INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 21, Number 3, Pages 375–406

© 2024 Institute for Scientific
Computing and Information
doi: 10.4208/ijnam2024-1015

CFIRM: AN INTEGRATED CODE PACKAGE FOR THE LOW-TEMPERATURE PLASMA SIMULATION ON STRUCTURED GRIDS

JINWEI BAI, HONGTAO LIU, XIAOMING HE, WEI JIANG, AND YONG CAO*

Abstract. This paper presents a recently developed full kinetic particle simulation code package, which is a two-dimensional highly integrated and universal framework for low-temperature plasma simulation on both Cartesian and axisymmetric coordinate systems. This code package is named CFIRM, since it is designed based on the continuous Galerkin immersed-finite-element (IFE) particle-in-cell (PIC) model with the polynomial-preserving-recovery (PPR) technique and the Monte-Carlo-collision (MCC) method. Both the traditional and implicit PIC methods were implemented in the package. Incorporating the advantages of all these methods together, the CFIRM code can adopt explicit or implicit PIC schemes to track the motion trajectory of charged particles and deal with the collisions between plasma and neutral gas. Additionally, it can conveniently handle complex interface problems on structured grids. The CFRIM code has excellent versatility in low-temperature plasma simulation and can easily extend to various particle processing modules, such as the variable weights and adaptive particle management algorithms which were incorporated into this code to reduce the memory utilization rate. The implementation for the main algorithms and the overall simulation framework of the CFIRM code package are rigorously described in details. Several simulations of the benchmark cases are carried out to validate the reliability and accuracy of the CFIRM code. Moreover, two typical low-temperature plasma engineering problems are simulated, including a hall thruster and a capacitively coupled plasma reactor, which demonstrates the applicability of this code package.

Key words. Low-temperature plasma, particle-in-cell, immersed-finite-element, polynomial-preserving-recovery, Monte-Carlo-collision.

1. Introduction

Low-temperature plasma is a state of matter characterized by the electron temperature being significantly higher than the ion temperature. It is commonly generated by gas discharge at low pressures with the help of direct current, radio frequency, or microwave sources. Nowadays, low-temperature plasma technology has been widely used in many industries [1, 2], such as etching [3], electric propulsion [4], accelerators [5], and material surface modification [6]. Numerical simulations have been demonstrated to be an economical and powerful method for the research of low-temperature plasma. They can be used to reveal the basic physical mechanisms of plasma, assist in optimizing structural design, and significantly reduce research and development costs.

The commonly used simulation methods of the low-temperature plasma include fluid [7, 8], direct kinetic (Boltzmann) [9, 10], and particle-in-cell (PIC) models [11]. The fluid models solve the conservation equations of mass, momentum, and energy to obtain the macroscopic quantities of each physical parameter, like velocity, density, temperature, and so on. It always assumes that the velocity distribution function satisfies Maxwell's distribution. Although fluid models have been fully developed and have enough advantages in computational efficiency, their results will become less accurate once the thermodynamic equilibrium assumption is broken down. The direct kinetic models can obtain the time evolution of the distribution

Received by the editors on December 8, 2023 and, accepted on March 6, 2024.
2000 *Mathematics Subject Classification.* 35Q70, 65N30, 70-04.
*Corresponding author.

function by solving the Boltzmann or Vlasov equation, which provides the probability of particles in the given state of physical space and velocity space. However, the direct kinetic models have much higher computational cost, especially in the simulation of three-dimensional problems. On the other hand, the PIC method is a compromise between the fluid and direct kinetic models, which regards the plasma as a large number of particles and can capture the non-equilibrium phenomena of plasma by tracking the motion of particles. Hence the PIC models have become the most popular kinetic methods for low-temperature plasma simulations.

The classical PIC method was proposed by Birdsall et al [12]. It constantly updates the velocities and positions of particles through the 'leap-frog' scheme and updates the electric field by solving Poisson's equation. In order to simulate the collisions between the particles, the Monte Carlo collision (MCC) method was introduced in the PIC model [11], and then the PIC-MCC framework was widely used in low-temperature plasma simulation [13, 14, 15, 16]. Nevertheless, the traditional PIC method necessitates the use of sufficiently small spatial and temporal step sizes to precisely track the behaviors of the electrons. The step sizes of the PIC method should satisfy the following conditions, namely $\Delta x \leq \lambda_D$, $\Delta t \leq 2/\omega_{pe}$ and $\Delta x/\Delta t < v_{te}$, where Δx is the spatial step size, Δt is the temporal step size, λ_D is the Debye length, ω_{pe} is the electron plasma frequency, and v_{te} is the electron thermal velocity. This leads to a huge computational cost in the simulations with high plasma density and a large simulation domain. Therefore, the implicit PIC method [17, 18, 19, 20, 21, 22] was developed to reduce the cost of the traditional PIC algorithm. It can eliminate the limitations of step sizes by damping the high-frequency modes of the plasma. Although the high-frequency characteristic of the electron is lost when the step sizes are enlarged, most of the kinetic behaviors of the plasma are maintained. Thus, the implicit PIC model is an important method for large-scale plasma simulation. No matter which PIC scheme is adopted, the structured grids are the optimal option to improve the efficiency of particle localization, especially Cartesian meshes. However, the structured grids with traditional field solvers cannot accurately handle the problems that involve complex physical interfaces in the domain.

The immersed-finite-element (IFE) method [23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39] is a special finite element method, which was developed to solve the complex interface problems on structured grids. The meshes of the IFE method are independent of the interface, thus it has great advantages in dealing with complex and moving interfaces. The IFE method has been utilized as a field solver of the PIC model in recent years, which leads to the so called IFE-PIC method [40, 41, 42, 43, 44, 45, 46]. The IFE-PIC method has been applied to the simulations of low-temperature plasma problems, such as electric thrusters [47, 48, 49, 50, 51, 52, 53, 54, 55, 56] and lunar surface charging [57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67].

The polynomial-preserving-recovery (PPR) technique was proposed by Naga and Zhang [68]. This method uses the fitted numerical solutions to recover the gradient. Due to its superconvergence property, the PPR has received widespread attention and development [69, 70, 71, 72]. Thus, we utilized the PPR technology to recover the electric field for the IFE-PIC method.

The purpose of this paper is to develop a two-dimensional highly integrated continuous Galerkin immersed-finite-element particle-in-cell code package, named as CFIRM. In CFIRM, the electric field is obtained by the PPR method, and the collisions between plasma and neutral gas are simulated by the MCC method. The CFIRM code is designed for simulating low-temperature plasma in both Cartesian and axisymmetric coordinate systems. To enhance the code's versatility and computational efficiency, we have incorporated both traditional and implicit PIC models into the code. Additionally, we have introduced various numerical schemes into

the CFIRM code, including variable weights and an adaptive particle management algorithm. These additions aimed at controlling the total number of simulated particles and improving computational efficiency. As a result, the CFIRM code offers excellent versatility and convenient scalability for low-temperature plasma simulations.

In this study, we begin by presenting a thorough overview of the main algorithms and the overall simulation framework implemented in the code package. Then we provide a series of numerical benchmark experiments to validate the accuracy and stability of the CFIRM code. The obtained results exhibit a high level of consistency with the benchmark results. Finally, we demonstrate the simulation capability of the CFIRM code through two typical low-temperature plasma problems: one for the simulation of a hall thruster and the other one for a capacitively coupled plasma reactor. These results confirm the robust computational capabilities of the CFIRM code package for low-temperature plasma simulations.

The rest of the paper is organized as follows. In Section 2, we describe the main methods and their implementations of the CFRIM code, including the PIC, IFE, MCC, and PPR methods. In Section 3, we provide some numerical benchmark experiments to illustrate the accuracy and stability of CFRIM. In Section 4 the applications are simulated to show the applicability of CFIRM in low-temperature plasma simulation. Finally, we conclude with a brief summary in Section 5.

2. The main algorithms and implementations of CFIRM

In a simulated loop of the CFIRM code, the motion trajectory of the particle is tracked by the PIC method, while the IFE method is used to solve Poisson's equation to obtain the potential. Then the electric field is recovered by the P-PR technique. And the collisions between the particles are handled by the MCC method. The above simulation generally requires multiple cyclic iterations until it reaches a steady state. In this section, we will briefly introduce the main algorithms and demonstrate the implementations for the CFIRM package. It should be noted that the description in this section is based on the Cartesian coordinate system. For the differences in the axisymmetric coordinate system, we provide some explanations in the remarks.

