Verification methodology for plasma simulations and application to a scrape-off layer turbulence code

Fabio Riva, Paolo Ricci, Federico D. Halpern, Sébastien Jolliet, Joaquim Loizu, and Annamaria Mosetto

Citation: Physics of Plasmas (1994-present) **21**, 062301 (2014); doi: 10.1063/1.4879778
View online: http://dx.doi.org/10.1063/1.4879778
View Table of Contents: http://scitation.aip.org/content/aip/journal/pop/21/6?ver=pdfcov
Published by the AIP Publishing

Verification methodology for plasma simulations and application to a scrape-off layer turbulence code

Fabio Riva,[a] Paolo Ricci, Federico D. Halpern, Sébastien Jolliet, Joaquim Loizu, and Annamaria Mosetto
École Polytechnique Fédérale de Lausanne (EPFL), Centre de Recherches en Physique des Plasmas (CRPP), CH-1015 Lausanne, Switzerland

(Received 31 January 2014; accepted 5 May 2014; published online 2 June 2014)

Bridging the gap between plasma physics and other scientific domains, in particular, the computational fluid dynamics community, a general, rigorous, and simple-to-apply methodology is presented for both the verification of the correct implementation of the model equations (code verification) and numerical error quantification (solution verification). The proposed code verification procedure consists in using the method of manufactured solutions and executing an order-of-accuracy test, assessing the rate of convergence of the numerical solution to the manufactured one. For the solution verification, the numerical error is quantified by applying the Richardson extrapolation, which provides an approximation of the analytical solution, and by using the grid convergence index to estimate the numerical uncertainty affecting the simulation results. The methodology is applied to verify the correct implementation of the drift-reduced Braginskii equations into the GBS code, and to estimate the numerical error affecting the GBS solutions. The GBS code is successfully verified, and an estimate of the numerical error affecting the simulation results is provided. [http://dx.doi.org/10.1063/1.4879778]

I. INTRODUCTION

In plasma physics and in related domains (e.g., computational fluid dynamics), the methodology used to assess the reliability of numerical simulation codes constitutes the Verification and Validation (V&V) procedure.[1–3] V&V is composed by two separated tasks: The verification process, which is a mathematical issue targeted to assess that the physical model is correctly solved, and the validation, used to assess the consistency of the code results, and therefore, of the physical model, with experimental data. Verification can moreover be separated into two different procedures:[2,3] First, the assessment of the correct implementation of the model equations in the code (also known as code verification); second, the estimate of the numerical error affecting the simulation results (this is typically referred to as solution verification).

While in plasma physics a rigorous methodology for code validation has been recently proposed[4,5] and has been applied to the analysis of the experimental data for the TORPEX experiment[6,7] and other experimental devices (see, e.g., Refs. 8 and 9), the absence of a systematic and rigorous approach to code verification persists and motivates the work presented in the present paper. As a matter of fact, there is a strong motivation to increase the reliability of the results of numerical simulations in our domain. Approaching the ITER[10] era, errors, affecting simulations that are used as fundamental tools to uncover the complex plasma dynamics in a tokamak, due both to mistakes present in the code and the implementation of a nonsufficiently accurate physical model, can have far reaching consequences on costly nuclear facilities.

To perform the code verification process, five different approaches have been developed and used, particularly, in the computational fluid dynamics community, here listed from the least to the most rigorous procedure:[2] (a) Simple tests, (b) code-to-code comparison (also known as code-to-code benchmark), (c) quantification of the discretization error with respect to a known solution, (d) convergence tests to a known solution, and (e) order-of-accuracy convergence tests. The first two procedures [(a) and (b)] are the simplest to perform, because they do not need any known analytical solution of the model equations. In the simple test category fall, for example, the symmetry tests, conservation tests, Galilean invariance tests, convergence tests to a solution obtained on a refined numerical grid. Code-to-code benchmark is referred to the comparison between the numerical results produced by a code used as reference and the ones obtained with the code to be verified (examples of application of this method are given in Refs. 11–15). Only the three other approaches [(c)–(e)] are rigorous, but they require an analytical solution of the model equations. Knowing the analytical solution, it is possible to compute the numerical error, which we can consequently quantify (discretization error quantification). We can also verify the convergence of the numerical solution to the analytical one (convergence tests), and we can assess its convergence rate (order-of-accuracy tests). A more detailed description of these five procedures is given in Ref. 2.

Regarding the solution verification procedure, we notice that there are four sources of numerical errors in plasma simulations: (a) Round-off errors (i.e., the errors due to the finite computational precision of computers), (b) statistical sampling errors (e.g., errors due to the evaluation of time-averaged quantities used for the validation of simulation

[a]Electronic mail: fabio.riva@epfl.ch

results), (c) iterative errors (e.g., iterative methods to solve linear systems of equations), and (d) discretization errors (i.e., the errors due to the finite grid spacing used in the numerical scheme).[2,3] All these have to be estimated in order to provide the uncertainty affecting the simulation results; this is necessary to perform a rigorous validation of the code results and to assess the reliability of the code predictions.

The most common approach used in plasma physics for code verification is a comparison between results of different simulation codes (code-to-code benchmark). While valuable, this test is not rigorous enough to ensure the correctness of the considered codes. In fact, a fully verified code of reference implementing the same mathematical model is needed to use this method,[1,16] and, generally, it is very difficult to understand if a difference in the code results is due to discretization errors or to a non-correct implementation of the model. Moreover, performing a benchmark between two simulation codes can be tedious due to different choices in normalization, coordinates, etc. We remark that more rigorous code verification procedures have been used in plasma physics;[17,18] however, their use remained limited to single routines, without approaching the full complexity of a simulation code. Concerning the numerical error affecting the simulations results, in the plasma physics domain, this is usually quantified by performing grid-refinement-based analysis, a systematic and rigorous methodology not being widely used yet. On the other hand, in some other scientific domains, in particular, within the computational fluid dynamics community, a complete and exhaustive study of the verification methodology has been done and a systematic methodology has been developed.

The goal of the present paper is twofold. First, we present the code and solution verification methodology, developed in particular by the computational fluid dynamics community, bridging the gap between our community and other scientific domains, where considerable experience was developed on the subject in the last years. Second, we show for the first time a complete and rigorous application of this verification procedure to a plasma turbulence code, namely the GBS code,[19] used to simulate plasma turbulence in the tokamak scrape-off layer (SOL) and in basic plasma devices.

For the code verification procedure, we focus on the order-of-accuracy tests, as those are the only ones able to ensure both the correct coding of the model equations and the correct implementation of the chosen numerical scheme.[2] Since an analytical solution is not available for most of the physical models used in plasma physics research, a systematic approach can be employed to overcome this issue, that is the method of manufactured solutions[3,20–22] (MMS). This approach has been developed by the computational fluid dynamics community, with the idea of reversing the considered problem: instead of searching the analytical solution of the problem, we impose a manufactured solution, and we modify the model equations by adding analytical terms with the goal of accommodating the manufactured solution. Concerning the numerical error estimate, we focus on grid-based approaches used to quantify the discretization error affecting the simulations, in particular, we illustrate the use of the Richardson extrapolation[23,24] as higher-order estimator of the analytical solution, and we introduce Roache's grid convergence index[25] (GCI) as a relative numerical uncertainty estimate. The proposed methodology is successfully applied to the GBS code, giving a rigorous verification of the code and an estimate of the numerical error affecting the simulation results.

