

## Development of Finite Element Field Solver in Gyrokinetic Toroidal Code

Hongying Feng<sup>1,2,3</sup>, Wenlu Zhang<sup>2,1,4,3\*</sup>, Zhihong Lin<sup>4,5</sup>,  
Xiaohe Zhufu<sup>6</sup>, Jin Xu<sup>6</sup>, Jintao Cao<sup>2,3</sup> and Ding Li<sup>2,3,1</sup>

<sup>1</sup> Department of Modern Physics, University of Science and Technology of China, Hefei, Anhui 230026, China.

<sup>2</sup> Beijing National Laboratory for Condensed Matter Physics and CAS Key Laboratory of Soft Matter Physics, Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China.

<sup>3</sup> University of Chinese Academy of Sciences, Beijing 100049, China.

<sup>4</sup> Department of Physics and Astronomy, University of California, Irvine, California 92697, USA.

<sup>5</sup> Fusion Simulation center, Peking University, Beijing 100871, China.

<sup>6</sup> Laboratory of Parallel Software and Computational Science, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China.

Received 24 June 2017; Accepted (in revised version) 17 August 2017

---

**Abstract.** A new finite element (FE) field solver has been implemented in the gyrokinetic toroidal code (GTC) in attempt to extend the simulation domain to magnetic axis and beyond the last closed flux surface, which will enhance the capability the GTC code since the original finite difference (FD) solver will lose its capability in such circumstances. A method of manufactured solution is employed in the unit fidelity test for the new FE field solver, which is then further verified through integrated tests with three typical physical cases for the comparison between the new FE field solver and the original finite difference field solver. The results by the newly implemented FE field solver are in great accord with the original solver.

**AMS subject classifications:** 68U20, 65C20, 35J05

**Key words:** Finite element method, finite difference method, Poisson equation, GTC.

---

## 1 Introduction

In the research of plasma physics, simulations have always served as an effective tool due to the complexity of theoretical analysis and the high cost of experiments. After

---

\*Corresponding author. Email addresses: wzhang@iphy.ac.cn (W. Zhang), tracyf@mail.ustc.edu.cn (H. Feng), zhihongl@uci.edu (Z. Lin), xiaohe@iscas.ac.cn (X. Zhufu), xu\_jin@iscas.ac.cn (J. Xu), jtcao@iphy.ac.cn (J. Cao), dli@cashq.ac.cn (D. Li)

several decades of fast development in the capability of high-performance-computing, it has become feasible to conduct massively parallel simulations to investigate more meaningful and complex physics processes using equilibrium and profiles close to realistic discharges in fusion plasmas. Along with the progress in computing power, a set of gyrokinetic theory [1–11] have been proposed and established to construct a set of simple theoretical and numerical models by eliminating the fine-scale gyro-phase dependence through gyroaveraging, which reduces the original phase space dimensionality from six to five. This not only assists the profound comprehension of low frequency physics, such as the anomalous transport that is critical for the magnetic fusion, but also facilitates the development and application of massively parallel simulation codes.

As one of the existing well benchmarked gyrokinetic codes, the gyrokinetic toroidal code (GTC) [10,12] is built upon the first-principles and adopts an efficient low-noise perturbative  $\delta f$  simulation method. The particle-in-cell (PIC) scheme is also utilized so that particles are treated with a Lagrangian scheme while fluid moments and field information are calculated with an Eulerian scheme. The capability of GTC has been extensively expanded and verified to deal with a wide range of physical problems, which include neoclassical and turbulence transport [13,14], energetic particle transport by microturbulence [15,16], Alfvén eigenmodes [17–20], radio frequency heating [21], static magnetic island [22] and current-driven instabilities [23,24].

GTC employs the magnetic flux coordinate system  $(\psi, \theta, \zeta)$  [25], where  $\psi$  is the poloidal magnetic flux,  $\theta$  is the poloidal angle and  $\zeta$  is the toroidal angle. The introduction of such a system makes it convenient to decompose a vector into components parallel and perpendicular to the direction of the magnetic field, and to separate the rapid particle motion along the magnetic field lines from the slow motion across the lines, which promotes the simplicity in theory analysis and efficiency in numerical simulation. Originally, GTC focused on the physics of microturbulence and the transport in the core area that are generally located in between a belt region away from the magnetic axis and Scrape-Off layer (SOL). When global modes and instabilities, such as kink mode and tearing mode, are considered, the on-axis region cannot be ignored and starts playing an important role. Meanwhile, the physics in the SOL region, a subregion of the halo, is the key for a holistic understanding to the plasma exhaust problem to ensure that the material surfaces will survive harsh plasma conditions and not interfere with core plasma in a magnetic fusion reactor. In order to extend the physics capabilities to the on-axis and SOL regions while improving numerical properties and avoiding the singularities naturally held by the concentric curvilinear coordinate systems, the finite element (FE) method is introduced and implemented in GTC [26] to replace the current on-axis solution which is simply an extrapolation of the gyrokinetic Poisson equation solution [23] to the magnetic axis.

The outline of this paper is as follows. In Section 2, equilibrium data setting and the particle-field interaction loop are introduced together with the related computing meshes. In Section 3, a brief introduction to the original finite difference (FD) Poisson solver is presented and the discrete form of the FD Poisson equation is demonstrated. In Section 4, the implementation of the finite element solver is introduced. In Section 5, the

accuracy of the FE solver is first verified and then a few typical cases are carried out in comparison to the old solver. In Section 6, a discussion is given.

## 2 Computing meshes in GTC

GTC, which is originally written in FORTRAN90, is one of the most widely used fusion codes in the magnetically confined fusion simulation community. GTC is the key production code for many multi-institutional collaborative research centers, projects and programs sponsored by, for example but not limited to, U.S. Department of Energy (DOE) Scientific Discovery through Advanced Computing (SciDAC) project and the National Special Research Program of China for ITER (International Thermonuclear Experimental Reactor). GTC has been growing from a single-developer and specific-problem code to a prominent code through continuously extending its capabilities for new physics by many independent users and contributors around the world. GTC also employs a multi-level hybrid programming in MPI, OpenMP and OpenACC/CUDA so that it can take advantage of the hardware hierarchy of modern massively paralleled computers and reach a scaling up to millions of conventional CPU cores and heterogeneous accelerating devices such as NVIDIA GPU and Intel Xeon Phi chips.