2.1. Particle pushing algorithm. The motion of the charged particle is induced by the electric field $\mathbf{E}$ and magnetic field $\mathbf{B}$. We can obtain the trajectory of the particles by Newton's Second Law. In order to enhance the simulation performance of the code package, we have integrated two particle pushing algorithms in the CFRIM code, including the explicit and direct implicit PIC schemes.

For the explicit scheme, the velocity $\mathbf{v}$ and position $\mathbf{x}$ of the particles are calculated by the known electric field in each step. They are updated approximately by the following leap-frog scheme [12]:

$$m\frac{\mathbf{v}^{n+1/2}-\mathbf{v}^{n-1/2}}{\Delta t}=q(\mathbf{E}^n+\mathbf{v}^n\times\mathbf{B}^n), \tag{1}$$

$$\frac{\mathbf{x}^{n+1}-\mathbf{x}^n}{\Delta t}=\mathbf{v}^{n+1/2}, \tag{2}$$

where m and q are the mass and charge of the particle, Δt is the temporal step size, and n is the time level.

For the direct implicit scheme [17, 19, 20], the electric field in the next step depends on the charge density in the future. Thus, the particle pushing procedure is decomposed into two steps. One is the pre-push step:

$$\tilde{\mathbf{v}}^{n+1/2}=\mathbf{T}\cdot(\mathbf{v}^{n-1/2}+\frac{1}{2}\bar{\mathbf{a}}^{n-1}\Delta t+\mathbf{v}^{n-1/2}\times\mathbf{\Omega}^n), \tag{3}$$

$$\tilde{\mathbf{x}}^{n+1} = \mathbf{x}^n + \tilde{\mathbf{v}}^{n+1/2}\Delta t, \tag{4}$$

and the other one is the post-push step:

$$\mathbf{v}^{n+1/2} = \tilde{\mathbf{v}}^{n+1/2} + \mathbf{T} \cdot \frac{q\mathbf{E}^{n+1}(\tilde{\mathbf{x}}^{n+1})\Delta t}{2m}, \tag{5}$$

$$\mathbf{x}^{n+1} = \tilde{\mathbf{x}}^{n+1} + \mathbf{T} \cdot \frac{q\mathbf{E}^{n+1}(\tilde{\mathbf{x}}^{n+1})\Delta t^2}{2m}, \tag{6}$$

where $\bar{\mathbf{a}}^n = \frac{\mathbf{a}^{n+1}+\bar{\mathbf{a}}^{n-1}}{2}$, $\mathbf{a}^{n+1} = \frac{q\mathbf{E}^{n+1}(\tilde{\mathbf{x}}^{n+1})}{m}$, $\mathbf{\Omega} = \frac{q\mathbf{B}\Delta t}{2m}$, and $\mathbf{T}$ is the magnetic field rotation tensor.

Algorithm 1 shows the pseudocode of the particle pushing algorithm in the CFRIM code. The parameter *PushFlag* is used to determine the mode of particle pushing.

Algorithm 1: Pseudocode of the explicit and implicit PIC algorithms in the CFIRM code.

```
Read: PushFlag ;
N_tot ← the total number of the simulated particles ;
while i < N_tot do
    if PushFlag = 0 then                                   /* Explicit move */
        Δv(1:3) = 0.5 * q(i) * E(1:3) * dt/m(i) ;
        Ω(1:3) = 0.5 * q(i) * B(1:3) * dt/m(i) ;
        f = 2/(1 + Ω(1:3) · Ω(1:3)) ;
        v(1:3,i) = v(1:3,i) + Δv(1:3) ;
        S(1:3) = (v(1:3,i) + v(1:3,i) × Ω(1:3)) * f ;
        v(1:3,i) = v(1:3,i) + S(1:3) × Ω(1:3) + Δv(1:3) ;
        x(1:3,i) = x(1:3,i) + dt * v(1:3,i) ;
    else if PushFlag = 1 then                          /* Implicit pre-push */
        Ω(1:3) = 0.5 * q(i) * B(1:3) * dt/m(i) ;
        S(1:3) = v(1:3,i) + 0.5 * ā(1:3,i) * dt + v(1:3,i) × Ω(1:3) ;
        v(1:3,i) = T · S(1:3) ;
        x(1:3,i) = x(1:3,i) + dt * v(1:3,i) ;
    else if PushFlag = 2 then                         /* Implicit post-push */
        a(1:3,i) = q(i) * E(1:3)/m(i) ;
        ā(1:3,i) = 0.5 * (ā(1:3,i) + a(1:3,i)) ;
        S(1:3) = 0.5 * a(1:3,i) * dt ;
        v(1:3,i) = v(1:3,i) + T · S(1:3) ;
        x(1:3,i) = x(1:3,i) + dt * v(1:3,i) ;

end while
```

Remark 2.1. *For the axisymmetric coordinate system, the physical parameters are described in r, z, and θ directions. They should first be transformed into the Cartesian coordinate system. Subsequently, we can update the position and velocity of the particles by the above Algorithm 1. Finally, the updated results are transformed back into the axisymmetric coordinate system again.*

2.2. Charge deposition algorithm. In PIC models, the particles are independent, and they can move randomly anywhere in the domain. When the position information of particles changes, the influence of the charge carried by particles on each node will synchronously change. Accurate weighting is needed to interpolate the particle charges on the discrete mesh points. For example, the contribution of the particle P, which is located at position $\mathbf{x}_P$ with charge q_P, to the charge at

node $\mathbf{x}_{i,j}$ can be calculated by

$$q_{i,j} = q_P \frac{S(\mathbf{x}_{i+1,j+1}, \mathbf{x}_P)}{S(\mathbf{x}_{i,j}, \mathbf{x}_{i+1,j+1})}, \tag{7}$$

where $S(A, B)$ is the area of rectangle whose diagonal is $\overline{AB}$. Then, the total charge density ρ^{n+1} can be obtained by dividing the total charge of each node by the area.

However, when the immersed-finite-element (IFE) method is utilized as the field solver for the electric potential in a domain with an irregular interface, there will be some interface elements which are cut through by the interface. The vertices of these interface elements are distributed on the two sides of the interface. Then the discontinuity caused by the interface will cause the interpolation method using equation (7) to be inaccurate in these interface elements. Hence the interpolation will be not able to keep the conservation of the total charge and charge density. This problem was addressed in our previous work [73]. And the pseudocode of the charge deposition algorithm for the CFIRM code is presented in Algorithm 2.

Remark 2.2. *For the axisymmetric coordinate system, each mesh actually represents a circular ring with a rectangular cross-section, whose volume is* $V = \pi(R^2 - r^2)\Delta z$, *where* R *and* r *are the radial maximum and minimum of the mesh. It means that the volume of the mesh is different in the radial direction although using an identical spatial step size. Hence, the weighting defined by formula* (7) *is not suitable in an axisymmetric coordinate system. And it will be redefined as follows. Assuming the particle is located at* (z_P, r_P), *the weights are*

$$\begin{cases} w_{i,j} = \dfrac{(z_{i+1} - z_P)(r_{j+1}^2 - r_P^2)}{(z_{i+1} - z_i)(r_{j+1}^2 - r_j^2)}, \\ w_{i+1,j} = \dfrac{(z_p - z_i)(r_{j+1}^2 - r_P^2)}{(z_{i+1} - z_i)(r_{j+1}^2 - r_j^2)}, \\ w_{i,j+1} = \dfrac{(z_{i+1} - z_P)(r_P^2 - r_j^2)}{(z_{i+1} - z_i)(r_{j+1}^2 - r_j^2)}, \\ w_{i+1,j+1} = \dfrac{(z_P - z_i)(r_P^2 - r_j^2)}{(z_{i+1} - z_i)(r_{j+1}^2 - r_j^2)}. \end{cases} \tag{8}$$

The implementation is similar to Algorithm 2 with different weights, hence will be omitted here.