This paper is structured as follows. After the Introduction, in Sec. II, we propose a systematic and rigorous approach, developed mainly by the computational fluid dynamics community, for code verification using the MMS (Sec. II A), and we discuss the solution verification methodology (Sec. II B), by illustrating the Richardson extrapolation method and the Roache's GCI. In Sec. III, we apply the proposed verification approach to the GBS code. The conclusion follows in Sec. IV.

II. VERIFICATION METHODOLOGY

The code and solution verification methodology are presented here separately. First, to rigorously verify a simulation code, we propose to perform an order-of-accuracy test using the MMS approach. Second, we illustrate the procedure to estimate the uncertainty affecting the simulation results considering the use of the Richardson extrapolation to approximate the analytical solution and the implementation of the GCI to quantify the numerical error affecting the simulations.

A. Code verification methodology

The order-of-accuracy test is the only one that can ensure the correct implementation of the physical model and of the numerical scheme.[2] This test analyzes the convergence of the numerical solution to a known analytical solution, also verifying that the discretization errors reduce at the rate expected for the numerical scheme, as the spatial mesh and the time step are refined.

Given a theoretical model M with an analytical solution s, such that $M(s)=0$, and the numerically discretized model of M, M_h, with a numerical solution s_h that satisfies $M_h(s_h)=0$ (h is a parameter representing the degree of refinement of the mesh), the error affecting the numerical results is expressed as $\epsilon_h = ||s_h - s||$, where $||\cdot||$ denotes a designed norm. The theoretical order of accuracy, p, associated with the numerically discretized operator M_h, represents the rate at which the numerical solution converges to the analytical solution as the mesh is refined. The numerical error, in fact, satisfies the relation $\epsilon_h = C_p h^p + O(h^{p+1})$, where C_p is independent of h, and p is the order of accuracy of the numerical scheme, typically evaluated through its Taylor expansion.[2,3,22] Having the two numerical solutions of M_h and M_{rh}, i.e., s_h and s_{rh}, where rh indicates coarsening the h mesh by a factor r, one can evaluate an observed order of accuracy, $\hat{p}$, using

$$\hat{p} = \frac{\ln(\epsilon_{rh}/\epsilon_h)}{\ln(r)}. \quad (1)$$

If $\hat{p}$ converges to p for $h \to 0$, i.e., when the discretization error is dominated by the lowest order term in the expansion (the so-called asymptotic regime), we can state that the code

is verified and the equations are correctly solved, with the order of accuracy expected for the numerical scheme.

The main issue related to the systematic evaluation of $\hat{p}$ is the need of the analytical solution s, necessary to compute the numerical error ϵ_h, that is unknown in most cases. The MMS has been developed to overcome this problem;[3,20–22] while this method is fairly common in other fields, for example in computational fluid dynamics,[3] to our knowledge, it has never been applied before for a complete and rigorous verification of plasma simulation code.

Instead of solving analytically a theoretical model, the MMS suggests to impose a solution to the model, the so-called manufactured solution, and to modify the model equations to accommodate the imposed solution; we then numerically solve the obtained modified model to compute the discretized error. More precisely, for a given model M, we choose an analytical function u and we compute a source term, $S = M(u)$, which is subsequently subtracted from M to obtain a new analytical model $N = M - S$, whose analytical solution is u [in fact, $N(u) = M(u) - S = 0$]. At this point, it is straightforward to compute the discretization of the new model, i.e., $N_h = M_h - S$. As a matter of fact, since the source term S is computed analytically, we do not add any new discretization errors to the numerical model considered, and, consequently, the behavior of the numerical error is preserved. This can be expressed as: $\epsilon_h = ||u_h - u|| = D_p h^p + O(h^{p+1})$. From a practical point of view, using the MMS for an order-of-convergence test implies adding source terms to the discretized equations, performing a simulation scan to obtain the observed order of accuracy, and comparing the observed order of accuracy to the theoretical one to verify the code.

To conclude the description of the MMS, we note that the initial conditions and the boundary conditions have to be imposed to u_h. Regarding the initial conditions, we impose $u_h|_{t=0} = u|_{t=0}$. When Neumann boundary conditions are considered, we enforce: $(\mathbf{n}\cdot\nabla)_h u_h|_{boundary} = \mathbf{n}\cdot\nabla u|_{boundary}$, where $\mathbf{n}$ is the unitary vector perpendicular to boundary and the operator $(\mathbf{n}\cdot\nabla)_h$ is the discretized derivative used by the code. In the case of Dirichlet boundary conditions, we require $u_h|_{boundary} = u|_{boundary}$. In some cases, for example, in GBS, more elaborated boundary conditions are used, which require the computation of further source terms (see Sec. III for a concrete example).

The idea behind the MMS is trivial; however, its implementation requires to consider some subtleties. First, the manufactured solution should satisfy the following requirements: (i) Be smooth enough, and not singular, to allow the code to attain the asymptotic regime within a reasonable numerical cost, (ii) be general enough to excite all the terms present in the equations and, therefore, have enough non vanishing derivatives to prevent the disappearance of some terms from the equations (however, its derivatives should be bounded by a sufficiently small constant to avoid strongly varying functions over space and time), (iii) satisfy the code constraints (e.g., positivity for the density or the temperature), and (iv) ensure that the magnitude of the different terms composing the equations are of the same order of magnitude (to avoid that a dominating term overshade the value of a subdominant one). Due to these constraints, the manufactured solutions are usually built as a combination of trigonometric and/or hyperbolic functions; as a matter of fact the code verification is a purely mathematical issue and, consequently, as the physics of the problem does not concern the manufactured solutions, no physical constraint is applied on the choice of the analytical functions. Second, the sources, the initial conditions, and the boundary conditions have to be implemented in the code; we need, therefore, access to the source code, where these quantities are computed. Third, the MMS cannot be applied to codes used to model singularities, shocks or discontinuities; the verification of these codes is still an open issue.[2] Finally, care must be taken computing the source terms and applying the boundary conditions, the use of symbolic computational software could result necessary for this purpose.

B. Solution verification methodology

The estimate of the numerical error affecting the simulations is the second step of the verification procedure.[2,3,26,27] In fact, due to the finite computational power available to perform simulations and, consequently, the finite precision achievable, the simulation results are always affected by numerical errors, even if the model equations are implemented correctly. The estimate of the amplitude of the numerical errors is crucial to ensure the reliability of the numerical results, and the knowledge of their magnitude is needed to perform a rigorous validation of the physical model against experimental results.

The numerical errors affecting a simulation have four sources: Round-off errors, iterative errors, statistical sampling errors, and discretization errors.[2,3] The round-off errors are due to the finite precision of computers; assuming that all the computations are performed in double precision, these errors can usually be neglected (we will assume that this is the case in the following). Iterative computational methods present in the code may be a source of error that we neglect here, as no iterative procedures are used by GBS (a complete discussion of this specific subject is given in Ref. 2). The statistical sampling errors entering, for example, in the evaluation of time-averaged quantities used for code validation, can be reduced or eliminated performing averages on steady-state simulations over a long enough time interval, hence, we will assume in the following that simulations are in steady-state and that long enough time can be considered, such that the statistical error vanishes. Consequently, we will focus on the discretization errors, the ones introduced by the numerical scheme used to discretize the physical model over a finite mesh, both in time and in space. Nevertheless, we note that, as it has been shown through gyrokinetic simulations,[28] statistical convergence can be difficult to achieve. For completeness, we should mention that the solution verification procedure does not limit itself to the estimate of the numerical errors; in fact, also the verification of the input parameters and the verification of post-processing tools are part of this process. However, assuming that the input parameters are correctly given and that post-processing tools are

working properly, the main issue of the solution verification procedure reduces to the estimate of the discretization errors.