As a gyrokinetic particle-in-cell [27, 28] code, GTC tracks individual charged marker particles in a Lagrangian frame in continuous phase-space [10, 11], whereas the moments of particle distribution of different species (thermal ion, thermal electron, fast ion, fast electron, etc.) are simultaneously computed on a stationary Eulerian field mesh. This field mesh is also used to interpolate the local electromagnetic fields at the marker particle positions in phase-space. The trajectories of charged marker particles are calculated by integrators of the equations of motion in the self-consistent electromagnetic fields computed on the field mesh. The number density and current density carried by each marker particle is then projected to the field mesh through interpolations. The moments of the distributions of species, such as number density, charge density and current density, are then calculated by accumulating the projected quantities of marker particles. The electromagnetic fields are then solved on mesh grids using proper combinations of Poisson equation, Ampere's law, Faraday's law and force-balance equations with finite difference methods [26] and finite element methods which will be discussed in this paper.

Prior to the main iteration loop, which is introduced in the previous paragraph, the equilibrium and profiles are set up on the field-aligned mesh in the initialization part of the code from either analytical profiles or the numerical data, taken from realist experimental discharges through Equilibrium Fitting (EFIT) [29] for example. In the latter case, the numerical data is transformed to a simple intermediate spline mesh, which is used to construct the field-aligned mesh in turn. As shown in Fig. 1, the spline mesh has a uniform grid size in the radial direction and the same angle in poloidal direction while the field-aligned mesh has the same length in radial direction and all field-aligned grids at different radial positions have similar arc lengths. Usually, the grid of these two meshes

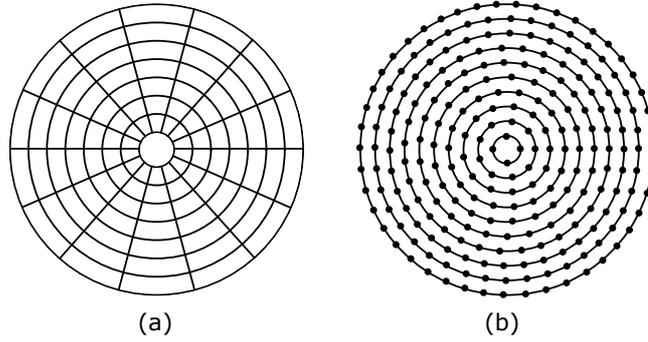


Figure 1: Two different meshes in GTC. Panel (a) is the spline mesh, panel (b) is the field-aligned mesh.

are totally different in both radial and poloidal directions. Thus a second-order polynomial interpolation based on the equilibrium data on the spline mesh is required for the calculation of the corresponding values on the field-aligned mesh, which are used in the main loop. For an arbitrary one dimensional variable  $f(x)$ , Eq. (2.1) gives the B-spline technique,

$$f(x_i) = f(x_\alpha) + f^{(1)}(x_\alpha)\Delta x + f^{(2)}(x_\alpha)(\Delta x)^2. \quad (2.1)$$

Here,  $x_i$  is the position of the  $i$ th grid point on field-aligned mesh and  $x_\alpha$  is the position of the  $\alpha$ th grid point on the spline mesh that is the nearest to but not larger than  $x_i$ .  $\Delta x$  stands for the gap between  $x_i$  and  $x_\alpha$ ,  $\Delta x = x_i - x_\alpha$ .  $f^{(1)}$  and  $f^{(2)}$  are coefficients related to the first and second order differential in  $x$  direction respectively, and are calculated through finite difference method on the spline mesh. If the dimensionality increases to two, adding the poloidal direction, then there become eight differential-related coefficients. For 3D problems, the number of interpolating coefficients increases to twenty-six.

After equilibrium profiles are determined on the field-aligned mesh, initial particles are randomly sampled in the gyrocenter coordinate system. Then the main loop of PIC scheme begins. To illuminate the procedure, we choose a small area of the field-aligned mesh for assistance, shown in Fig. 2. In gyrokinetic systems, particle motion is replaced by gyrocenter motion through gyroaveraging. The motions of gyrocenters are governed by weight equations [10]. For gyrocenter  $f_{gy}$ , the gyroaveraging process is carried out by four points of  $f_a$ ,  $f_b$ ,  $f_c$ , and  $f_d$  [2]. The values at these averaging points are calculated via interpolation of surrounding grids. For example,  $f_c$  is calculated through  $c_1$ ,  $c_2$ ,  $c_3$ , and  $c_4$ . Given all the gyroaveraged drive terms caused by field, gyrocenter  $f_{gy}$  is then pushed, resulting in renewed information such as position, velocity, and distribution function. Then the updated gyrocenter  $f_{gy}$  information is distributed back onto the field-aligned meshes via the same interpolation procedure. The total distributed information is a summation of all gyrocenters. With accumulated moments of gyrocenter distribution on the mesh, the perturbed field information is evolved in turn, demonstrated by field equations, which together with motion equations and weight equations of particles, complete the full loop of particle-field interactions.

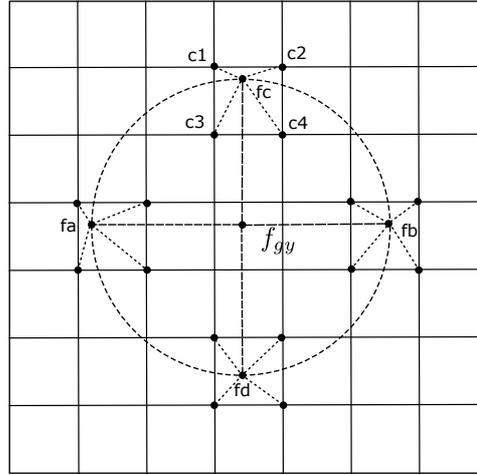


Figure 2: The PIC and gyroaveraging technique in GTC code.

This whole loop is performed in a field-aligned coordinate, with a finite difference method adopted for solving the field equation, which will be further discussed in the following section. On the contrast, the newly implemented FE solver creates a branch of the loop for FD solving routine of the field equation. The calculation in FE solver is still on the field-aligned mesh, but mapping it to a new Cartesian coordinate  $(X, Z)$ , where  $X$  stands for distance from the considered point to the cross-section axis, and  $Z$  is the position in the vertical  $Z$  direction, automatically removes the singularity problem. Next, the perturbed field solution on the mesh obtained by the FE solver, is used again in the field-aligned coordinate for the calculation of gyrocenter advancing, which eventually encloses the PIC circle.

### 3 Poisson equation and Laplacian operator

In this section, the gyrokinetic Poisson equations for both electrostatic and electromagnetic cases are listed. The different forms of Poisson equations arise from the application of a hybrid-kinetic model [30], the main idea of which is to divide electron response into a dominant adiabatic part and a high-order kinetic perturbation for better numerical performance. Later the finite difference method aimed at solving the Poisson equation in GTC under the magnetic flux coordinate [26] is briefly introduced.