2.3. The immersed-finite-element method. In the explicit PIC model, the electric potential ϕ can be updated by solving the Poisson's equation:

$$-\nabla \cdot \varepsilon \nabla \phi^{n+1} = \rho^{n+1}, \tag{9}$$

where ε is the dielectric constant. On the other hand, the Poisson's equation for the implicit PIC model is modified as

$$-\nabla \cdot [\varepsilon(\boldsymbol{I} + \boldsymbol{\chi})] \nabla \phi^{n+1} = \tilde{\rho}^{n+1}, \tag{10}$$

$$\boldsymbol{\chi} = \sum_i^{N_s} \mathbf{T}_i \cdot \frac{q_i \tilde{\rho}_i^{n+1} \Delta t^2}{2\varepsilon m_i}, \tag{11}$$

where $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{\chi}$ is the effective susceptibility, i is the index of particle species and N_s is the number of particle species. $\mathbf{T}_i$ is the magnetic field rotation tensor, whose format is

$$\mathbf{T}_i = \frac{1}{1 + \Omega_i^2} \begin{bmatrix} 1 + \Omega_{i1}^2 & \Omega_{i1}\Omega_{i2} + \Omega_{i3} \\ \Omega_{i1}\Omega_{i2} - \Omega_{i3} & 1 + \Omega_{i2}^2 \end{bmatrix}, \tag{12}$$

Algorithm 2: Pseudocode of the charge deposit algorithm in the CFIRM code.

```
rho_s ← the charge density of each type of particle ;
rho ← the total charge density of plasma ;
while i_part < ntot do
    isp ← the index of particle species ;
    i,j ← INT(x/Δx), INT(y/Δy) ; dx,dy ← x/Δx-i, y/Δy-j ;
    A(1) ← (1-dx)*(1-dy) ; A(2) ← dx*(1-dy) ;
    A(3) ← (1-dx)*dy ; A(4) ← dx*dy ;
    rho_s(i ,j ,isp) = rho_s(i ,j ,isp) + A(1) ;
    rho_s(i+1,j ,isp) = rho_s(i+1,j ,isp) + A(2) ;
    rho_s(i ,j+1,isp) = rho_s(i ,j+1,isp) + A(3) ;
    rho_s(i+1,j+1,isp) = rho_s(i+1,j+1,isp) + A(4) ;
    N_inobj ← the total number inside the object of nodes (i,j), (i+1,j), (i,j+1),
    and(i+1,j+1) ;
    index(1:4) ← the index of which node is located inside the object ;
    if N_inobj = 1 then
        if index(1) = 1 then
            /* Assume node (i,j) is inside the object, and the others are
               similar.                                                    */
            A_total = A(2)+A(3)+A(4) ;
            rho_s(i+1,j ,isp) = rho_s(i+1,j ,isp) + rho_s(i ,j ,isp) *A(2)/A_total ;
            rho_s(i ,j+1,isp) = rho_s(i ,j+1,isp) + rho_s(i ,j ,isp) *A(3)/A_total ;
            rho_s(i+1,j+1,isp) = rho_s(i+1,j+1,isp) + rho_s(i ,j ,isp) *A(4)/A_total ;
            rho_s(i ,j ,isp) = 0 ;
        end if
    else if N_inobj = 2 then
        if index(1) = 1 AND index(2) = 1 then
            /* Assume nodes (i,j) and (i+1,j) are inside the object, and the
               others are similar.                                         */
            A_total = A(3)+A(4) ;
            rho_s(i ,j+1,isp) = rho_s(i ,j+1,isp) + (rho_s(i ,j ,isp) + rho_s(i+1,j ,isp))
            *A(3)/A_total ;
            rho_s(i+1,j+1,isp) = rho_s(i+1,j+1,isp) + (rho_s(i ,j ,isp) + rho_s(i+1,j ,isp))
            *A(4)/A_total ;
            rho_s(i ,j ,isp) = 0 ; rho_s(i+1,j ,isp) = 0 ;
        end if
    else if N_inobj = 3 then
        if index(1) = 1 AND index(2) = 1 AND index(3) = 1 then
            /* Assume nodes (i,j), (i+1,j) and (i,j+1) are inside the object, and
               the others are similar.                                     */
            rho_s(i+1,j+1,isp) = rho_s(i+1,j+1,isp) + rho_s(i ,j ,isp) + rho_s(i+1,j ,isp) +
            rho_s(i ,j+1,isp) ;
            rho_s(i ,j ,isp) = 0 ; rho_s(i+1,j ,isp) = 0 ; rho_s(i ,j+1,isp) = 0 ;
        end if

end while
for isp ← 1 to isp_tot do
    rho_s(:,:,isp) = rho_s(:,:,isp) * weight(isp) * q(isp) / area(:,:) ;
    rho(:,:) = rho(:,:) + rho_s(:,:,isp) ;
end for
```

where $(\Omega_{i1}, \Omega_{i2}, \Omega_{i3}) = \mathbf{\Omega}_i = \frac{q_i \mathbf{B} \Delta t}{2m_i}$. The boundary conditions on the Dirichlet boundary Γ_D and the Neumann boundary Γ_N are

$$\phi|_{\Gamma_D} = g(\mathbf{x}), \tag{13}$$

Algorithm 3: Pseudocode of obtaining the IFE basis functions.

E_type, x_D, y_D, x_E, $y_E \leftarrow$ the type of interface element and the coordinates of intersection points, and they are obtained in module 'Mesh_Objects_Intersection_info_2D' ;
$(x_i, y_i),\ i = 1, 2, 3, 4 \leftarrow$ the coordinates of the element nodes ;
$R = \varepsilon^-/\varepsilon^+ \leftarrow$ the ratio of the dielectric coefficient ;
Initialize the matrix: $A = zeros(8, 8)$, the vector: $b = zeros(8, 1)$;
if E_type $= 1$ **then** `/* The first type interface element in Fig. 2 */`
  $A(1,:) = [x_1, y_1, x_1y_1, 1, 0, 0, 0, 0]$;
  $A(2,:) = [0, 0, 0, 0, x_2, y_2, x_2y_2, 1]$;
  $A(3,:) = [0, 0, 0, 0, x_3, y_3, x_3y_3, 1]$;
  $A(4,:) = [0, 0, 0, 0, x_4, y_4, x_4y_4, 1]$;
  $A(5,:) = [x_D, y_D, x_Dy_D, 1, -x_D, -y_D, -x_Dy_D, -1]$;
  $A(6,:) = [x_E, y_E, x_Ey_E, 1, -x_E, -y_E, -x_Ey_E, -1]$;
  $A(7,:) = [0, 0, 0, 1, 0, 0, 0, -1]$;
  $A(8,:) = [R(y_D - y_E), R(x_E - x_D), R\frac{(y_D^2 - y_E^2) + (x_E^2 - x_D^2)}{2}, 0, -(y_D - y_E), -(x_E - x_D), -\frac{(y_D^2 - y_E^2) + (x_E^2 - x_D^2)}{2}, 0]$;
else if E_type $= 2$ **then** `/* The second type interface element in Fig. 2 */`
  $A(1,:) = [x_1, y_1, x_1y_1, 1, 0, 0, 0, 0]$;
  $A(2,:) = [0, 0, 0, 0, x_2, y_2, x_2y_2, 1]$;
  $A(3,:) = [0, 0, 0, 0, x_3, y_3, x_3y_3, 1]$;
  $A(4,:) = [x_4, y_4, x_4y_4, 1, 0, 0, 0, 0]$;
  $A(5,:) = [x_D, y_D, x_Dy_D, 1, -x_D, -y_D, -x_Dy_D, -1]$;
  $A(6,:) = [x_E, y_E, x_Ey_E, 1, -x_E, -y_E, -x_Ey_E, -1]$;
  $A(7,:) = [0, 0, 0, 1, 0, 0, 0, -1]$;
  $A(8,:) = [R(y_D - y_E), R(x_E - x_D), R\frac{(y_D^2 - y_E^2) + (x_E^2 - x_D^2)}{2}, 0, -(y_D - y_E), -(x_E - x_D), -\frac{(y_D^2 - y_E^2) + (x_E^2 - x_D^2)}{2}, 0]$;
for $i \leftarrow 1$ **to** 4 **do**
  $C_i = [a_i^+, b_i^+, c_i^+, d_i^+, a_i^-, b_i^-, c_i^-, d_i^-]^T \leftarrow$ the vector of the coefficients ;
  b(:,1) $= 0$;
  b(i,1) $= 1$;
  $C_i = A^{-1}b$;
end for

$$\frac{\partial \phi}{\partial \mathbf{n}|_{\Gamma_N}} = p(\mathbf{x}), \tag{14}$$

where $g(\mathbf{x})$ and $p(\mathbf{x})$ are the given functions on boundary.