As an *a priori* study of the numerical scheme to obtain an analytical expression characterizing the behavior of the error is, most of the time, extremely complex to perform, we use an *a posteriori* method to compute the numerical error affecting the simulations. This requires an estimate of the analytical solution, which in most cases is not known.

In the early 20th century, Richardson developed a method,[23,24] later extended,[29,30] to accelerate the rate of convergence of a numerical sequence. This method is based on the use of two numerical solutions obtained using two different meshes, s_h and s_{rh}, to compute a new solution that presents a convergence rate that is, in general, one order higher than the original solution. Concretely, the Richardson extrapolation is defined as

$$\bar{s} = s_h + \frac{s_h - s_{rh}}{r^p - 1}, \tag{2}$$

where p is the formal order of accuracy defined in Sec. II A. Noting that $||s_h - s|| = C_p h^p + O(h^{p+1})$, it follows that the extrapolated solution $\bar{s}$ satisfies $||\bar{s} - s|| = D_p h^{p+1} + O(h^{p+2})$; therefore, for $h \to 0$, $\bar{s} \to s$ faster than the numerical solutions obtained from the simulations. Consequently, we can use $\bar{s}$ as an estimate of the exact solution s and approximate the numerical error with the expression

$$\epsilon_h \simeq \|s_h - \bar{s}\| = \left\| \frac{s_{rh} - s_h}{r^p - 1} \right\|. \tag{3}$$

The relative discretization error (RDE) is therefore approximated as

$$RDE = \frac{s_h - s}{s} \simeq \frac{s_h - \bar{s}}{\bar{s}} = \frac{s_{rh} - s_h}{s_h r^p - s_{rh}}. \tag{4}$$

For $\bar{s}$ to be a reasonable estimate of s, however, several assumptions should be satisfied. First, the Richardson extrapolation method requires the use of uniform mesh spacing, meaning that the degree of the refinement of the meshes can be represented solely by the parameter h discussed before. Therefore, s_h and s_{rh} should be computed over two meshes that are one the uniform systematic refinement of the other one and, consequently, the application of Richardson extrapolation to computations involving local mesh refinement or mesh adaptation is not allowed. Second, the simulations used to evaluate $\bar{s}$ should be in the asymptotic regime, meaning that the discretization error is dominated by its lower order term, $C_p h^p$. This requirement could result in computationally very expensive simulations, due to the potential need of very fine meshes. Third, to apply the method presented above for the estimate of the numerical error, it is required that the solutions are smooth enough and do not present singularities and/or discontinuities. In fact, to allow the expansion of the numerical error in term of powers of the parameter h, the derivatives of the analytical solution should exist and be continuous. Moreover, we should 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.

Usually, it is problematic to satisfy the requirement of being in the asymptotic regime, due to the high computational cost of the simulations. Moreover, it has to be demonstrated that the numerical solutions are in the asymptotic regime, by showing that the observed order of accuracy matches the formal one. This requires at least three simulations, resulting from two subsequently refinements of the coarser mesh of a factor r, from which the observed order of accuracy can be evaluated as

$$\hat{p} = \frac{\ln\left[(s_{r^2h} - s_{rh})/(s_{rh} - s_h)\right]}{\ln(r)}. \tag{5}$$

If only two simulations are available, or if the observed order of accuracy does not match the formal one, we should substitute the numerical error estimates in Eqs. (3) and (4) with a numerical uncertainty quantification. As a matter of fact, in general, the error estimate in Eqs. (3) and (4) may depend strongly on the refinement factor r and on the precision of the numerical scheme used by the model; it is, therefore, difficult to rely on such error estimate. To overcome these issues, Ref. 25 introduces the GCI, defined as

$$GCI = \frac{F_s}{r^{\tilde{p}} - 1} \left| \frac{s_{rh} - s_h}{s_h} \right|, \tag{6}$$

that represents an estimate of the relative discretization error affecting the simulation results. The GCI is obtained by approximating in Eq. (4) $s_h r^p - s_{rh} \simeq (r^{\tilde{p}} - 1) s_h$. The factor of safety F_s and $\tilde{p}$ ensure that the GCI is larger than the numerical discretization error in 95% of the cases. Oberkampf and Roy[2] propose the following: if the difference between p and $\hat{p}$ is less than 10%, we can assume that the simulation is in the asymptotic regime, and we use $F_s = 1.25$, as well as $\tilde{p} = p$. If the difference between p and $\hat{p}$ is larger than 10%, a more conservative factor of safety, $F_s = 3$, has to be 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 $\tilde{p} = p$ are used. We remark that, although these definitions are reasonable, there still is an ongoing discussion in the verification community about their generality.

To conclude our presentation of the error estimate methodology, we discuss a few details. First of all, we draw the attention to the fact that the presented procedure can be applied not only to point-by-point solution values but also to solution functionals. This is important for the use of this methodology to estimate the numerical error affecting the observables used in the validation of the physical model.[3] Second, as s_h and s_{rh} are, in general, computed on different meshes, the results on the coarser mesh have to be interpolated on the finest grid, using an interpolation scheme whose order is equal or higher than the order of the numerical scheme used by the code. A complete discussion of this topic is found in Ref. 29. Finally, we illustrate a useful propriety of the GCI, that is the possibility of computing the overall GCI analyzing each coordinate of the problem independently. As

it can result numerically very expensive to perform a uniform refinement of the grid along all the coordinates at the same time, it is possible to refine separately each coordinate of the mesh by a factor r_i, where the index i refers to the coordinate under investigation. This allows us to compute a GCI_i and a $\tilde{p}_i$ for the i coordinate, and obtain the overall GCI as $GCI = \sum_i GCI_i$.

III. APPLICATION OF THE VERIFICATION METHODOLOGY TO THE GBS CODE

To illustrate a concrete example of application of the methodology discussed above for the verification of a plasma turbulence simulation code, we apply the procedure to the GBS code. This code is a fluid code that has been used to simulate plasma turbulence in magnetic confinement devices for fusion and basic plasma physics experiments. It constitutes a test bed for the proposed verification methodology.

First, the GBS code is presented (Sec. III A), in particular, we discuss the fields and the operators used by the code, and we illustrate the numerical scheme implemented to discretize the model equations. Second, using the MMS procedure, an order-of-accuracy test is executed (Sec. III B) to verify the correct implementation of the physical model in GBS. Finally, some physical relevant simulations are performed (Sec. III C) using several different meshes, and the numerical error affecting the simulation results is quantified.