#### 3.1 The form of Poisson equation

We start from the gyrokinetic Poisson equation [2],

$$\sum_{s \in gy} \frac{Z_s^2 n_{0s}}{T_s} (\phi - \tilde{\phi}_s) = \sum_{s \in gy} Z_s \tilde{n}_s - n_e,$$

where index  $s$  stands for particle species governed by gyrokinetics equations,  $Z_s$  stands for the charge number for species  $s$ ,  $\bar{n}_s$  stands for the gyrocenter density  $\bar{n}_s = \int d\mathbf{Z} \delta(\mathbf{R} + \boldsymbol{\rho} - \mathbf{x}) \delta f_s(\mathbf{Z})$ ,  $\tilde{\phi}_s$  stands for the second-gyro-averaged potential  $\tilde{\phi}_s = (1/n_{0s}) \int d\mathbf{Z} \delta(\mathbf{R} + \boldsymbol{\rho} - \mathbf{x}) \bar{\phi}(\mathbf{R}, \boldsymbol{\rho} f_{0s}(\mathbf{Z}))$  with  $\bar{\phi}(\mathbf{R}, \boldsymbol{\rho})$  being the gyroaveraged potential  $\bar{\phi}(\mathbf{R}, \boldsymbol{\rho}) = (1/2\pi) \int d\varphi d\mathbf{x} \delta(\mathbf{R} + \boldsymbol{\rho} - \mathbf{x}) \phi(\mathbf{x})$ , and  $n_e$  stands for the electron response which is expanded into an adiabatic part and a nonadiabatic part  $n_e = n_e^{(0)} + n_e^{(1)}$ . By using the Pade approximation, the representation of the second-gyroaveraged potential in Fourier space can be written as

$$\phi - \tilde{\phi} = (1 - \Gamma_0(k_{\perp}^2 \rho_s^2)) \phi \approx \frac{k_{\perp}^2 \rho_s^2}{1 + k_{\perp}^2 \rho_s^2} \phi.$$

After transforming it back to real space, we obtain

$$\frac{Z_i^2 e \rho_i^2}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} \nabla_{\perp}^2 \phi = - (1 - \rho_i^2 \nabla_{\perp}^2) \left( \sum_{s \in \text{gy}} \bar{n}_s - n_e \right).$$

Since the adiabatic electron response in the electrostatic scenario has the form  $n_e^{(0)} = en_{0e} \phi / T_e$ , thus it can be moved from the right side of the equation to the left side, resulting in a diagonal term in the Laplacian operator. This operation leads to different expression forms of Poisson equations for electrostatic and electromagnetic situations, as shown below,

$$\begin{aligned} \left( \frac{Z_i^2 e}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} + \frac{en_{0e}}{T_e} \right) \rho_i^2 \nabla_{\perp}^{2l} \phi &= \rho_i^2 \nabla_{\perp}^{2l} \sum_{s \in \text{gy}} \bar{n}_s - \left( \frac{1}{1+l} \right) \left( \sum_{s \in \text{gy}} \bar{n}_s - n_e^{(1)} \right), \\ \frac{Z_i^2 e}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} \rho_i^2 \nabla_{\perp}^2 \phi &= \rho_i^2 \nabla_{\perp}^2 \left( \sum_{s \in \text{gy}} \bar{n}_s - n_e \right) - \left( \sum_{s \in \text{gy}} \bar{n}_s - n_e \right), \end{aligned}$$

where  $\nabla_{\perp}^{2l}$  is the result of adding the diagonal term related to the adiabatic electron response to  $\nabla_{\perp}^2$ , and  $l$  is defined as

$$l = \frac{en_{0e}}{T_e} \left/ \left( \frac{Z_i^2 e}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} \right) \right.$$

Note that the solution of partial differential equation which has the form  $A \nabla^2 X = \nabla^2 B + C$  can be interpreted as the sum of two parts  $X = X_1 + X_2$ , where  $X_1 = B/A$  and  $X_2$  is the root of the equation  $A \nabla^2 X = C$ , so the equations that need to be solved actually take the following forms,

$$\begin{aligned} - \left( \frac{Z_i^2 e}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} + \frac{en_{0e}}{T_e} \right) \rho_i^2 \nabla_{\perp}^{2l} \phi_2 &= \left( \frac{1}{1+l} \right) \left( \sum_{s \in \text{gy}} \bar{n}_s - n_e^{(1)} \right), \\ \text{with } \phi_1 &= \sum_{s \in \text{gy}} \bar{n}_s \left/ \left( \frac{Z_i^2 e}{T_i} \sum_{s \in \text{gy}} \frac{n_{0s} m_s}{m_i} + \frac{en_{0e}}{T_e} \right) \right., \end{aligned} \quad (3.1)$$

$$\begin{aligned}
& -\frac{Z_i^2 e}{T_i} \sum_{s \in gy} \frac{n_{0s} m_s}{m_i} \rho_i^2 \nabla_{\perp}^2 \phi_2 = \left( \sum_{s \in gy} \bar{n}_s - n_e \right), \\
& \text{with } \phi_1 = \left( \sum_{s \in gy} \bar{n}_s - n_e \right) \bigg/ \frac{Z_i^2 e}{T_i} \sum_{s \in gy} \frac{n_{0s} m_s}{m_i}.
\end{aligned} \tag{3.2}$$

Here Eq. (3.1) is for the electrostatic case while Eq. (3.2) is for the electromagnetic case.

### 3.2 Laplacian operator solver in GTC

A finite difference method is employed for solving the partial differential field equations described by Eq. (3.1) and Eq. (3.2). So in this section, the discretization technique for the Laplacian operator under the field-aligned coordinate [26] is demonstrated. Generally, the Laplacian operator can be expressed as

$$\nabla^2 f = \frac{1}{J} \frac{\partial}{\partial \bar{\zeta}^{\alpha}} \left( J \nabla \bar{\zeta}^{\alpha} \cdot \nabla \bar{\zeta}^{\beta} \frac{\partial}{\partial \bar{\zeta}^{\beta}} f \right),$$

where  $\alpha$  (or  $\beta$ ) denotes an arbitrary direction of the three basis vector.  $J$  is the Jacobian. Note that the Einstein's summation convention is spontaneously applied here and after. It is convenient to define a matrix tensor  $g^{\bar{\zeta}^{\alpha} \bar{\zeta}^{\beta}} \equiv \nabla \bar{\zeta}^{\alpha} \cdot \nabla \bar{\zeta}^{\beta}$ . Then we link the arbitrary curvilinear coordinate  $(\bar{\zeta}^1, \bar{\zeta}^2, \bar{\zeta}^3)$  to the magnetic flux coordinate  $(\psi, \theta, \zeta)$ . For simulation efficiency purpose [31], a new field-aligned coordinate  $(\psi, \theta_0, \zeta_0)$  is introduced by applying the following transformation,  $\theta_0 = \theta - \zeta/q$ ,  $\zeta_0 = \zeta$ . Considering the gyrokinetic ordering  $k_{\perp} \gg k_{\parallel}$ , the perpendicular Laplacian operator can finally be expressed as follows,