In many plasma physics and engineering problems, the computational domain often includes multiple objects of different materials. Thus, in the domain there will be one or more interfaces which are the surfaces of the objects, shown in Fig. 1. For the simplicity of presentation, we assume that the domain $\Omega \subset R^2$ is a convex polygonal domain, and the interface Γ is an arbitrary curve separating Ω into two sub-domains Ω^-, Ω^+ such that $\Omega = \Omega^- \cup \Omega^+ \cup \Gamma$. The dielectric coefficient $\varepsilon(\mathbf{x})$ is discontinuous across the interface due to the material property changes, and we assume it is a piecewise constant function as follows:

$$\varepsilon(\mathbf{x}) = \begin{cases} \varepsilon^+, & \mathbf{x} \in \mathbf{\Omega}^+, \\ \varepsilon^-, & \mathbf{x} \in \mathbf{\Omega}^-, \end{cases} \tag{15}$$

where $\mathbf{x} = [x, y]^t$. In addition, the following interface jump conditions are satisfied across the interface:

$$[\phi]|_\Gamma = 0, \tag{16}$$

$$[\varepsilon \nabla \phi \cdot \mathbf{n}]|_\Gamma = 0 \text{ or } [\varepsilon(\boldsymbol{I} + \boldsymbol{\chi})\nabla \phi \cdot \mathbf{n}]|_\Gamma = 0, \tag{17}$$

Algorithm 4: Pseudocode of the IFE method in the CFIRM code.

$N_e \leftarrow$ number of mesh elements ;
$N_b \leftarrow$ number of the finite element nodes ;
$N_{lb} \leftarrow$ number of local finite element nodes ;
$T_b(j,i) \leftarrow$ global node indices of the jth point at the ith element ;
Initialize the matrix: $A = sparse(N_b, N_b)$ and $E_l = zeros(N_{lb}, N_{lb})$;
Initialize the vector: $b = sparse(N_b, 1)$ and $r_l = zeros(N_{lb}, 1)$;
`/* Assembly of the stiffness matrix and the load vector. */`
for $e \leftarrow 1$ **to** N_e **do**
  for $\beta \leftarrow 1$ **to** N_{lb} **do**
    for $\alpha \leftarrow 1$ **to** N_{lb} **do**
      if *Implicit PIC* **then**
        $c = \varepsilon(\boldsymbol{I} + \boldsymbol{\chi})$;
      else
        $c = \varepsilon$;
      end if
      $E_l(\beta,\alpha) = \int_{E_e} c\nabla\varphi_{i\alpha} \cdot \nabla\varphi_{i\beta}\, dxdy$;
    end for
    $r_l(\beta,1) = \int_{E_e} \rho\varphi_{i\beta}\, dxdy$;
  end for
  $A(T_b(:,e), T_b(:,e)) = A(T_b(:,e), T_b(:,e)) + E_l(:,:)$;
  $b(T_b(:,e),1) = b(T_b(:,e),1) + r_l(:,1)$;
end for
`/* Treat the boundary conditions. */`
$nbn \leftarrow$ the number of boundary nodes ;
$g \leftarrow$ the Dirichlet boundary condition ;
for $k \leftarrow 1$ **to** nbn **do**
  if *Dirichlet boundary* **then**
    $i \leftarrow$ the global node index of the kth boundary node ;
    $A(i,:) = 0$; $A(i,i) = 1$; $b(i,1) = g$;
  end if
end for
$nbe \leftarrow$ the number of boundary edges ;
$p \leftarrow$ the Neumann boundary condition ;
for $k \leftarrow 1$ **to** nbe **do**
  if *Neumann boundary* **then**
    $i \leftarrow$ the index of the element which contains the kth boundary edge ;
    for $\beta \leftarrow 1$ **to** N_{lb} **do**
      $r = \int_k cp\nabla\varphi_{i\beta} ds$;
      $b(T_b(\beta,i),1) = b(T_b(\beta,i),1) + r$;
    end for
  end if
end for
`/* Solve the sparse linear system using PCG and GMRES et al. */`
$\phi = A^{-1}b$;

where $\mathbf{n}$ is the unit normal vector of Γ pointing from Ω^- to Ω^+.

The traditional way to solve the elliptic equations involving complex interfaces is based on unstructured body-fitting grids. However, in order to improve the computational speed and efficiency of particle localization and tracking, the structured grids are preferred for the PIC model. Hence, we adopt the bilinear immersed-finite-element (IFE) method [74, 75, 76, 77, 78, 79, 80, 81, 82] to handle this dilemma in the CFIRM code. One major difference between the IFE method and the traditional finite element method is that interface elements are cut through and divided into two parts by the interface. There are only two types of rectangular interface elements as long as the mesh size is small enough, shown in Fig. 2.

Algorithm 5: Pseudocode of the PPR algorithms in the CFIRM code.

$N_{node} \leftarrow$ the number of mesh nodes ;
point_patch $\leftarrow$ store the information of each node and its sampling nodes ;
while $z < N_{node}$ **do**
  $n_z = \text{size(point\_patch)}$; `/* The number of the sampling nodes for each node */`
  for $i \leftarrow 1$ **to** n_z **do**
    $(\hat{x}_i, \hat{y}_i) = \dfrac{(x_i, y_i) - (x_z, y_z)}{max(\Delta x, \Delta y)}$;
    $\hat{\mathbf{x}}_i = [1, \hat{x}_i, \hat{y}_i, \hat{x}_i^2, \hat{x}_i\hat{y}_i, \hat{y}_i^2]^T$;
  end for
  $\phi_z = [\phi_1, \phi_2, ..., \phi_{n_z}]^T$;
  $\mathbf{A}_z = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, ..., \hat{\mathbf{x}}_{n_z}]^T$;
  $\mathbf{B}_z = \mathbf{A}_z^T \mathbf{A}_z$;
  $\mathbf{c}_z = \mathbf{B}_z^{-1} \mathbf{A}_z^T \phi_z$;
  $\hat{p}_z(\hat{x}_i, \hat{y}_i) = \hat{\mathbf{x}}_i^T \mathbf{c}_z,\ i = 1, 2, ..., n_z$;
  $p_z \leftarrow \hat{p}_z$; `/* The least-squares polynomial approximation (LSPA) of` ϕ_h `at` z `*/`
  $E_z^x = \partial p_z / \partial x$; `/* The electric field in x direction */`
  $E_z^y = \partial p_z / \partial y$; `/* The electric field in y direction */`
end while

The local bilinear IFE nodal basis functions [74, 75, 76] can be defined by piecewise bilinear polynomials:

$$\varphi_i(\mathbf{x}) = \begin{cases} \varphi_i^+(\mathbf{x}) = a_i^+ x + b_i^+ y + c_i^+ xy + d_i^+, \ \mathbf{x} \in T^+, \\ \varphi_i^-(\mathbf{x}) = a_i^- x + b_i^- y + c_i^- xy + d_i^-, \ \mathbf{x} \in T^-, \end{cases} \quad (i = 1, 2, 3, 4). \tag{18}$$

The coefficients of the piecewise linear polynomials can be uniquely determined by nodal value conditions, the continuity of the basis function condition on the interface, and the normal derivative jump condition on the interface. Algorithm 3 shows the pseudocode of computing the coefficients for local bilinear IFE nodal basis functions. After replacing the traditional finite element basis functions by the IFE basis functions on the interface elements, the assembly process of the stiffness matrix and load vector of the IFE method [75, 79, 81, 82] is similar to the traditional finite element methods. The corresponding pseudocode is demonstrated in Algorithm 4.

Remark 2.3. *For the axisymmetric coordinate system, the Poisson's equation can be written as*

$$-\varepsilon \left(\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{\partial^2}{\partial z^2} \right) \phi = \rho. \tag{19}$$

The IFE basis functions can still be obtained by the above Algorithm 3 on the mesh of the axisymmetric coordinate system. However, the Galerkin formulation of the finite element method in the axisymmetric coordinate system becomes

$$\int_\Omega \varepsilon \nabla \phi_h \cdot \nabla v_h r d\Omega = \int_\Omega \rho v_h r d\Omega + \int_{\partial\Omega} \varepsilon p v_h dS, \ \forall v_h \in S_h(\Omega), \tag{20}$$

where $S_h(\Omega)$ is the global bilinear immersed-finite-element space on the whole domain. Thus, the process of assembling the stiffness matrix and the load vector will be different from the situation in Cartesian coordinates. The main procedure is consistent with the above Algorithm 4, but the local stiffness matrix and vector will be replaced by

$$E_l(\beta, \alpha) = \int_{E_e} c \nabla \varphi_{i\alpha} \cdot \nabla \varphi_{i\beta} r \ dr dz, \tag{21}$$

Algorithm 6: Pseudocode of the adaptive particle management algorithm in the CFIRM code.

