445, 1962.

[3] Charles K. Birdsall and A. Bruce Langdon. *Plasma physics via computer simulation.* Series in plasma physics. Taylor & Francis, New York, 2005. Originally published: New York ; London : McGraw-Hill, 1985.

[4] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles.* Taylor & Francis, Inc., Bristol, PA, USA, 1988.

[5] Y.N. Grigoryev, V.A. Vshivkov, and M.P. Fedoruk. *Numerical "Particle-in-Cell" Methods: Theory and Applications.* De Gruyter, 2002.

[6] John M. Dawson. Computer modeling of plasma: Past, present, and future. *Physics of Plasmas*, 2(6):2189, 1995.

[7] John Villasenor and Oscar Buneman. Rigorous charge conservation for local electromagnetic field solvers. *Computer Physics Communications*, 69(2-3):306–316, mar 1992.

[8] G. Chen, L. Chacón, and D.C. Barnes. An energy- and charge-conserving, implicit, electrostatic particle-in-cell algorithm. *Journal of Computational Physics*, 230(18):7018–7036, aug 2011.

[9] Giovanni Lapenta and Stefano Markidis. Particle acceleration and energy conservation in particle in cell simulations. *Physics of Plasmas*, 18(7):072101, 2011.

[10] H. Denavit. Time-Filtering Particle Simulations. *Jouranl of Computational Physics*, 366:337–366, 1981.

[11] Rodney J. Mason. Implicit moment particle simulation of plasmas. *Journal of Computational Physics*, 41(2):233–244, jun 1981.

[12] J.U Brackbill and D.W Forslund. An implicit method for electromagnetic plasma simulation in two dimensions. *Journal of Computational Physics*, 46(2):271–308, may 1982.

[13] A.Bruce Langdon, Bruce I Cohen, and Alex Friedman. Direct implicit large time-step particle simulation of plasmas. *Journal of Computational Physics*, 51(1):107–138, jul 1983.

[14] Giovanni Lapenta, J. U. Brackbill, and Paolo Ricci. Kinetic approach to microscopic-macroscopic coupling in space and laboratory plasmas. *Physics of Plasmas*, 13(5):055904, 2006.

[15] D. W. Forslund and C. R. Shonk. Formation and Structure of Electrostatic Collisionless Shocks. *Physical Review Letters*, 25(25):1699–1702, dec 1970.

[16] D. W. Forslund, K. B. Quest, J. U. Brackbill, and K. Lee. Collisionless dissipation in quasi-perpendicular shocks. *Journal of Geophysical Research*, 89(A4):2142, 1984.

[17] B Lembege and Dawson J M. Formation of Double Layers Within an Oblique Collisioinless Shock. *Physical Review Letters*, 62(23):2683–2686, 1989.

[18] P. L. Pritchett. Geospace Environment Modeling magnetic reconnection challenge: Simulations with a full particle electromagnetic code. *Journal of Geophysical Research: Space Physics*, 106(A3):3783–3798, mar 2001.

[19] J F Drake, M Swisdak, C Cattell, M A Shay, B N Rogers, and A Zeiler. Formation of electron holes and particle energization during magnetic reconnection. *Science (New York, N.Y.)*, 299(5608):873–7, feb 2003.

[20] Paolo Ricci, J. U. Brackbill, W. Daughton, and Giovanni Lapenta. Collisionless magnetic reconnection in the presence of a guide field. *Physics of Plasmas*, 11(8):4102, 2004.

[21] C. Joshi, W. B. Mori, T. Katsouleas, J. M. Dawson, J. M. Kindel, and D. W. Forslund. Ultrahigh gradient particle acceleration by intense laser-driven plasma density waves. *Nature*, 311(5986):525–529, oct 1984.

[22] S P D Mangles, C D Murphy, Z Najmudin, A G R Thomas, J L Collier, A E Dangor, E J

Divall, P S Foster, J G Gallacher, C J Hooker, D A Jaroszynski, A J Langley, W B Mori, P A Norreys, F S Tsung, R Viskup, B R Walton, and K Krushelnick. Monoenergetic beams of relativistic electrons from intense laser–plasma interactions. *Nature*, 431(7008):535–538, sep 2004.

[23] S. C. Wilks, W. L. Kruer, M. Tabak, and A. B. Langdon. Absorption of ultra-intense laser pulses. *Physical Review Letters*, 69(9):1383–1386, aug 1992.

[24] D. Tskhakaya and S. Kuhn. Particle-in-cell simulations of the plasma-wall transition with a magnetic field almost parallel to the wall. *Journal of Nuclear Materials*, 313-316(SUPPL.):1119–1122, 2003.

[25] Joaquim Loizu, Paolo Ricci, and Christian Theiler. Existence of subsonic plasma sheaths. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 83(1):016406, jan 2011.

[26] P. W. Terry, M. Greenwald, J.-N. Leboeuf, G. R. McKee, D. R. Mikkelsen, W. M. Nevins, D. E. Newman, and D. P. Stotler. Validation in fusion research: Towards guidelines and best practices. *Physics of Plasmas*, 15(6):062503, 2008.

[27] Martin Greenwald. Verification and validation for magnetic fusion. *Physics of Plasmas*, 17(5):058101, 2010.