$$\begin{aligned}
\nabla_{\perp}^2 f &= g^{\psi\psi} \frac{\partial^2 f}{\partial \psi^2} + 2g^{\psi\theta} \frac{\partial^2 f}{\partial \psi \partial \theta_0} + \left( g^{\theta\theta} + g^{\zeta\zeta}/q^2 \right) \frac{\partial^2 f}{\partial \theta_0^2} + \frac{1}{J} \left( \frac{\partial J g^{\psi\psi}}{\partial \psi} + \frac{\partial J g^{\psi\theta}}{\partial \theta_0} \right) \frac{\partial f}{\partial \psi} \\
&+ \frac{1}{J} \left( \frac{\partial J g^{\psi\theta}}{\partial \psi} + \frac{\partial J g^{\theta\theta}}{\partial \theta_0} \right) \frac{\partial f}{\partial \theta_0}.
\end{aligned} \tag{3.3}$$

From (3.3), it is straightforward to recognize that the following efforts should be laid in the discretization of the operators  $\partial/\partial\psi$ ,  $\partial/\partial\theta_0$ ,  $\partial^2/\partial\psi^2$ ,  $\partial^2/\partial\theta_0^2$ , and  $\partial^2/\partial\psi\partial\theta_0$  in the poloidal cross-section, as shown in Fig. 3. Here an 11-point interpolation method is used. Taking point 6 in Fig. 3(b) for an example,  $\theta_0$  differential is calculated by point 5 and 7 on the same flux surface  $i$  while  $\psi$  differential is calculated by point  $\alpha$  on the inner surface  $i-1$  and  $\beta$  on the outer surface  $i+1$ . However,  $\alpha$  and  $\beta$  are usually not the mesh grid, so that the value at  $\alpha$  is obtained via the interpolation of point 1, 2, 3, and 4, while  $\beta$  from point 8, 9, 10 and 11.

## 4 Implementation of the FE solver

Apart from the finite difference method mentioned above, finite element method is another numerical technique to obtain a solution for a partial differential problem, which

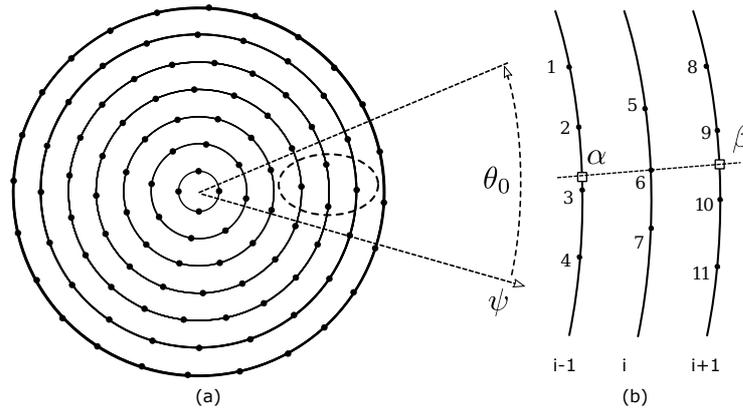


Figure 3: Discretization method in the poloidal cross-section.

originates from the variation method. In our FE scheme [32–34], we start from the equation with the form

$$\nabla \cdot [f_a(\mathbf{x}) \nabla u(\mathbf{x})] - f_b(\mathbf{x})u(\mathbf{x}) = f_c(\mathbf{x}) \tag{4.1}$$

so that it covers the equations ready to be solved, described by (3.1) and (3.2), as two special cases. Multiplied by a testing function  $\varphi(\mathbf{x})$ , and integrated over the domain  $\Omega$  with the Dirichlet boundary condition, (4.1) can be rewritten as

$$\int_{\Omega} \nabla \varphi(\mathbf{x}) \cdot [f_a(\mathbf{x}) \nabla u(\mathbf{x})] ds + \int_{\Omega} \varphi(\mathbf{x}) [f_b(\mathbf{x})u(\mathbf{x})] ds = - \int_{\Omega} \varphi(\mathbf{x}) f_c(\mathbf{x}) ds. \tag{4.2}$$

Consider a standardized triangle element  $D^e(r,s) = \{(r,s) | -1 \leq r, s \leq 1; r+s \leq 0\}$ . Super-script  $e$  indicates the sequential index of the triangle and  $(r,s)$  is the position of a point within the element according to the following mapping,

$$\mathbf{x} = -\frac{r+s}{2} \mathbf{v}^1 + \frac{1+r}{2} \mathbf{v}^2 + \frac{1+s}{2} \mathbf{v}^3,$$

where  $\mathbf{x}$  is the physical position of the point,  $\mathbf{v}^1, \mathbf{v}^2, \mathbf{v}^3$  are the coordinates of the vertices of the element. An arbitrary variant  $u(r,s)$  can be expanded to a linear combination of polynomials  $\psi_{ij} \in D^p = \{r^i s^j | 0 \leq i+j \leq p\}$ ,

$$u(r,s) = \sum_{0 \leq j+k < p} \hat{u}_{jk} \psi_{jk}(r,s), \text{ with } \psi_{ij}(r,s) = \sqrt{\frac{(2i+1)(i+j+1)}{2}} P_i^{0,0} \left(-\frac{2(r+s)}{1-s}\right) P_j^{2i+1,0}(s),$$

where  $P_n^{\alpha,\beta}(x)$  is the  $n$ th-order Jacobi Polynomial. Note that for any pair of  $(i,j)$ , it is convenient to use one subscript  $k$  to rewrite the basis function  $\phi_k = \psi_{ij}$ . Define the Vandermonde matrix  $V_{ij} = \phi_j(r_i, s_i)$  for the  $i$ th node. If the matrix is well conditioned, a function, whose values are known at nodes, can be uniquely expressed as

$$\hat{u}_i = \sum_{1 \leq j < N} V_{ij}^{-1} u(r_j, s_j), \text{ with } N = (p+1)(p+2)/2.$$

Subsequently, we can obtain

$$u(\mathbf{x}) = \sum_{k=0}^N \hat{u}_k \phi_k(\mathbf{x}) = \sum_{k=0}^N \phi_k(\mathbf{x}) \sum_{j=0}^N V_{jk}^{-1} u(r_j, s_j).$$