N_{min}, N_{max} ← set the minimum and maximum values of the number of simulation particles ;
w_{min} ← set a minimum weight to prevent the weights of certain particles from being split too small ;
w_{max} ← set a maximum weight to prevent the weights of certain particles from being merged too large ;
A, B ← the index of the two particles that are created by the split or coalescence ;
if $N_{tot} < N_{min}$ **then** `/* Split one particle into two */`
  while $i < N_{tot}$ **do**
    if $w_i > w_{min}$ **then**
      $\mathbf{x}$, $\mathbf{v}$, w ← the position, velocity, and weight of the particle before split ;
      $\mathbf{x}_A = \mathbf{x}_B = \mathbf{x}$, $\mathbf{v}_A = \mathbf{v}_B = \mathbf{v}$, $w_A = w_B = 0.5w$;
    end if
  end while
else if $N_{tot} > N_{max}$ **then** `/* Merge three particles into two */`
  count = 0 ;
  while $i < N_{tot}$ **do**
    if $w_i < w_{max}$ **then**
      count = count + 1 ;
      select the particle as a particle to be merged and store its information ;
    end if
    if *count* = 3 **then**
      $\mathbf{x}_j$, $\mathbf{v}_j$, w_j, $j = 1, 2, 3$ ← the positions, velocities, and weights of simulated particles before coalescence ;
      $\mathbf{x}_c$ ← the barycenter of the original three particles ;
      $X_\alpha = \sum_j w_j (x_{j\alpha} - x_{c\alpha})^2$, α indicates the components of the velocity and it can be replaced by x, y and z ;
      $M_\alpha = \sum_j w_j v_{j\alpha}$, $E_\alpha = \sum_j w_j v_{j\alpha}^2$, $W = \sum_j w_j$;
      $w_A = w_B = 0.5W$;
      $x_{A\alpha} = x_{c\alpha} + \sqrt{X_\alpha}$, $v_{A\alpha} = M_\alpha + \sqrt{E_\alpha - M_\alpha^2}$;
      $x_{B\alpha} = x_{c\alpha} - \sqrt{X_\alpha}$, $v_{B\alpha} = M_\alpha - \sqrt{E_\alpha - M_\alpha^2}$;
      count = 0 ;
    end if
  end while
Delete the particles that are split or merged, and add the particles that are created after the split or coalescence ;

and

$$r_l(\beta, 1) = \int_{E_e} \rho \varphi_{i\beta} r \; dr dz. \tag{22}$$

2.4. The PPR method of electric field calculation. When the potential is solved, the electric field $\mathbf{E^{n+1}}$ can be calculated by:

$$\mathbf{E}^{n+1} = -\nabla \phi^{n+1}. \tag{23}$$

In the traditional PIC method, this equation will be handled by using the central difference method, such as

$$E_x^{n+1}(i,j) = \frac{\phi^{n+1}(i-1,j) - \phi^{n+1}(i+1,j)}{2\Delta x}, \tag{24}$$

$$E_y^{n+1}(i,j) = \frac{\phi^{n+1}(i,j-1) - \phi^{n+1}(i,j+1)}{2\Delta y}. \tag{25}$$

Then the electric field applied to each particle can be obtained by interpolation, which uses the same weighting of the charge deposit. However, this method is only

Algorithm 7: The overall solution procedure of the CFIRM code.

```
Start ;
Read inputs: domain, objects, physical parameters, type of coordinate, step sizes,
  IMPIC_index, t_restart, T_dump, T_out and t_end, etc. ;
Initialization: nodes, meshes, boundaries, external electrostatic/magnetostatic fields, and
  normalization ;
/* Solve the initial electric field                                          */
if t = 0 then
    Obtain the initial potential using the IFE method (Algorithm 4) ;
    Obtain the initial electric field by the PPR method (Algorithm 5) ;
else
    t ← t_restart ;
    Read the electric field from the dump file ;
end if
/* Initial electric field solved                                             */
/* Start the main PIC/MCC loop                                               */
while t < t_end do
    Inject particles from the inlet boundaries ;
    Handle the collisions between the particles by the MCC model (Algorithm 8);
    if IMPIC_index then
        /* Implicit PIC model is selected                                    */
        Move particles (pre-push of Algorithm 1) and handle the particles crossing
          boundaries ;
        Charge deposit: obtain the total charge density at each node (Algorithm 2) ;
        Solve the potential by the IFE method (Algorithm 4);
        Obtain the electric field using the PPR method(Algorithm 5);
        Move particles (post-push of Algorithm 1) and handle the particles crossing
          boundaries ;
    else
        /* Explicit PIC model is selected                                    */
        Move particles (explicit move of Algorithm 1) and handle the particles crossing
          boundaries ;
        Charge deposit: obtain the total charge density at each node (Algorithm 2);
        Solve the potential by the IFE method (Algorithm 4);
        Obtain the electric field using the PPR (Algorithm 5);
    end if
    if t Mod T_dump = 0 then
        Output the dump files ;
    end if
    if t Mod T_out = 0 then
        Output the result files ;
    end if
    t = t + Δt ;
end while
/* Main PIC/MCC loop finished                                                */
End
```

suitable for non-interface elements. It is not applicable to interface elements and will lose the accuracy because the potential gradient is discontinuous at the interface. To address this issue, the polynomial-preserving recovery (PPR) technique is employed instead of the central difference method in the CFIRM code.

The PPR method [68, 71, 72] constructs approximate gradients for finite element solutions based on the principle of least squares fitting and generates more accurate gradients in arbitrarily shaped grids at a rational cost. To obtain the least-squares polynomial approximation (LSPA) of ϕ_h at each node, the related nodes of each node should be selected first. These nodes are called sampling nodes.

The selection of sampling nodes in the CFIRM code is shown in Fig. 3. For a nominal interior node, such as Fig. 3 (a), all adjacent nodes around it are selected as sampling nodes. For a boundary node that has three possible situations, the sampling points are not enough if just simply selecting its adjacent nodes. Thus, we continue to expand a layer of mesh nodes outward, shown in Fig. 3 (b)-(d). For an interface node, the sampling points are selected in the same way as the boundary node, which use two layers of mesh nodes, see Fig. 3 (e). The information of each node and its related sampling nodes are saved in the array 'point_patch' for the CFIRM code. Then, the LSPA of ϕ_h at each node can be obtained by the algorithm 5, and the electric field is calculated by taking the derivative of the LSPA. For the detailed theory of the PPR method, see the references [68, 71, 72].

2.5. The MCC method. The collision processes between the particles are taken into account by the Monte Carlo collision (MCC) method [83, 84, 11], in which the collisions between plasma and neutral atoms are handled via a null-collision technique [85, 86, 87]. Table 1 lists the main types of collisions, where G indexes the gases that include argon, helium, xenon, and so on. The cross sections of each gas come from LXCat.

Since the MCC method with a null-collision technique has been well developed, we only introduce the numerical implementation for the CFIRM code. Algorithm 8 shows the pseudocode of the MCC method applied in our CFIRM code. It should be noted that the velocities after a collision between the electrons or ions with the neutral atoms are calculated by Vahedi's method [85]. For elastic collision, we do not consider the energy loss of the incident particles. For excitation and ionization collisions, the incident particles will lose some energy whose value is the same as that of the corresponding threshold energy. In addition, a pair of new electron and ion will be created after the ionization collision. The energy of the new electron can be given by the energy balance. The energy of the new ion is assumed to be identical to the atom, and the velocity is sampled from a Maxwellian distribution using the atomic temperature, which is also applicable for charge exchange collisions between ions and atoms.

2.6. Overall implementation of the CFIRM code package. The CFIRM code is designed and implemented using Fortran language and adopting object-oriented programming. Some index variables are defined to choose which algorithm process will be executed, such as using a Cartesian coordinate system or an axisymmetric coordinate system, whether restart from the interrupt location, and which PIC model will be adopted to move particles. Some special data structures are defined as classes in the code, such as the classes of mesh, object, particle, and so on. All of the interactions with CFIRM are performed through input files using an easy and flexible syntax, see A.

In addition, we also introduce some numerical techniques to improve the simulation efficiency and accuracy. First, the variable weights algorithm is applied. Each type of particle can use an independent weight, which is mainly to reduce the discrepancy of the simulated particle numbers when the density difference between particles is too large. Moreover, different weights can also be used in different grids. It is very useful for axisymmetric models to ensure that the number of simulated particles in the grids with different radial positions is approximately equal. Second, an adaptive particle management algorithm is utilized to maintain the number of simulated particles at an optimal range. On one side, the more simulation particles are used, the more time it takes to push particles. On the other side, the fewer simulation particles are used, the numerical noise of the PIC model is greater. Thus, a reasonable range of the number of simulation particles will be set in this code. When the number of simulated particles is below the range, some particles will be

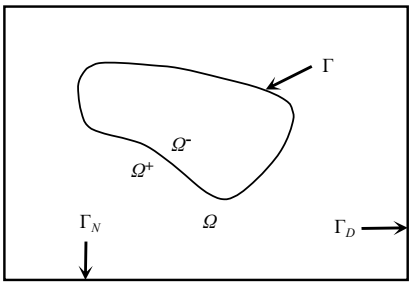

FIGURE 1. A sketch of the domain for the two-dimensional interface problem.