A. The GBS code

The GBS code has been developed in the last few years to simulate plasma turbulence in the open field region of magnetic confinement devices, evolving the full plasma profiles, without any separation between equilibrium and perturbation quantities.[19] To develop the GBS code, increasingly complex magnetic configurations have been considered: First, the code was developed to describe basic plasma physics devices, in particular, linear devices such as LAPD[31] and simple magnetized toroidal devices such as TORPEX;[32–34] it was then extended to the tokamak geometry, and it is now able to model the tokamak SOL region in limited plasmas.[35–38] A validation methodology has been developed to assess the predictive capability of GBS; the procedure has been performed by comparing GBS simulations with TORPEX experimental results.[6,7]

To describe the plasma dynamics, GBS uses the Braginskii equations[39] in the drift approximation, which is valid for $d/dt \ll \omega_{ci}$ (where $\omega_{ci} = eB/m_i$ is the ion gyrofrequency) and $|\nabla_\parallel| \ll |\nabla_\perp|$.[40,41] For the purpose of the present paper, only the electrostatic model of a SOL tokamak in the infinite aspect ratio limit is considered. Moreover, the Boussinesq approximation (its use for turbulent dynamics models is discussed in Refs. 42–44) and the cold ion approximation ($T_i \ll T_e$) are used. Under these assumptions, the equations constituting the drift-reduced Braginskii model are

$$\partial_t n = -\frac{R}{\rho_{s0}}[\phi, n] + 2\left[C(p_e) - nC(\phi)\right] - \nabla_\parallel(n v_{\parallel e}) + D_n \nabla_\perp^2 n + S_n, \tag{7}$$

$$\partial_t \omega = -\frac{R}{\rho_{s0}}[\phi, \omega] + \frac{2}{n}C(p_e) - v_{\parallel i}\nabla_\parallel \omega + \frac{1}{n}\nabla_\parallel j_\parallel + \frac{1}{3n}C(G_i) + D_\omega \nabla_\perp^2 \omega, \tag{8}$$

$$\partial_t v_{\parallel e} = -\frac{R}{\rho_{s0}}[\phi, v_{\parallel e}] + \frac{m_i}{m_e}\left[\nabla_\parallel \phi - \frac{1}{n}\nabla_\parallel p_e - 0.71 n \nabla_\parallel T_e + \nu j_\parallel - \frac{2}{3n}\nabla_\parallel G_e\right] - v_{\parallel e}\nabla_\parallel v_{\parallel e} + D_{v_{\parallel e}}\nabla_\perp^2 v_{\parallel e}, \tag{9}$$

$$\partial_t v_{\parallel i} = -\frac{R}{\rho_{s0}}[\phi, v_{\parallel i}] - v_{\parallel i}\nabla_\parallel v_{\parallel i} - \frac{1}{n}\nabla_\parallel p_e - \frac{2}{3n}\nabla_\parallel G_i + D_{v_{\parallel i}}\nabla_\perp^2 v_{\parallel i}, \tag{10}$$

$$\partial_t T_e = -\frac{R}{\rho_s}[\phi, T_e] + \frac{4}{3}T_e\left[\frac{7}{2}C(T_e) + \frac{T_e}{n}C(n) - C(\phi)\right] - v_{\parallel e}\nabla_\parallel T_e + S_{T_e} + \frac{2}{3}T_e\left[0.71\nabla_\parallel v_{\parallel i} - 1.71\nabla_\parallel v_{\parallel e} + 0.71\left(\frac{v_{\parallel i} - v_{\parallel e}}{n}\right)\nabla_\parallel n\right] + D_{T_e}\nabla_\perp^2 T_e, \tag{11}$$

where $j_\parallel = n(v_{\parallel i} - v_{\parallel e})$ is the parallel current, $p_e = nT_e$ is the electron pressure, and ν is the plasma resistivity. The system is closed by the Poisson's equation $\omega = \nabla_\perp^2 \phi$. The Poisson brackets are defined as $[f, g] = \mathbf{b} \cdot (\nabla f \times \nabla g)$, the parallel gradient as $\nabla_\parallel = \mathbf{b} \cdot \nabla$, and the curvature operator $C(a) = B/2[\nabla \times (\mathbf{b}/B)] \cdot \nabla a$, where $\mathbf{b}$ is the unitary vector oriented along $\mathbf{B}$. The plasma outflow coming from the closed flux surfaces region is mimicked by a density source S_n and an electron temperature source S_{Te}. The expressions of the two terms representing the gyroviscous contribution are, respectively, given by $G_i = -\eta_{0i}\left[2\nabla_\parallel v_{\parallel i} + C(\phi)\right]$ and $G_e = -\eta_{0e}\left[2\nabla_\parallel v_{\parallel e} - C(p_e)/n + C(\phi)\right]$. Small perpendicular diffusion terms of the form $D_a \nabla_\perp^2 a$ are added for numerical reasons. All quantities are normalized according to (tilde denotes a physical quantity in MKS units): $t = \tilde{t}/(R/c_{s0})$, $n = \tilde{n}/n_0$, $T_e = \tilde{T}_e/T_{e0}$, $\phi = e\tilde{\phi}/T_{e0}$, $v_{\parallel e} = \tilde{v}_{\parallel e}/c_{s0}$, $v_{\parallel i} = \tilde{v}_{\parallel i}/c_{s0}$, $B = \tilde{B}/B_0$, $\nu = (e^2 n_0 R)/(m_i \sigma_\parallel c_{s0})$, where $\sigma_\parallel$ is the parallel conductivity, n_0, T_{e0}, and B_0 are the reference density, temperature, and magnetic field, while the normalized quantities c_{s0} and ρ_{s0} are given by $c_{s0} = \sqrt{T_{e0}/m_i}$ and $\rho_{s0} = c_{s0} m_i c/(eB_0)$. Distances perpendicular to $\mathbf{B}$ are normalized to ρ_{s0}, while parallel distances are normalized to R.

To carry out the verification exercise, we consider a limited tokamak configuration with circular magnetic flux surfaces and a toroidal limiter on the high-field side equatorial midplane, with no magnetic shear. Starting from the toric coordinate system (θ, r, φ), where θ is the straight-field-line coordinate in the infinite aspect ratio limit and φ is the toroidal angle, we introduce the right-handed coordinate system (y, x, z) used by GBS as $y = a\theta$, $x = r - a$, and $z = \varphi$, where a is the minor radius of the device. Consequently, the operators can be rewritten as $C = -\sin\theta\partial_x - \cos\theta\partial_y$, $[f, g] = \partial_x g \partial_y f - \partial_x f \partial_y g$, $\nabla_\perp^2 = \partial_x^2 + \partial_y^2$, and $\nabla_\parallel = \partial_z + a/(qR)\partial_y$, where q is the safety factor and $\theta = y/a$ is the poloidal angle defined such that $\theta = 0$ and

$\theta = 2\pi$ are the equatorial high-field side midplane corresponding to the limiter position.

Equations (7)–(11) constituting the GBS model are completed by a set of boundary conditions, which describe the interface with the magnetic pre-sheath.[45] They are given by

$$v_{\parallel i} = \pm c_s, \tag{12}$$

$$v_{\parallel e} = \pm c_s \exp(\Lambda - \phi/T_e), \tag{13}$$

$$\partial_y T_e = \kappa_T \partial_y \phi, \tag{14}$$

$$\partial_y n = \mp \frac{n}{c_s} \partial_y v_{\parallel i}, \tag{15}$$

$$\omega = -\cos^2 \alpha \left[\left(\partial_y v_{\parallel i} \right)^2 \pm c_s \partial_y^2 v_{\parallel i} \right], \tag{16}$$

$$\partial_y \phi = \mp c_s \partial_y v_{\parallel i}, \tag{17}$$

where $\Lambda \approx 3$, $\cos^2\alpha$ is assumed equal to 1 [$\alpha = a/(qR) \ll 1$], $\kappa_T \approx 0.1$, and the radial gradients are neglected. Here, the upper signs apply to the case of magnetic field directed towards the wall, while the lower ones apply to the opposite case.