By adopting the polynomial representations of the test function  $\varphi(\mathbf{x})$  and the targeted function  $u(\mathbf{x})$ , (4.2) can be simplified to the form

$$\mathbf{A} \cdot \mathbf{u} = \mathbf{b}, \tag{4.3}$$

$$\text{with } \mathbf{A} = \int_{\Omega} \nabla \varphi(\mathbf{x}) \cdot [f_a(\mathbf{x}) \nabla \phi(\mathbf{x})] ds + \int_{\Omega} \varphi(\mathbf{x}) [f_b(\mathbf{x}) \phi(\mathbf{x})] ds,$$

$$\mathbf{b} = - \int_{\Omega} \varphi(\mathbf{x}) f_c(\mathbf{x}) ds,$$

$$\mathbf{u} = \begin{pmatrix} \hat{u}_0 \\ \vdots \\ \hat{u}_N \end{pmatrix}.$$

Furthermore, integral and differential in (4.3) can also be discretized on grid base,

$$(f, g) = J \sum_{i=0}^N \sum_{j=0}^N \sum_{k=0}^N f_i (V_{ji}^{-1} V_{jk}^{-1}) g_k,$$

$$\frac{\partial f}{\partial x}(r_i, s_i) = \sum_{j=0}^N (r_x D_{ij}^r + s_x D_{ij}^s) f(r_j, s_j),$$

$$\frac{\partial f}{\partial y}(r_i, s_i) = \sum_{j=0}^N (r_y D_{ij}^r + s_y D_{ij}^s) f(r_j, s_j),$$

where  $D_{ij}^r$  and  $D_{ij}^s$  are defined as  $D_{ij}^r = \sum_{k=1}^N \partial \phi_k(r, s) / \partial r V_{kj}^{-1}$  and  $D_{ij}^s = \sum_{k=1}^N \partial \phi_k(r, s) / \partial s V_{kj}^{-1}$  with  $V_{kj}^{-1}$  standing for the inverse Vandemonde matrix.  $J$  is the Jacobian of the transformation from normal Cartesian coordinate  $\mathbf{x}$  to the triangle coordinate  $(r, s)$ . Any choice of a straight-lined triangle element will result in a Jacobian independent of coordinate  $(r, s)$ , as employed in this paper. In real calculations, test function  $\varphi$  is chosen to be the basis function  $\phi$ .

In order to implement the FE solver into the GTC code, an interface between the solver and GTC code should be constructed, which contains two major parts, the physical coordinates of each grid and the coefficients in Eq. (4.1). Therefore at the very beginning, we need to acquire the positions of grid points. Here a simple transformation from the field-aligned coordinate to the Cartesian coordinate is carried out,

$$x = r \cos(\theta_0), \quad z = r \sin(\theta_0), \tag{4.4}$$

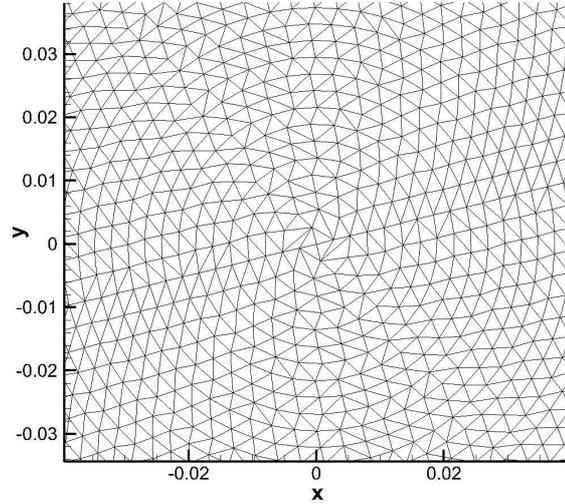


Figure 4: Mesh generated around the magnetic axis.

where  $r$  is the radial position of each point and can be obtained through its relation with the toroidal magnetic flux  $\psi_t$  that reads  $\psi_t = B_0 r^2 / 2$  by definition, in addition to the property  $d\psi/d\psi_t = 1/q(\psi)$ . Given grid points information acquired above, triangle elements are generated and labeled with sequential indexes for calculation. Fig. 4 gives a clear illumination of the mesh generated around the magnetic axis area. Next, the determination of the form of the coefficients  $f_a(\mathbf{x})$ ,  $f_b(\mathbf{x})$ , and  $f_c(\mathbf{x})$  emerged in (4.1) is considered. By simply comparing the forms of (4.1) with (3.1) and (3.2), we can reach that,

$$\left\{ \begin{array}{l} f_a = - \left( \frac{Z_i^2 e}{T_i} \sum_{s \in gy} \frac{n_{0s} m_s}{m_i} + \frac{en_{0e}}{T_e} \right) \rho_i^2, \\ f_b = - \frac{en_{0e}}{T_e}, \\ f_c = \left( \frac{1}{1+l} \right) \left( \sum_{s \in gy} \bar{n}_s - n_e^{(1)} \right), \end{array} \right. \quad \text{or} \quad \left\{ \begin{array}{l} f_a = - \frac{Z_i^2 e}{T_i} \sum_{s \in gy} \frac{n_{0s} m_s}{m_i} \rho_i^2, \\ f_b = 0, \\ f_c = \left( \sum_{s \in gy} \bar{n}_s - n_e \right), \end{array} \right. \quad (4.5)$$

where the left equations are for electrostatic cases and the right for electromagnetic cases. By now, we have successfully built the interface between the solver and the GTC source code through (4.4) and (4.5).

## 5 Verification of the FE solver

The verification process of the newly implemented FE field solver is carried out through two steps. First a test function  $n(r, \theta_0)$  is carefully manufactured so that the equation

$$\nabla^2 \phi = n, \quad (5.1)$$

with a zero boundary condition at both inner and outer boundaries, holds a simple analytic solution, to which the results calculated by both the FE solver and original solver can be compared. Secondly, three typical cases that consist of ion temperature gradient (ITG) mode, kink mode and fast-electron driven beta-induced alfvén eigenmode (e-BAE) are chosen to be simulated via the FE solver and the original solver as well for benchmarking.