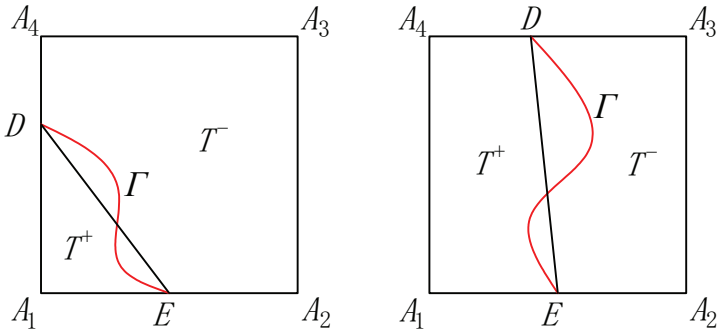

FIGURE 2. Two typical interface elements.

TABLE 1. List of the main types of collisions in CFIRM

Type	Reaction	Formula
electron-neutral	Elastic	$e + G \to e + G$
	Excitation	$e + G \to e + G^*$
	Ionization	$e + G \to e + G^+ + e$
ion-neutral	Elastic	$G^+ + G \to G^+ + G$
	Charge Exchange	$G^+ + G \to G + G^+$

randomly selected and split into multiple ones. When the number of simulated particles exceeds the range, some particles will be merged to reduce the number of simulated particles by following a ternary scheme which coalesces three particles into two to keep the preservation of both overall momentum and energy. Algorithm 6 shows the pseudocode of the adaptive particle management algorithm. The detailed algorithm for this part can be found in Ref. [88].

The simulation of the low-temperature plasma problems often requires a long period of iterative calculation to complete. Algorithm 7 shows the main iterative process of the CFIRM code in the form of pseudocode.

3. Code validations

In this section, we will verify the accuracy and reliability of the CFIRM code through some numerical experiments that have been used as benchmarks.

3.1. Numerical experiment 1. The first numerical experiment is a benchmark case from Ref. [89]. We will illustrate it in details in the following.

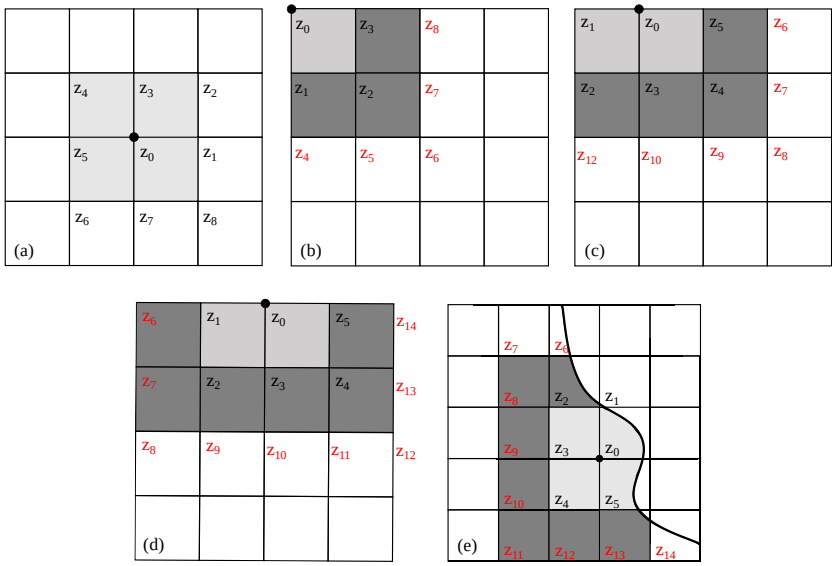

FIGURE 3. The selection of sampling nodes of the PPR method in different situations. (a) nominal interior node, (b)-(d) boundary nodes, (e) interface nodes.

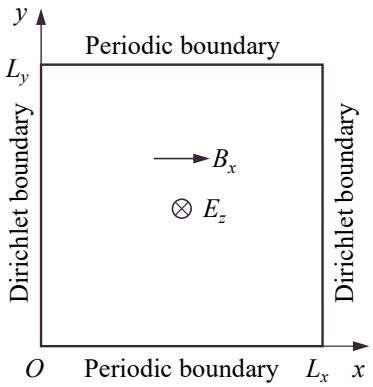

FIGURE 4. Schematic diagram of simulation domain for numerical experiment 1.

3.1.1. Problem description and simulation setup. It has a 2D square simulation domain with a side length of 1.28cm, as shown in Fig. 4. There is an external magnetic field $B_x = 200$G in the x-direction and an external electric field $E_z = 10$ kV m^{-1} in the z-direction. The left and right boundary conditions are the Dirichlet boundaries with a fixed potential $\phi_l = \phi_r = 0$V. The upper and lower boundary conditions are the periodic boundaries, i.e. $\phi_u = \phi_b$. We adopt the traditional explicit PIC model to simulate this case. The spatial step size is $\Delta x = \Delta y = 50$ μm and the temporal step size is $\Delta t = 1.5 \times 10^{-11}$s.

A collisionless plasma is considered only with electrons and singly charged xenon ions (Xe^+). In the initial stages, the electrons and ions are loaded uniformly in the domain (100 macro-particles per cell for each species) with a density $n_0 = 5 \times 10^{16}$ m^{-3}, and their velocities are determined from the Maxwellian distribution at the initial temperature $T_{e,0}$ and $T_{i,0}$. When the particles cross the left or right boundary, they will be absorbed and removed from the domain. When the particles

TABLE 2. The physical and numerical parameters of the experiment 1.

Parameters	Symbol	Value	Unit
Side length of domain	$L_x = L_y$	1.28	cm
Virtual axial length	L_z	1.0	cm
Spatial step size	$\Delta x = \Delta y$	50	μm
Number of cells	N_{cell}	256×256	-
Plasma density	n_0	5×10^{16}	m^{-3}
Electron temperature	$T_{e,0}$	10	eV
Ion temperature	$T_{i,0}$	0.5	eV
Number of particles/cell	$N_{ppc,ini}$	100	-
Magnetic field	B_x	200	G
Electric field	E_z	10	kV m^{-1}
Time step	Δt	1.5×10^{-11}	s
Average time range	N_a	$1000\Delta t$	s
Final time	t_{max}	20	μs

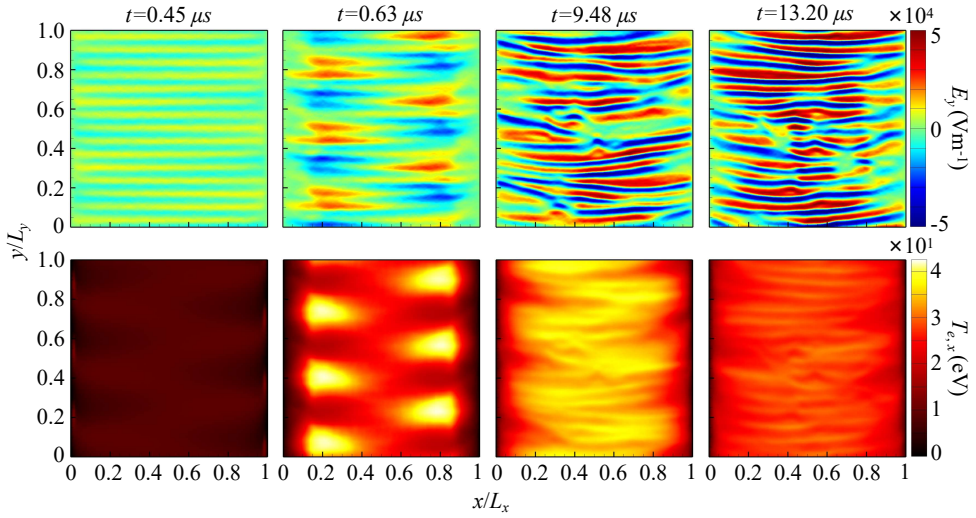

FIGURE 5. 2D snapshots of electric field E_y (first row) and electron temperature $T_{e,x}$ (second row) at times $t = 0.45$ μs, $t = 0.63$ μs, $t = 9.48$ μs, and $t = 13.20$ μs.