To ensure the positivity of the plasma density and of the electron temperature, these two quantities are implemented using the relations $n = \exp(\theta)$ and $T_e = \exp(t_e)$, being the quantities θ and t_e the fields evolved by GBS. Equations (7)–(11) are consequently rewritten in terms of these two fields.

To solve Eqs. (7)–(11) and Poisson's equation, we use a second-order finite difference scheme in the spatial dimensions, while the Poissons brackets are discretized with a second order Arakawa scheme.[46] Time is advanced using a standard fourth-order Runge-Kutta scheme. The boundary conditions [Eqs. (12)–(17)] are implemented in the model using a second-order finite difference scheme in the spatial dimensions. Consequently, the expected orders of accuracy characterizing the numerical model are $p_s = 2$, in the spatial directions, and $p_t = 4$, for the time discretization. Defining $h = \Delta y/\Delta y_0 = \Delta x/\Delta x_0 = \Delta z/\Delta z_0 = (\Delta t/\Delta t_0)^2$, we expect an overall $p = 2$ for the numerical scheme.

B. Correct implementation of the model equations (code verification)

In order to verify that the model described in Sec. III A is correctly coded in GBS, the methodology illustrated in Sec. II A is applied proceeding as follows. First, the discretization scheme used to solve Eqs. (7)–(11) and the Poisson's equation is analyzed, using, for simplicity, Dirichlet (for $v_{\parallel i}$, $v_{\parallel e}$, and ω) and Neumann (for n, T_e, and ϕ) boundary conditions. Dirichlet boundary conditions are applied at the grid points (therefore, no numerical error results), and Neumann boundary conditions are discretized with a second-order numerical scheme. Second, we study the order of accuracy characterizing the discretization scheme of the boundary conditions, Eqs. (12)–(17), decoupling these from the solution of Eqs. (7)–(11). Finally, the two sets of equations [Eqs. (7)–(11) and Eqs. (12)–(17)] are coupled to complete the verification of the overall GBS code. The code verification methodology is divided in these three steps to simplify the investigation of possible implementation errors; in the present paper, we show only the final results [i.e., verification of Eqs. (7)–(11) coupled with Eqs. (12)–(17)], which summarize the verification results obtained for GBS. We remark that the methodology for the code verification allowed us to find and correct a minor bug, related to the discretization of the G_i and G_e terms at the boundaries. Luckily, we were able to verify that the generated numerical error was very small, and its influence on the previous GBS results completely negligible.

To verify the implementation of the drift-reduced Braginskii equations into GBS and to satisfy the requirements given in Sec. II A, we choose to manufacture the model solution as the combination of trigonometric functions. More precisely, the functions used to represent the six fields appearing in Eqs. (7)–(11) are expressed as

$$f(y,x,z,t) = A_f \left\{ B_f + \sin\left[C_f \left(z - \frac{q}{a} y \right) \right] \times \sin(D_f y) \sin(E_f t + F_f x) \right\}, \tag{18}$$

where A_f, B_f, C_f, D_f, E_f, and F_f are arbitrary constants and $f = n, T_e, v_{\parallel i}, v_{\parallel e}, \omega, \phi$ are the fields present in GBS equations. The B_f's are used to ensure the positivity of n and T_e, the others coefficients to calibrate the amplitude of the errors in order to guarantee that there is no dominating term in the equations. This means that the amplitude of the coefficients is chosen such that, for the used meshes, the simulations are in the asymptotic regime and the errors affecting the different terms of Eqs. (7)–(11) are of the same magnitude. As GBS is developed to simulate turbulent modes mainly aligned to the field lines, we impose the dependence on y and z as the product of two terms: The first one perfectly aligned to the field lines (the term containing C_f) and a second term (containing D_f) representing a perturbation in the poloidal direction (i.e., along the y coordinate), chosen small, not to have the discretization error on the parallel derivative dominating over all the others. The E_f and F_f terms introduce the time and radial dependencies. We note that C_f must be an integer, to satisfy the periodicity of the system along the z coordinate, and that the manufactured solutions are defined for the two fields n_e and T_e, while the GBS code evolves the two fields θ and t_e. Consequently, by performing the order-of-accuracy test, we study the behavior of the numerical error characterizing the two fields of physical interest, n_e and T_e.

The computation of the source terms is trivial: it consists in plugging the analytical functions presented in Eq. (18) into Eqs. (7)–(11) and in Poisson's equation to obtain the source term S. This process is particularly tedious, but it involves only straightforward algebraic manipulations with no conceptual difficulties. As the results of these computations do not present any theoretical interest, we do not present those herein. We just mention that we compute the source terms using the symbolic manipulation software *Mathematica*,[47] which allows the direct translation into

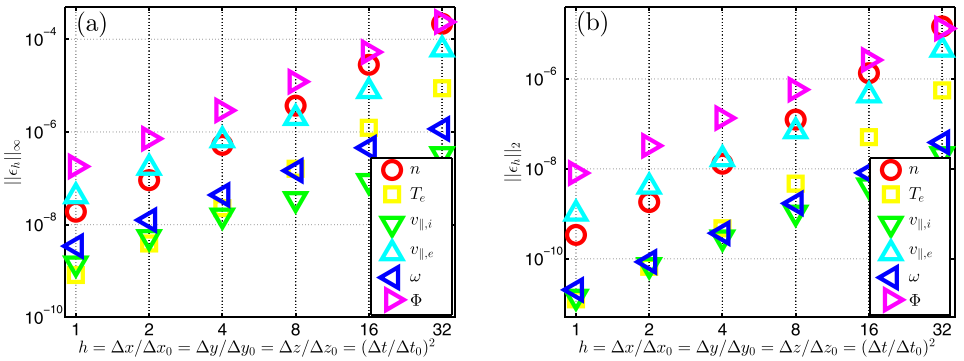

FIG. 1. Norm of the numerical error affecting the discretization scheme used in GBS, plotted as function of the refinement degree h, for the two norms L_∞ (a) and L_2 (b).

Fortran language. This enables the implementation of the obtained expressions in GBS, without any significant difficulty and reducing the possibilities of mistakes.

The verification of the boundary conditions described by Eqs. (12)–(17) requires the computation of additional source terms. In fact, the manufactured solutions given in Eq. (18) do not satisfy the boundary conditions; consequently, as done for the equations governing the physics of the SOL region, we insert the manufactured solution into Eqs. (12)–(17), and we add the resulting source terms to the boundary conditions equations.

For the computation of the error and to estimate $\hat{p}$, we consider the two norms, L_2 (i.e., $\|\mathbf{f}\|_2 = \sqrt{\sum_i^N f_i^2/N}$) and L_∞ (i.e., $\|\mathbf{f}\|_\infty = \max|f_i|$); L_2 is appropriate to ensure the correct global convergence of the results, while L_∞ is used to assess the local convergence in all points of the domain. Figures 1 and 2 summarize the GBS verification results. Six simulations are performed with $h = 1, 2, 4, 8, 16, 32$, and the corresponding errors, computed using the L_2 and L_∞ norms, are shown in Fig. 1. We observe that the numerical error clearly decreases when refining the mesh (i.e., decreasing the value of h); on a logarithmic scale, the numerical error decreases linearly, with slope p, as expected. We also note that our scan leads to a reduction of the numerical error by at least three orders of magnitude, this gives confidence that there are not subdominant errors decreasing at a rate different than the expected one. The estimate of the observed order of accuracy, evaluated according to Eq. (1), is plotted as a function of the parameter h in Fig. 2. Clearly, with the refinement of the meshes, $\hat{p}$ tends to p for all the fields, as expected, although the rate of convergence is field dependent. This is due to the fact that the coefficients of the Taylor expansion of the numerical error are different for each field. Consequently, we demonstrate that Eqs. (7)–(11), the Poisson's equation, and the boundary conditions [Eqs. (12)–(17)] are correctly coded in GBS, with a numerical scheme that satisfies the theoretical order of convergence.