### 5.1 Manufactured function verification

Choose  $\phi = \sin[2\pi(r-r_0)/(r_1-r_0)]\cos(\theta_0)$ , so that the source term on the right side of (5.1) can be set as  $n = \nabla_{\perp}^2(\phi)$ , with  $\nabla_{\perp}^2 = \partial^2/\partial r^2 + \partial/r\partial r + (1/r^2 + 1/q^2 X^2)\partial^2/\partial\theta_0^2$  and  $X = r\cos(\theta)/R_0$ . Here, only the cylinder geometry is taken into consideration for simplicity. With zero boundary condition, the analytic solution is  $\phi_{analytic} = \sin[2\pi(r-r_0)/(r_1-r_0)]\cos(\theta_0)$ . Then compare it with the numerical solutions of both the original solver and the FE solver. As Fig. 5 shows, panel (a) and (c) are the result of the FE solver and the original solver, while panel (b) and (d) stand for the difference from the analytic solution for each solver. Since the calculation is of single precision, where results with an error level of  $10^{-4}$  can be regarded as satisfactorily accurate, it is safe to conclude that both the FE solver and original solver can give a solution of a very good accuracy, with a 0.02% error for FE solver and a 0.08% error for the original solver. The radial profiles of three solutions at  $\theta_0 = 0$  are also displayed in Fig. 6, showing great agreement among the three.

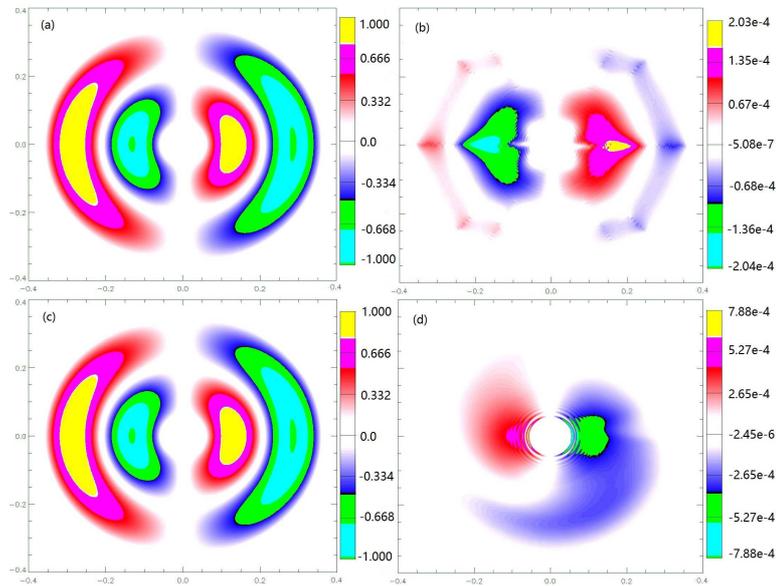


Figure 5: (Color online) Contour of the results of original solver and FE solver. Panel (a) and (c) are the results of FE solver and original solver respectively. Panel (b) and (d) denote the difference between the analytic solution and the FE, original solution.

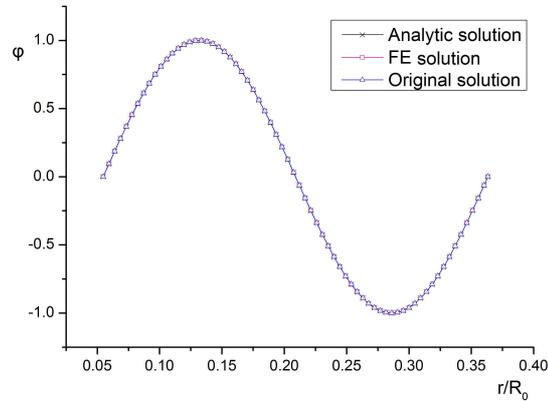


Figure 6: (Color online) Radial solution comparison between analytic, FE solver and original solver results.

## 5.2 Physics case verification

In this section, three typical physical cases are selected to cover a wide range of physical scenarios for the solver verification. For instance, linear ITG mode is an electrostatic instability which contains the contribution of gyrokinetic ions and adiabatic electrons, thus falls into the situation described by (3.1). Linear kink mode is an electromagnetic magnetohydrodynamics (MHD) instability that expands its effective domain to the very center near the magnetic axis where (3.2) is applicable. Meanwhile linear e-BAE is an electromagnetic instability which consists of a rather complex combination of different species, gyrokinetic thermal ions, fluid adiabatic thermal electrons, and drift-kinetic fast electrons. Different cases indicate different choices and combinations of the coefficients listed in (4.1). To investigate the results produced by each solver, comparisons are carried out through different perspectives of poloidal cross-section mode structure and spectrum, growth rate, and frequency. First we consider the mode structure and spectrum. For all comparisons from Figs. 7 to 9, upper half panels stand for FE results while the rest half are original solver results. Since the ITG mode is electrostatic, only perturbed potential is shown, while the rest two cases contain both perturbed potential and parallel vector potential. The upper panel of a spectrum diagram, take Fig. 7(b1) for an instance, gives the poloidal spectrum, and the bottom panel stands for the parallel spectrum. It is obvious that the outcome of the two approaches is nearly identical. Following that, calculations of growth rate and frequency of each case are carried out as well for further comparison. Results are displayed in Table 1, which also show a high agreement between the two solvers with errors no more than 0.7%. Note that all the results are normalized by the ion gyro-frequency of each case.

From both the manufactured test function method and selected physical case verifications, we can draw the conclusion that the newly added FE solver in GTC code can

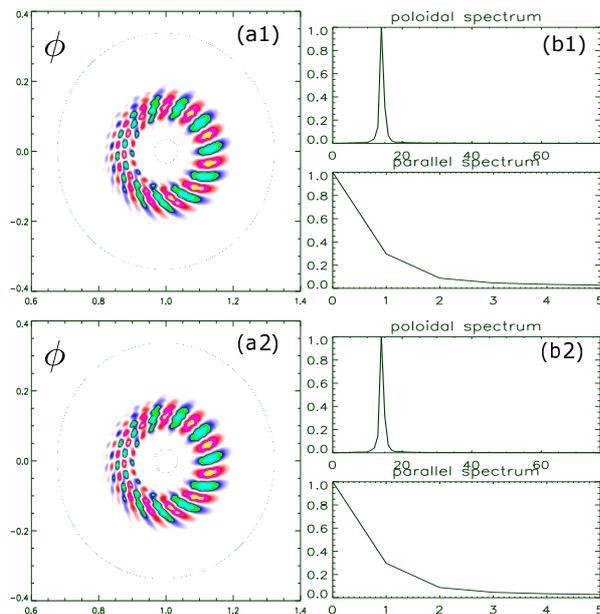


Figure 7: (Color online) Comparison of mode structure and spectrum calculated by different solvers for ITG.