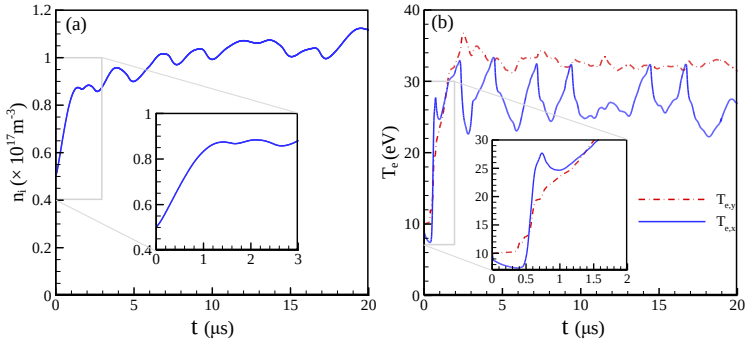

FIGURE 6. Temporal profiles of (a) ion density n_i and (b) electron temperatures $T_{e,x}$ and $T_{e,y}$ up to 20μs.

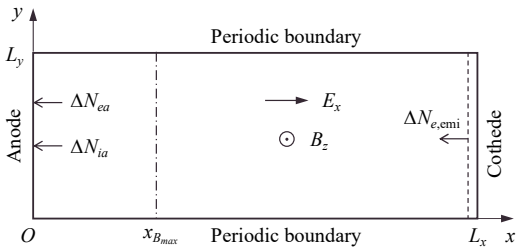

FIGURE 7. Schematic diagram of simulation domain for numerical experiment 2.

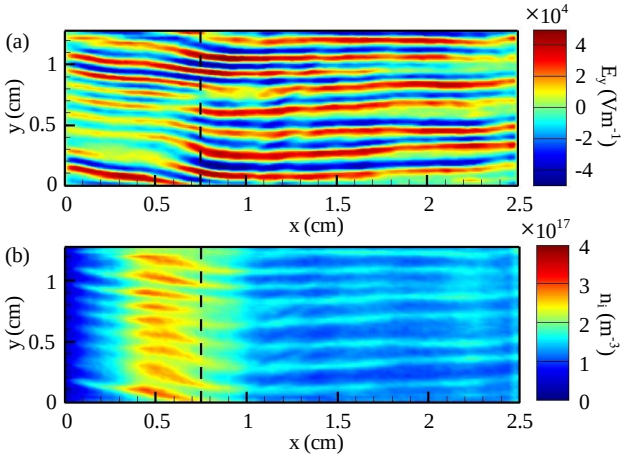

FIGURE 8. The 2D spatial distributions of (a) electric field E_y and (b) ion density n_i obtained with code CFIRM at $t = 20$ μs. The dashed line corresponds to the position of the maximum magnetic field.

cross the upper or lower boundary, they will enter the domain from the other boundary. It should be noted that all particles are initialized at the plane $z = 0$. We set a virtual axial in the z-direction, the range is $\pm L_z$. When the particles cross the virtual boundary (i.e. $|z| > L_z$), they will be re-injected into the plane $z = 0$ with the same x and y, but the velocity will be initialized based on their initial temperature (i.e. $T_{e,0}$ and $T_{i,0}$). The diagnostic data are averaged during the computation over $1000\Delta t$ and the output files are generated every N_a. A full list of the physical and numerical parameters is summarized in Table 2.

In addition, a fixed ionization source term is used to generate new particles to compensate for the particle loss at the left and right boundaries. Assume the ionization is uniform in the y-direction and its profile in the x-direction is given by

$$\begin{cases} S(x) = S_0 cos\left(\pi \dfrac{x - x_M}{x_2 - x_1}\right), \ x_1 \le x \le x_2, \\ S(x) = 0, \ x < x_1 \text{ or } x > x_2, \end{cases} \tag{26}$$

where $S_0 = 8.9 \times 10^{22}$ m^{-3}s^{-1} is the maximum value of the source term, $x_2 - x_1$ is the width of the ionization zone with $x_1 = 0.09$ cm and $x_2 = 1.19$ cm, and $x_M = L_x/2$ is the median line. Based on this source term, the number of physical pairs of Xe^+/e^- to be injected into the domain at each iteration can be calculated

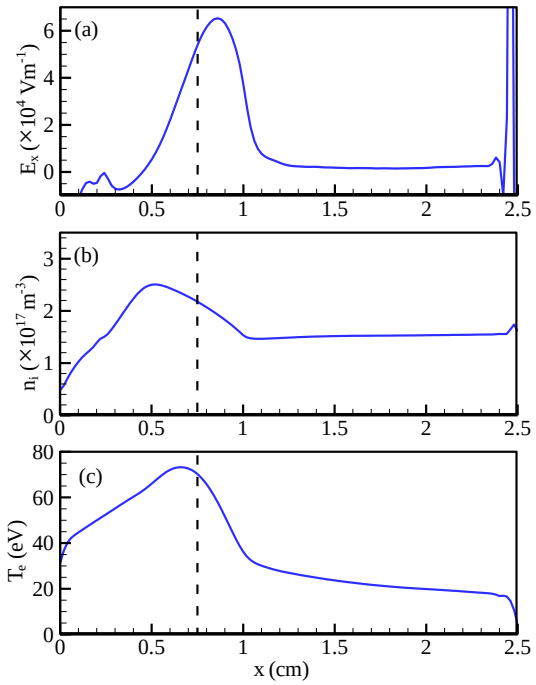

FIGURE 9. The profiles of (a) the electric field E_x, (b) ion density n_i, and (c) electron temperature T_e that are averaged in space (y-direction) and in time (between 16 and 20 μs).

by

$$N_{\mathrm{Xe}^+/\mathrm{e}^-} = L_y \Delta t \int_0^{L_x} S(x)dx. \tag{27}$$

The new particles still are injected into the plane $z = 0$ and their velocities are initialized based on their initial temperature (i.e. $T_{e,0}$ and $T_{i,0}$). The location (x_i, y_i) of each particle is randomly chosen according to the ionization profile (Eq. 26), it can be explicitly expressed as

$$\begin{cases} x_i = arcsin(2\alpha - 1)\dfrac{x_2 - x_1}{\pi} + x_M, \\ y_i = \beta L_y, \end{cases} \tag{28}$$

where α and β are the random numbers between 0 and 1. For more detailed settings on this numerical example, please refer to Ref. [89].

3.1.2. Simulation results. This numerical experiment is calculated by the C-FIRM code, and the total simulation time is only 20μs. The simulation results are consistent with the results in Ref. [89].

Fig. 5 shows the spatial distributions of the electric field in the y-direction and the electron temperature in the x-direction at different time steps. It indicates that the electric field E_y has a simple instability in the $\mathbf{E} \times \mathbf{B}$ direction at the beginning ($t = 0.45$ μs). Then, another instability occurs, which leads the electric field E_y to display a complex instability with oscillations in both x and y-directions at $t = 0.63$ μs. These instabilities are identified as the electron cyclotron drift instability (ECDI) and the modified two-streams instability (MTSI) respectively, which are carefully discussed in Ref. [89]. At this point, the electrons are periodically heated in the x-direction significantly. Next, the plasma enters a dynamically changing

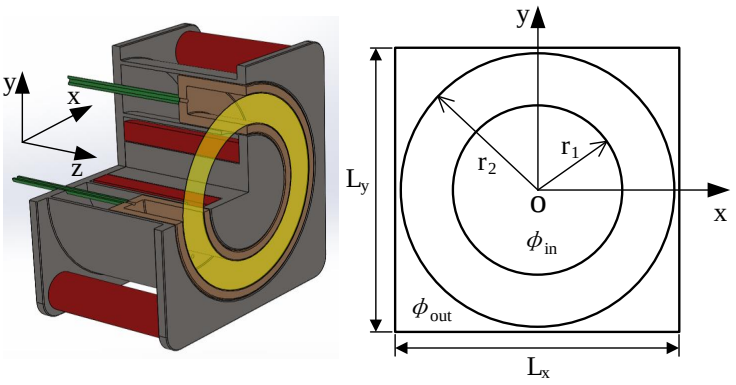

FIGURE 10. The hall thruster model and simulation domain.

TABLE 3. The main parameters of the simulation domain and initialization for the hall thruster.