C. Numerical error estimate (solution verification)

The estimate of the numerical error affecting a simulation is needed not only to ensure the reliability of the numerical results, but also to perform the validation of the physical model. Therefore, the quantification of the numerical error is a fundamental process of the verification methodology. In this subsection, we apply the approach presented in Sec. II B to the GBS code to exemplify the procedure and to assess the reliability of the GBS results.

GBS has been used to study quantities like temporal and spatial averages of $v_{\parallel i}$ and $v_{\parallel e}$, $\langle v_{\parallel i}\rangle$ and $\langle v_{\parallel e}\rangle$; time-averaged radial profiles, e.g., of $p_e = nT_e$ and ϕ; and the pressure equilibrium scale length, $L_p = -p_e/\nabla p_e$. In the following, we focus on numerical error affecting these quantities, that are functionals of the GBS solutions. The time-average of these quantities is done in the time interval $40 \le t \le 80$, during which the turbulence is in a quasi-steady state (except n, which still shows a secular trend, although relatively weak, and $v_{\parallel i}$ and $v_{\parallel e}$, which present an even weaker secular trend). Moreover, we consider a standard SOL simulation characterized by $q = 4$, $\nu = e^2 n_0 R/(m_i \sigma_\parallel c_{s0}) = 0.1$, $m_e/m_i = 200$, $L_y = 400$, $R = 500$ (see Ref. 38 for the physical investigation of those results).

To apply the methodology described in Sec. II B, we analyze separately the spatial and the temporal coordinates. More precisely, in order to obtain the Richardson extrapolation, Eq. (2), and to compute the observed order of accuracy, Eq. (5), we execute five simulations using five different

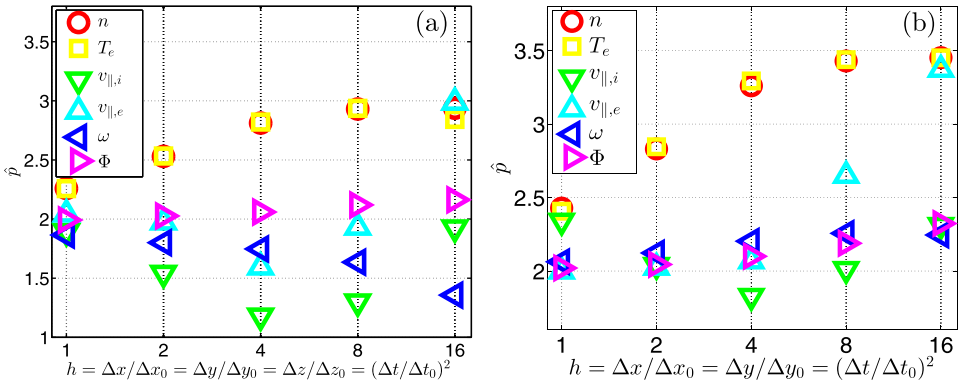

FIG. 2. Observed order of accuracy, $\hat{p}$, characterizing the discretization scheme of GBS, computed applying Eq. (1) and plotted as function of the refinement degree h, for the two norms L_∞ (a) and L_2 (b).

TABLE I. Values of L_p, $\langle v_{\|i}\rangle$ and $\langle v_{\|e}\rangle$ computed on five different meshes, as characterized by h_s and h_t.

Grid ($n_y \times n_x \times n_z$)	Time step	h_s	h_t	L_p	$\langle v_{\|i}\rangle$	$\langle v_{\|e}\rangle$
$192 \times 80 \times 24$	3.00×10^{-5}	2.25	1.00	25.56	−0.039	0.066
$288 \times 120 \times 36$	3.00×10^{-5}	1.50	1.00	27.22	−0.067	0.091
$432 \times 180 \times 54$	3.00×10^{-5}	1.00	1.00	27.22	−0.070	0.100
$432 \times 180 \times 54$	4.50×10^{-5}	1.00	1.50	26.67	−0.071	0.100
$432 \times 180 \times 54$	6.75×10^{-5}	1.00	2.25	23.89	−0.069	0.107

TABLE II. Values of GCI_s and RDE computed using the parameters $\hat{p}_s$, $\tilde{p}_s$, and F_s, valid for the finest mesh [$h_s = h_t = 1.0$].

Field	$\hat{p}_s$	$\tilde{p}_s$	F_s	GCI_s (%)	RDE (%)
$\langle v_{\|i}\rangle$	3.41	2	3	20.7	−6.5
$\langle v_{\|e}\rangle$	2.87	2	3	20.5	−6.4
p_e	1.86	2	1.25	12.0	−8.8
ϕ	3.08	2	3	7.3	−2.4

meshes: Starting from the most refined mesh (in space and time), we perform two subsequent spatial grid coarsening by $r_s = 1.5$; the remaining two meshes result from the subsequent multiplication of the time step by 1.5 ($r_t = 1.5$), without any change of the spatial grid. Hence, the five meshes can be characterized by two parameters, $h_s = \Delta y/\Delta y_0 = \Delta x/\Delta x_0 = \Delta z/\Delta z_0$ and $h_t = \Delta t/\Delta t_0$, where $\Delta y_0 = 0.93$, $\Delta x_0 = 0.56$, $\Delta z_0 = 0.12$, $\Delta t_0 = 3.0 \times 10^{-5}$, h_s describes the discretization in the spatial coordinates, and h_t defines the degree of refinement of the time step.

For the five simulations considered, the quantities of interest are listed in Table I (L_p, $\langle v_{\|i}\rangle$, and $\langle v_{\|e}\rangle$) and shown in Fig. 3 (radial profiles of p_e and ϕ). We note that $\langle v_{\|i}\rangle$ and $\langle v_{\|e}\rangle$ are computed taking the average of the parallel velocities over the entire spatial domain of interest. The radial profiles of p_e and ϕ are obtained taking the average of these quantities along the poloidal and toroidal directions; the pressure equilibrium scale length is computed as the radial distance between the maximum value of the radial profile of p_e and the half of its maximum value.

The results presented in Table I and in Fig. 3 show that the differences of the various quantities computed on the meshes characterized by $h_t = 1.00, 1.50, 2.25$ are very small, if compared to the changes due to the spatial discretization. The only quantity presenting a meaningful dependence on the time step is L_p; for this quantity we apply the methodology described in Sec. II B, finding $\hat{p}_t = 3.97$ and $GCI_t = 0.6\%$ (here, GCI_t is referred to the most refined mesh; the GCI_t value relative to the other meshes is obtained by multiplying the GCI_t of the most refined mesh by h_t^4, and similarly for the spatial discretization). The observed order of accuracy is remarkably close to the expected $p_t = 4$, and the resulting numerical error is very small. Therefore, in the following, we neglect the numerical error associated to the time discretization with respect to the one due to spatial discretization.