[28] A. M. Dimits, G. Bateman, M. A. Beer, B. I. Cohen, W. Dorland, G. W. Hammett, C. Kim, J. E. Kinsey, M. Kotschenreuther, A. H. Kritz, L. L. Lao, J. Mandrekas, W. M. Nevins, S. E. Parker, A. J. Redd, D. E. Shumaker, R. Sydora, and J. Weiland. Comparisons and physics basis of tokamak transport models and turbulence simulations. *Physics of Plasmas*, 7(3):969, 2000.

[29] G L Falchetto, B D Scott, P Angelino, A Bottino, T Dannert, V Grandgirard, S Janhunen, F Jenko, S Jolliet, A Kendl, B F McMillan, V Naulin, A H Nielsen, M Ottaviani, A G Peeters, M J Pueschel, D Reiser, T T Ribeiro, and M Romanelli. The European turbulence code benchmarking effort: turbulence driven by thermal gradients in magnetically confined plasmas. *Plasma Physics and Controlled Fusion*, 50(12):124015, dec 2008.

[30] M. M. Turner, A. Derzsi, Z. Donko, D. Eremin, S. J. Kelly, T. Lafleur, and T. Mussenbrock. Simulation benchmarks for low-pressure plasmas: Capacitive discharges. *Physics of Plasmas*, 20(1):013507, 2013.

[31] R. V. Bravenec, Y. Chen, J. Candy, W. Wan, and S. Parker. A verification of the gyrokinetic microstability codes GEM, GYRO, and GS2. *Physics of Plasmas*, 20(10):104506, 2013.

[32] J.U. Brackbill. On energy and momentum conservation in particle-in-cell plasma simulation. *Journal of Computational Physics*, 317:405–427, jul 2016.

[33] William L. Oberkampf and Christopher J. Roy. *Verification and Validation in Scientific Computing*. Cambridge University Press, New York, NY, USA, 2010.

[34] F. Riva, P. Ricci, F. D. Halpern, S. Jolliet, J. Loizu, and A. Mosetto. Verification methodology for plasma simulations and application to a scrape-off layer turbulence code. *Physics of Plasmas*, 21(6):062301, jun 2014.

[35] Stanly Steinberg and Patrick J Roache. Symbolic manipulation and computational fluid dynamics. *Journal of Computational Physics*, 57(2):251–284, jan 1985.

[36] Patrick J. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Publishers, Albuquerque, NM, USA, 1998.

[37] Patrick J. Roache. Code Verification by the Method of Manufactured Solutions. *Journal of Fluids Engineering*, 124(1):4, 2002.

[38] Christopher J. Roy. Review of code and solution verification procedures for computational simulation. *Journal of Computational Physics*, 205(1):131–156, may 2005.

[39] B. D. Dudson, J. Madsen, J. Omotani, P. Hill, L. Easy, and M. Løiten. Verification of BOUT++ by the method of manufactured solutions. *Physics of Plasmas*, 23(6):062303, jun 2016.

[40] P. Tamain, H. Bufferand, G. Ciraolo, C. Colin, D. Galassi, Ph Ghendrih, F. Schwander, and E. Serre. The TOKAM3X code for edge turbulence fluid simulations of tokamak plasmas in versatile magnetic geometries. *Journal of Computational Physics*, 321:606–623, sep 2016.

[41] L. F. Richardson. The Approximate Arithmetical Solution by Finite Differences of Physical Problems Involving Differential Equations, with an Application to the Stresses in a Masonry Dam. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 210(459-470):307–357, jan 1911.

[42] L. F. Richardson and J. A. Gaunt. The Deferred Approach to the Limit. Part I. Single Lattice. Part II. Interpenetrating Lattices. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 226(636-646):299–361, jan 1927.

[43] Patrick J Roache. Perspective: A Method for Uniform Reporting of Grid Refinement Studies. *Journal of Fluids Engineering*, 116(3):405, 1994.

[44] A. Y. Aydemir. A unified Monte Carlo interpretation of particle simulations and applications

to non-neutral plasmas. *Physics of Plasmas*, 1(4):822, 1994.

[45] S.J. Allfrey and R. Hatzky. A revised δf algorithm for nonlinear PIC simulation. *Computer Physics Communications*, 154(2):98–104, aug 2003.

[46] Genze Hu and John A Krommes. Generalized weighting scheme for δf particle-simulation method. *Physics of Plasmas*, 1(4):863, 1994.

[47] F. James. *Statistical Methods in Experimental Physics*. World Scientific Publishing Co Inc, 2006.

[48] L. Lista. *Statistical Methods for Data Analysis in Particle Physics*. Lecture Notes in Physics. Springer International Publishing, 2015.

[49] A. N. Kolmogorov. Sulla Determinazione Empirica di una Legge di Distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, 4:83–91, 1933.

[50] J. A. Peacock. Two-dimensional goodness-of-fit testing in astronomy. *Monthly Notices of the Royal Astronomical Society*, 202(3):615–627, mar 1983.

[51] R. H. C. Lopes, I. Reid, and P. R. Hobson. The two-dimensional Kolmogorov-Smirnov test. In *XI International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, 2007.

[52] G. Fasano and A. Franceschini. A multidimensional version of the Kolmogorov-Smirnov test. *Monthly Notices of the Royal Astronomical Society*, 225(1):155–170, mar 1987.