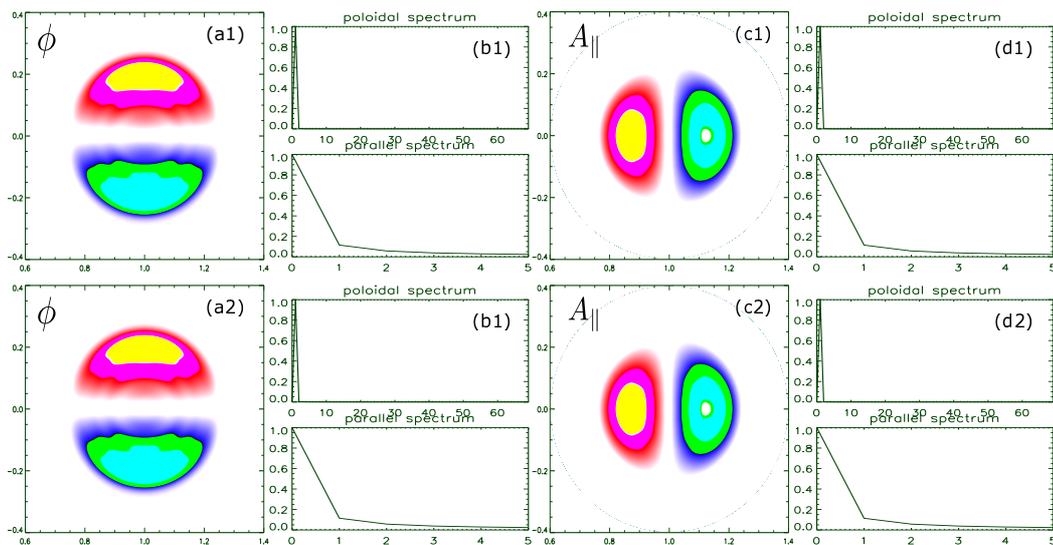


Figure 8: (Color online) Comparison of mode structure and spectrum calculated by different solvers for kink.

reach a very satisfying extent of accuracy, even slightly better than the original solver, indicated by the nearly twice smaller difference from the analytic solution in the manufactured test function verification. Furthermore, by comparing the computing time of all physics cases solved by the original solver and the new FE solver, we've found that for

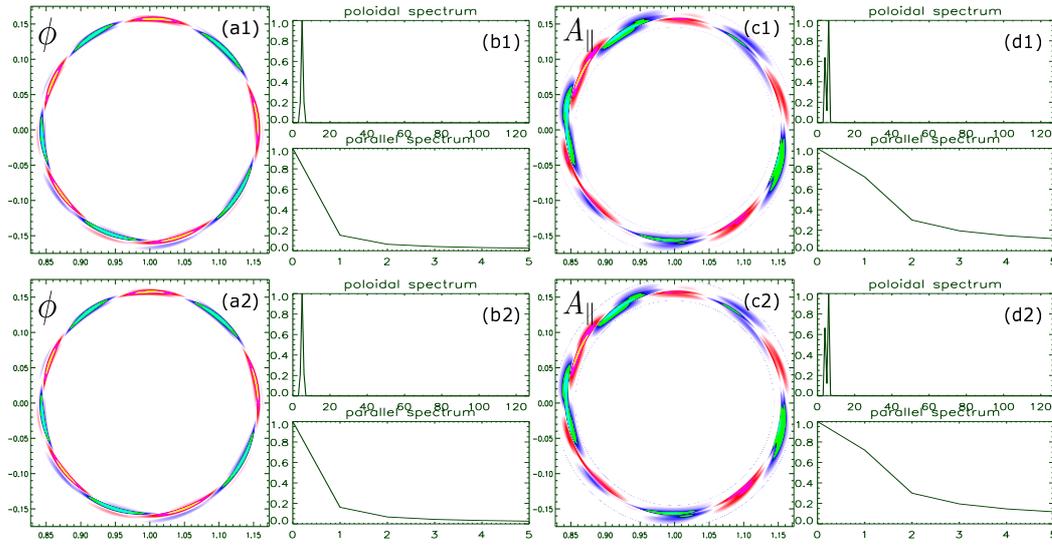


Figure 9: (Color online) Comparison of mode structure and spectrum calculated by different solvers for e-BAE.

Table 1: Growth rate and frequency benchmarks of different physical cases. Positive frequency means the wave propagates in the direction of electron diamagnetic drift while the negative frequency means the wave propagates in the direction of ion diamagnetic drift.

	ITG		kink mode		e-BAE	
	FE	Original	FE	Original	FE	Original
Growth rate	4.11	4.14	6.79	6.77	3.57	3.60
Frequency	-5.48	-5.48	0	0	19.7	19.7

the same physics case, the total time as well as the time spent in the solver subroutine is nearly the same when using different solver, which indicates that the FE solver has the same efficiency as the old solver.

## 6 Conclusion

Starting from the Poisson equation, we have derived the equivalent form of the equations that are ready to be solved. Next we have also introduced a new finite element method. And through a manufactured test function and physical case verification, the accuracy and efficiency of the newly added solver can be claimed at least as acceptable as the original finite difference method. Finally and more importantly, we believe that, when it comes to the problems where our future interest lies, such as the problems around the edge and divertor areas, and the original finite difference method loses its capability, our newly added FE solver will reveal its advantages.

## Acknowledgments

This work was supported by the China National Magnetic Confinement Fusion Science Program under Grant Nos. 2013GB111003 and 2013GB112011, the National Natural Science Foundation of China under Grant Nos. 11675257 and 11675256, the Strategic Priority Research Program of Chinese Academy of Sciences under Grant No. XDB16010300, the Key Research Program of Frontier Science of Chinese Academy of Sciences under Grant No. QYZDJ-SSW-SYS016, and the External Cooperation Program of Chinese Academy of Sciences under Grant No. 112111KYSB20160039. This research used resources of the National Supercomputer Center in Tianjin (NSCC-TJ), the Oak Ridge Leadership Computing Facility (OLCF) at Oak Ridge National Laboratory and the National Energy Research Scientific Computing Center (NERSC).