The evaluation of the numerical error due to the spatial discretization affecting the quantities of interest is summarized in Table II. We start our analysis by considering $\langle v_{\|i}\rangle$ and $\langle v_{\|e}\rangle$. By applying Eq. (5) to these quantities, we obtain a value of $\hat{p}_s$ larger than p_s. The difference between $\hat{p}_s$ and p_s is probably due to the fact that, in the present scenarios, the parallel velocities average to very small quantities, if compared to the local value of $v_{\|i}$ and $v_{\|e}$ (even one order of magnitude lower). Therefore, $\langle v_{\|i}\rangle$ and $\langle v_{\|e}\rangle$ are very sensitive quantities; this can lead, not surprisingly, to a difference between $\hat{p}_s$ and p_s. Using Eq. (6) and the conservative value $F_s = 3$, the resulting GCI_s are relatively large.

The analysis of the radial profiles of p_e and ϕ is very similar: depending on the difference between $\hat{p}_s$ and p_s, we choose the corresponding value of F_s and $\tilde{p}_s$, by which we compute the GCI_s. At a mesh similar to the one generally used for GBS simulations, we find that the numerical error affecting these quantities is of the order of 20%–25%.

Finally, we note that the differences between the values of L_p computed on the three meshes characterized by $h_s = 1.00, 1.50, 2.25$ are of the same order of the spatial grid size ($\Delta x = 0.56$ for $h_s = 1.0$) and, therefore, below the numerical error necessary to perform the Richardson extrapolation (in fact, Δx is the intrinsic uncertainty on L_p; therefore, it is not possible to distinguish between two values of L_p whose difference is below or equal to $2\Delta x$). It follows that we can

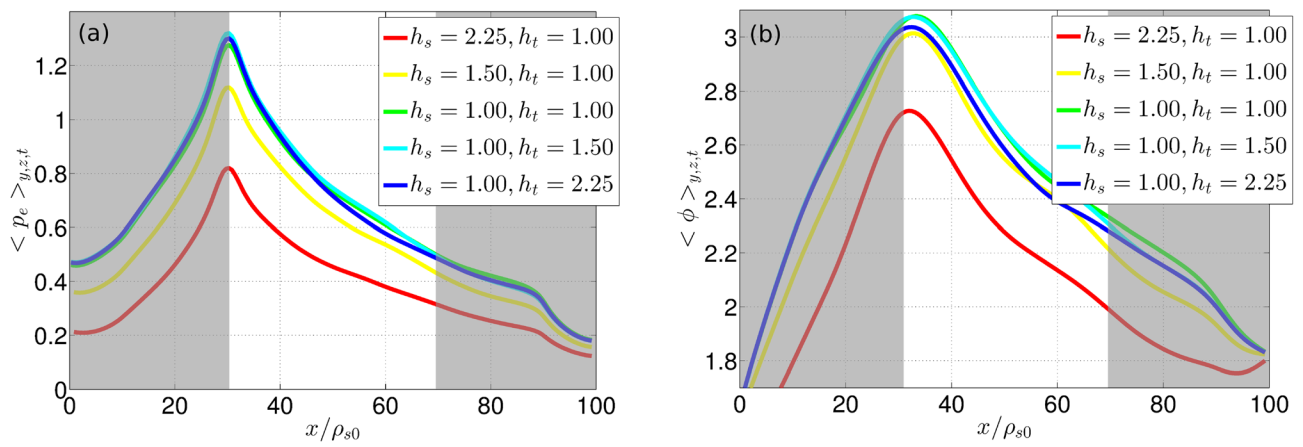

FIG. 3. Radial profiles of p_e (a) and ϕ (b), averaged over time and along the toroidal and poloidal directions, for five meshes, as characterized by h_s and h_t.

assume the numerical error affecting L_p comparable to the spatial grid size.

Several observations should be addressed to our results. First, the quantities are clearly converging at a rate that is typically not very different from the expected one. Second, our analysis allow us to estimate the numerical error affecting the different quantities of interest; this will be used in the future to choose the mesh refinement necessary to achieve the desired accuracy. In any case, it is reassuring that, the correct qualitative behavior is retrieved by all GBS simulations even at the coarser meshes. Third, as the value of $F_s = 3$ is generally thought to be conservative, our estimate of the numerical error is quite safe. Finally, throughout our solution verification, we have assumed that statistical errors are negligible. As a matter of fact, the most refined simulation is computationally extremely expensive, and we could not verify this assumption. Moreover, as previously pointed out, n is not in perfect steady state; it is possible that the value of the GCI is reduced by considering longer time intervals.

IV. CONCLUSION

In the present paper, we discuss the methodology for plasma simulation code verification, proposing concrete approaches for the verification of the coding of equations (code verification) and numerical error quantification (solution verification). The methodology we propose for the verification of plasma simulation codes is general, rigorous, simple-to-apply, and does not present any conceptual difficulties. Code verification requires to choose an adequate manufactured solution which satisfies some reasonable assumptions; then, the source terms to be added to the model equations, as well as the boundary conditions, are readily evaluated. At this point, it is possible to compute $\hat{p}$ [Eq. (1)] by performing a number of simulations corresponding to more and more refined meshes. If $\hat{p} \to p$ for $h \to 0$, then the code is verified. On the other hand, the procedure we propose to quantify the numerical error is definitely valid for simulations belonging to the asymptotic regime; if this condition is satisfied and the assumptions required to apply the Richardson extrapolation are met, the implementation of the solution verification methodology is trivial. For simulations not belonging to the asymptotic regime, the GCI still allows to estimate the numerical uncertainty.

The application of the proposed procedure to the GBS code allowed us to find and correct a minor bug that was generating very small numerical error, with completely negligible influence on the previous GBS results. This shows the power of the proposed methodology. The solution verification gave an estimate of the amplitude of the numerical error affecting the GBS results, useful for the validation of the code results with experimental data.

The final result of the study described herein is that the implementation of the physical model in the GBS code has been completely and rigorously verified, ensuring the correct solution of the model equations and bounding the numerical error affecting the simulation results. As a matter of fact, the verification exercise largely increases the confidence on the numerical results obtained using the GBS code. We believe that such procedure could become a standard of reference for all the codes used in plasma simulations.

ACKNOWLEDGMENTS

Part of the simulations presented herein were carried out at the Swiss National Supercomputing Centre (CSCS) under Project ID s346; and part were carried out using the HELIOS supercomputer system at the Computational Simulation Centre of the International Fusion Energy Research Centre (IFERC-CSC), Aomori, Japan, under the Broader Approach collaboration between Euratom and Japan, implemented by Fusion for Energy and JAEA. This research was supported by the Swiss National Science Foundation.