## References

- [1] W. W. Lee. Gyrokinetic approach in particle simulation. *The Physics of Fluids*, 26(2):556–562, 1983.
- [2] W.W Lee. Gyrokinetic particle simulation model. *Journal of Computational Physics*, 72(1):243–269, 1987.
- [3] Robert G Littlejohn. Variational principles of guiding centre motion. *Journal of Plasma Physics*, 29(01):111–125, 1983.
- [4] AJa Brizard and TSb Hahm. Foundations of nonlinear gyrokinetic theory. *Reviews of modern physics*, 79(2):421, 2007.
- [5] TS Hahm. Nonlinear gyrokinetic equations for tokamak microturbulence. *Physics of Fluids (1958-1988)*, 31(9):2670–2673, 1988.
- [6] EA Frieman and Liu Chen. Nonlinear gyrokinetic equations for low-frequency electromagnetic waves in general plasma equilibria. *Physics of Fluids (1958-1988)*, 25(3):502–508, 1982.
- [7] Andre Rogister and Ding Li. Kinetic and transport theories of turbulent, axisymmetric, low-collisionality plasmas and turbulence constraints. *Physics of Fluids B: Plasma Physics (1989-1993)*, 4(4):804–830, 1992.
- [8] Zhihong Lin and Liu Chen. A fluid–kinetic hybrid electron model for electromagnetic simulations. *Physics of Plasmas (1994-present)*, 8(5):1447–1450, 2001.
- [9] Yu Lin, Xueyi Wang, Zhihong Lin, and Liu Chen. A gyrokinetic electron and fully kinetic ion plasma simulation model. *Plasma physics and controlled fusion*, 47(4):657, 2005.
- [10] I. Holod, W. L. Zhang, Y. Xiao, and Z. Lin. Electromagnetic formulation of global gyrokinetic particle simulation in toroidal geometry. *Physics of Plasmas*, 16(12):122307, 2009.
- [11] Pengfei Liu, Zhang Wenlu, Dong Chao, Lin Jingbo, Lin Zhihong, and Cao Jintao. A closed vlasov-maxwell simulation model for high-frequency nonlinear processes in plasma. *Submitted to Nuclear Fusion*, 2017.
- [12] Z. Lin, T. S. Hahm, W. W. Lee, W. M. Tang, and R. B. White. Turbulent transport reduction by zonal flows: Massively parallel simulations. *Science*, 281(5384):1835–1837, 1998.
- [13] Z. Lin, I. Holod, L. Chen, P. H. Diamond, T. S. Hahm, and S. Ethier. Wave-particle decorrelation and transport of anisotropic turbulence in collisionless plasmas. *Phys. Rev. Lett.*, 99:265003, Dec 2007.

- [14] Yong Xiao and Zhihong Lin. Turbulent transport of trapped-electron modes in collisionless plasmas. *Phys. Rev. Lett.*, 103:085004, Aug 2009.
- [15] Wenlu Zhang, Zhihong Lin, and Liu Chen. Transport of energetic particles by microturbulence in magnetized plasmas. *Phys. Rev. Lett.*, 101:095001, Aug 2008.
- [16] Wenlu Zhang, Viktor Decyk, Ihor Holod, Yong Xiao, Zhihong Lin, and Liu Chen. Scalings of energetic particle transport by ion temperature gradient microturbulence. *Physics of Plasmas*, 17(5):055902, 2010.
- [17] Wenlu Zhang, Ihor Holod, Zhihong Lin, and Yong Xiao. Global gyrokinetic particle simulation of toroidal alfvén eigenmodes excited by antenna and fast ions. *Physics of Plasmas*, 19(2):022507, 2012.
- [18] Chenxi Zhang, Wenlu Zhang, Zhihong Lin, and Ding Li. Comparison of toroidicity-induced alfvén eigenmodes and energetic particle modes by gyrokinetic particle simulations. *Physics of Plasmas*, 20(5):052501, 2013.
- [19] Zhixuan Wang, Zhihong Lin, Ihor Holod, W. W. Heidbrink, Benjamin Tobias, Michael Van Zeeland, and M. E. Austin. Radial localization of toroidicity-induced alfvén eigenmodes. *Phys. Rev. Lett.*, 111:145003, Oct 2013.
- [20] Junyi Cheng, Wenlu Zhang, Zhihong Lin, Ihor Holod, Ding Li, Yang Chen, and Jintao Cao. Gyrokinetic particle simulation of fast-electron driven beta-induced alfvén eigenmode. *Physics of Plasmas*, 23(5):052504, 2016.
- [21] A. Kuley, Z. Lin, J. Bao, X. S. Wei, Y. Xiao, W. Zhang, G. Y. Sun, and N. J. Fisch. Verification of nonlinear particle simulation of radio frequency waves in tokamak. *Physics of Plasmas*, 22(10):102515, 2015.
- [22] Jiang Peng, Lin Zhihong, Ihor Holod, and Xiao Chijie. The implementation of magnetic islands in gyrokinetic toroidal code. *Plasma Science and Technology*, 18(2):126, 2016.
- [23] J. McClenaghan, Z. Lin, I. Holod, W. Deng, and Z. Wang. Verification of gyrokinetic particle simulation of current-driven instability in fusion plasmas. i. internal kink mode. *Physics of Plasmas*, 21(12):122519, 2014.
- [24] Dongjian Liu, Wenlu Zhang, Joseph McClenaghan, Jiaqi Wang, and Zhihong Lin. Verification of gyrokinetic particle simulation of current-driven instability in fusion plasmas. ii. resistive tearing mode. *Physics of Plasmas*, 21(12):122520, 2014.
- [25] R. B. White and M. S. Chance. Hamiltonian guiding center drift orbit calculation for plasmas of arbitrary cross section. *The Physics of Fluids*, 27(10):2455–2467, 1984.
- [26] Yong Xiao, Ihor Holod, Zhixuan Wang, Zhihong Lin, and Taige Zhang. Gyrokinetic particle simulation of microturbulence for general magnetic geometry and experimental profiles. *Physics of Plasmas*, 22(2):022516, 2015.
- [27] John M Dawson. Particle simulation of plasmas. *Reviews of modern physics*, 55(2):403, 1983.
- [28] Charles K Birdsall and A Bruce Langdon. *Plasma physics via computer simulation*. CRC Press, 2004.
- [29] L L Lao, H E St John, R D Stambaugh, A G Kellman, and W Pfeiffer. Reconstruction of current profile parameters and plasma shapes in tokamaks. *Nuclear Fusion*, 25(11):1611–1622, 1985.
- [30] Liu Chen, Zhihong Lin, and Roscoe White. On resonant heating below the cyclotron frequency. *Physics of Plasmas*, 8(11):4713–4716, 2001.
- [31] S Ethier, W M Tang, and Z Lin. Gyrokinetic particle-in-cell simulations of plasma microturbulence on advanced computing platforms. *Journal of Physics: Conference Series*, 16(1):1, 2005.
- [32] J. Xu, P. N. Ostroumov, B. Mustapha, and J. Nolen. Scalable direct vlasov solver with dis-

- continuous galerkin method on unstructured mesh. *SIAM Journal on Scientific Computing*, 32(6):3476–3494, 2010.
- [33] J Xu, B Mustapha, PN Ostroumov, and J Nolen. Solving 2d2v vlasov equation with high order spectral element method. *Communication in Computational Physics*, 8:159–184, 2010.
- [34] Jin Xu, Peter N Ostroumov, and Jerry Nolen. A parallel 3d poisson solver for space charge simulation in cylindrical coordinates. *Computer Physics Communications*, 178(4):290–300, 2008.