[1]W. L. Oberkampf and T. G. Trucano, Prog. Aerosp. Sci. **38**, 209 (2002).
[2]W. L. Oberkampf and C. J. Roy, *Verification and Validation in Scientific Computing* (Cambridge University Press, New York, NY, USA, 2010).
[3]P. J. Roache, *Verification and Validation in Computational Science and Engineering* (Hermosa Publishers, Albuquerque, NM, USA, 1998).
[4]P. W. Terry, M. Greenwald, J.-N. Leboeuf, G. R. McKee, D. R. Mikkelsen, W. M. Nevins, D. E. Newman, and D. P. Stotler, Phys. Plasmas **15**, 062503 (2008).
[5]M. Greenwald, Phys. Plasmas **17**, 058101 (2010).
[6]P. Ricci, C. Theiler, A. Fasoli, I. Furno, B. Labit, S. H. Muller, M. Podesta, and F. M. Poli, Phy. Plasmas **16**, 055703 (2009).
[7]P. Ricci, C. Theiler, A. Fasoli, I. Furno, K. Gustafson, D. Iraji, and J. Loizu, Phys. Plasmas **18**, 032109 (2011).
[8]T. Rhodes, C. Holland, S. Smith, A. White, K. Burrell, J. Candy, J. DeBoo, E. Doyle, J. Hillesheim, J. Kinsey, G. McKee, D. Mikkelsen, W. Peebles, C. Petty, R. Prater, S. Parker, Y. Chen, L. Schmitz, G. Staebler, R. Waltz, G. Wang, Z. Yan, and L. Zeng, Nucl. Fusion **51**, 063022 (2011).
[9]N. Howard, A. White, M. Reinke, M. Greenwald, C. Holland, J. Candy, and J. Walk, Nucl. Fusion **53**, 123011 (2013).
[10]M. Shimada, D. Campbell, V. Mukhovatov, M. Fujiwara, N. Kirneva, K. Lackner, M. Nagami, V. Pustovitov, N. Uckan, J. Wesley, N. Asakura, A. Costley, J. Donné, E. Doyle, A. Fasoli, C. Gormezano, Y. Gribov, O. Gruber, T. Hender, W. Houlberg, S. Ide, Y. Kamada, A. Leonard, B. Lipschultz, A. Loarte, K. Miyamoto, T. Osborne, A. Polevoi, and A. Sips, Nucl. Fusion **47**, S1 (2007).
[11]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, Phys. Plasmas **7**, 969 (2000).
[12]J. Birn, J. F. Drake, M. A. Shay, B. N. Rogers, R. E. Denton, M. Hesse, M. Kuznetsova, Z. W. Ma, A. Bhattacharjee, A. Otto, and P. L. Pritchett, J. Geophys. Res. **106**, 3715, doi:10.1029/1999JA900449 (2001).
[13]P. Ricci, J. U. Brackbill, W. Daughton, and G. Lapenta, Phys. Plasmas **11**, 4102 (2004).
[14]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, Plasma Phys. Controlled Fusion **50**, 124015 (2008).
[15]R. V. Bravenec, Y. Chen, J. Candy, W. Wan, and S. Parker, Phys. Plasmas **20**, 104506 (2013).
[16]W. L. Oberkampf and T. G. Trucano, Nucl. Eng. Des. **238**, 716 (2008).
[17]C. S. Chang, S. Ku, P. Diamond, M. Adams, R. Barreto, Y. Chen, J. Cummings, E. D'Azevedo, G. Dif-Pradalier, S. Ethier, L. Greengard, T. S. Hahm, F. Hinton, D. Keyes, S. Klasky, Z. Lin, J. Lofstead, G. Park, S. Parker, N. Podhorszki, K. Schwan, A. Shoshani, D. Silver, M. Wolf, P. Worley, H. Weitzner, E. Yoon, and D. Zorin, J. Phys.: Conf. Ser. **180**, 012057 (2009).
[18]D. Twarog, R. Stankiewicz, and K. Drozdowicz, "Test of the European transport solver in the frame of integrated tokamak modelling," Technical Report No. 2051/AP, Institute of Nuclear Physics, Polish Academy of Sciences, 2011.

[19]P. Ricci, F. D. Halpern, S. Jolliet, J. Loizu, A. Mosetto, A. Fasoli, I. Furno, and C. Theiler, Plasma Phys. Controlled Fusion **54**, 124047 (2012).
[20]S. Steinberg and P. J. Roache, J. Comput. Phys. **57**, 251 (1985).
[21]P. J. Roache, J. Fluids Eng. **124**, 4 (2002).
[22]C. J. Roy, J. Comput. Phys. **205**, 131 (2005).
[23]L. F. Richardson, Philos. Trans. R. Soc., A **210**, 307 (1911).
[24]L. F. Richardson and J. A. Gaunt, Philos. Trans. R. Soc. A **226**, 299 (1927).
[25]P. J. Roache, J. Fluids Eng. **116**, 405 (1994).
[26]P. J. Roache, Annu. Rev. Fluid Mech. **29**, 123 (1997).
[27]F. Stern, R. V. Wilson, H. W. Coleman, and E. G. Paterson, J. Fluids Eng. **123**, 793 (2001).
[28]B. F. McMillan, S. Jolliet, T. M. Tran, L. Villard, A. Bottino, and P. Angelino, Phys. Plasmas **15**, 052308 (2008).
[29]P. J. Roache and P. M. Knupp, Commun. Numer. Methods Eng. **9**, 365 (1993).
[30]S. A. Richards, Commun. Numer. Methods Eng. **13**, 573 (1997).
[31]B. N. Rogers and P. Ricci, Phys. Rev. Lett. **104**, 225002 (2010).
[32]P. Ricci, B. N. Rogers, and S. Brunner, Phys. Rev. Lett. **100**, 225002 (2008).
[33]P. Ricci and B. N. Rogers, Phys. Plasmas **16**, 062303 (2009).
[34]P. Ricci and B. N. Rogers, Phys. Rev. Lett. **104**, 145001 (2010).
[35]F. D. Halpern, S. Jolliet, J. Loizu, A. Mosetto, and P. Ricci, Phys. Plasmas **20**, 052306 (2013).
[36]F. Halpern, P. Ricci, B. Labit, I. Furno, S. Jolliet, J. Loizu, A. Mosetto, G. Arnoux, J. Gunn, J. Horacek, M. Kočan, B. LaBombard, and C. Silva, Nucl. Fusion **53**, 122001 (2013).
[37]A. Mosetto, F. D. Halpern, S. Jolliet, J. Loizu, and P. Ricci, Phys. Plasmas **20**, 092308 (2013).
[38]P. Ricci and B. N. Rogers, Phys. Plasmas **20**, 010702 (2013).
[39]S. I. Braginskii, "Transport processes in a plasma," in *Reviews of Plasma Physics*, edited by M. A. Leontovich (Consultants Bureau Enterprise, New York, 1965), Vol. 1, p. 205.
[40]R. Hazeltine and J. Meiss, Phys. Rep. **121**, 1 (1985).
[41]A. Zeiler, J. F. Drake, and B. Rogers, Phys. Plasmas **4**, 2134 (1997).
[42]D. A. Russell, D. A. D'Ippolito, and J. R. Myra, Bulletin of the American Physical Society, 54th Annual Meeting of the APS Division of Plasma Physics, Vol. 57, No. 12 (2012), available at http://meetings.aps.org/link/BAPS.2012.DPP.BP8.159.
[43]G. Q. Yu, S. I. Krasheninnikov, and P. N. Guzdar, Phys. Plasmas **13**, 042508 (2006).
[44]K. Bodi, G. Ciraolo, P. Ghendrih, F. Schwander, E. Serre, and P. Tamain, in *38th EPS Conference on Plasma Physics*, Strasbourg, France (2011), p. 121.
[45]J. Loizu, P. Ricci, F. D. Halpern, and S. Jolliet, Phys. Plasmas **19**, 122307 (2012).
[46]A. Arakawa, J. Comput. Physics **1**, 119 (1966).
[47]Wolfram Research, Inc., *Mathematica* (Champaign, Illinois, 2010).