Parameters	Symbol	Value	Unit
Total length in x direction	L_x	5.2	cm
Total length in y direction	L_y	5.2	cm
Virtual length in z direction	L_z	2.2	cm
Inner wall radius	r_1	1.825	cm
Outer wall radius	r_2	2.525	cm
Plasma density	n	1.0×10^{17}	m^{-3}
Ion drift velocity	v_d	1.7×10^{4}	ms^{-1}
Initial electron temperature	T_e	10.0	eV
Initial ion temperature	T_i	1.0	eV
Spatial step size	Δh	3.7143×10^{-4}	m
Temporal step size	Δt	2.8027×10^{-11}	s
Axial electric field	E_z	2.0×10^{4}	Vm^{-1}
Particle weight	w	4.0×10^{6}	–

state with the influence of the two instabilities. There are two typical states that can be observed. The first one is that the instability in the x-direction is visible, and the electron temperature $T_{e,x}$ has an obvious local maximum at this moment, like at $t = 9.48$ μs. The second one is that the instability in the x-direction is weaker, and the distribution of the electron temperature $T_{e,x}$ is relatively uniform, like at $t = 13.20$ μs.

Moreover, Fig. 6 shows the temporal profiles of ion density and electron temperature in the x and y directions from 0 to 20 μs. The ion density is increased linearly due to the fixed ionization source term at the beginning. Then the ion density enters a quasi-stable state with periodic changes, where the number of the ions lost at the left and right boundaries approaches equilibrium with the number of newly generated ions. The electron temperature $T_{e,y}$ maintains its initial energy for about 0.5 μs at the starting stage, then quickly acquires energy but stabilizes at around 33 eV under the limitations of the virtual axis in the z-direction. The electron temperature $T_{e,x}$ initially decreases slightly and then rapidly increases, and it enters an oscillatory plateau, in which the maximum and minimum values correspond to the two instability states in Fig. 5. This result is the same as the one in Fig. 3 of Ref. [89]. It provides sufficient evidence for the effectiveness and correctness of the CFIRM code.

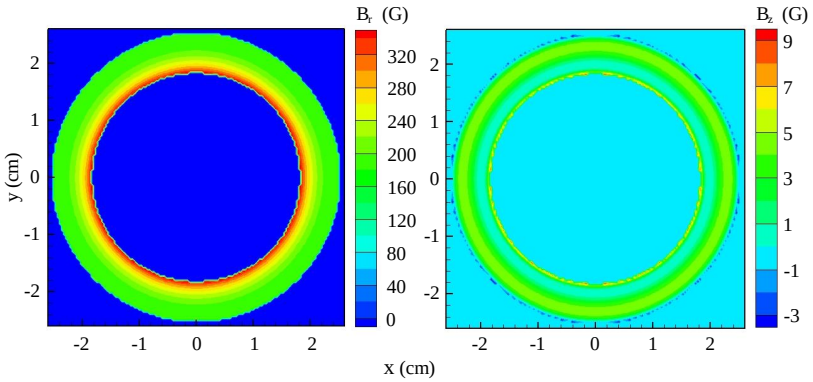

FIGURE 11. The magnetic field profiles in this simulation.

3.2. Numerical experiment 2. The second example is a benchmark case from Ref. [90].

3.2.1. Problem description and simulation setup. The geometry structure of the simulation is shown in Fig. 7, and it is a rectangular region with $L_x = 2.5$ cm and $L_y = 1.28$ cm. There is an external magnetic field $B_z(x)$ in the z-direction that has a profile in the x-direction. The left boundary is an anode with a fixed potential of $\phi_l = 200$ V. The right boundary is a cathode whose potential is $\phi_r = 0$ V. The upper and lower boundary conditions are the periodic boundaries, namely $\phi_u = \phi_b$. By using the implicit PIC scheme, we can adopt relatively larger step sizes for the simulation. The spatial step size is $\Delta x = \Delta y = 200$ μm and the temporal step size is $\Delta t = 2 \times 10^{-11}$ s.

The electrons and singly charged xenon ions (Xe^+) are initially distributed with a density of $n_0 = 5 \times 10^{16}$ m^{-3} uniformly throughout the domain (150 macro-particles per cell for each species). Their velocities are sampled from a Maxwellian distribution at the initial temperature $T_{e,0}$ and $T_{i,0}$. When the particles move out from the upper or lower boundary, they will enter the domain with the same states from the other boundary. When the particles reach the left or right boundary, they are removed. Moreover, the electrons are injected from the cathode for each iteration, and the emission position is set on a line segment, at 1 mm away from the right boundary, as illustrated in Fig. 7. Based on the current conservation, the number of electrons emitted can be obtained by

$$\Delta N_{e,emi} = \Delta N_{ea} - \Delta N_{ia}, \tag{29}$$

where ΔN_{ea} and ΔN_{ia} are the number of electrons and ions that are removed from the left boundary at each step.

Furthermore, the profile of the magnetic field is given by

$$B(x) = a_k \exp\left\{-\frac{(x - x_{B_{max}})^2}{2\sigma_k^2}\right\} + b_k. \tag{30}$$

If $x < x_{B_{max}}$, $k = 1$; and if $x > x_{B_{max}}$, $k = 2$. The values of the a_k and b_k can be calculated according to the following conditions: $B_{max} = 10$ mT, $x_{B_{max}} = 0.75$ cm, $\sigma_1 = \sigma_2 = 0.625$ cm, $B(x = 0) = 6$ mT, and $B(x = L_x) = 1$ mT. This benchmark case still has a fixed source term to replace the process of ionization. The profile of the ionization rate $S(x)$ is the same as Eq. 26, but $S_0 = 5.23 \times 10^{23}$ $m^{-3}s^{-1}$, $x_1 = 0.25$ cm, $x_2 = 1$ cm, and $x_M = (x_1 + x_2)/2$. Similarly, the number of injected e^--Xe^+ pairs is calculated by Eq. 27 at each step. Then the injected positions

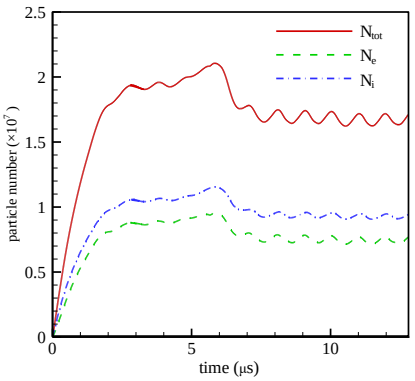

FIGURE 12. The curve of the simulated particle number with time.

(x_i, y_i) are randomly given by Eq. 28. The detailed information on this case can be found in Ref. [90].

3.2.2. Simulation results. Fig. 8 shows the spatial distributions of the electric field E_y and ion density at $t = 20\mu$s. The basic trends of distributions are consistent with the results in Fig. 4 of Ref. [90], but the oscillation wavelength of plasma is larger in our simulation results, because of the larger spatial and temporal step sizes (about four times than that in Ref. [90]) are applied to reduce the computational time. Although the larger step size will result in a decrease in the resolution of high-frequency and small-wavelength oscillations, it will not affect the observation of the overall characteristics of the plasma. Fig. 9 shows the profiles of the electric field E_x, ion density, and electron temperature in the x-direction, which are averaged in the y-direction and in time (between 16 and 20 μs). These results in Ref. [90] are given in Fig. 5. By comparison, it can be found that our results can provide reasonable distributions, for instance, the position of the peak value and the profile for each parameter are the same as in the literature. However, since larger spatial and temporal step sizes used for the implicit PIC method lead to the excessive damping of the high-frequency modes [90], the peak value of each parameter is different. This would be improved by using smaller step sizes when needed. In summary, through this example, we validate the implicit PIC part of the CFIRM code package. It is capable of significantly reduce the computational cost in large-scale simulations but maintain the most basic physical characteristics.

4. Application to the low-temperature plasma simulation

In this section, we apply the CFIRM code to simulate two typical low-temperature plasma problems to demonstrate the simulation capability. One is the azimuthal plasma oscillations of a hall thruster that is widely used in the aerospace propulsion field. The other one is a radio frequency capacitively coupled plasma (CCP) reactor that is widely applied in the semiconductor industry. In the following, we will first introduce the physical problems and the setups, and then present the results and discussions.

4.1. Application 1: hall thruster.

4.1.1. Description and setup of a 2D annular model for hall thruster. The discharge channel of a hall thruster is composed of two insulated cylindrical tubes where gas discharge and plasma acceleration occur simultaneously. In this simulation, a two-dimensional annular outlet section is selected as the simulation domain, which is used to observe the azimuthal behavior of plasma, as shown

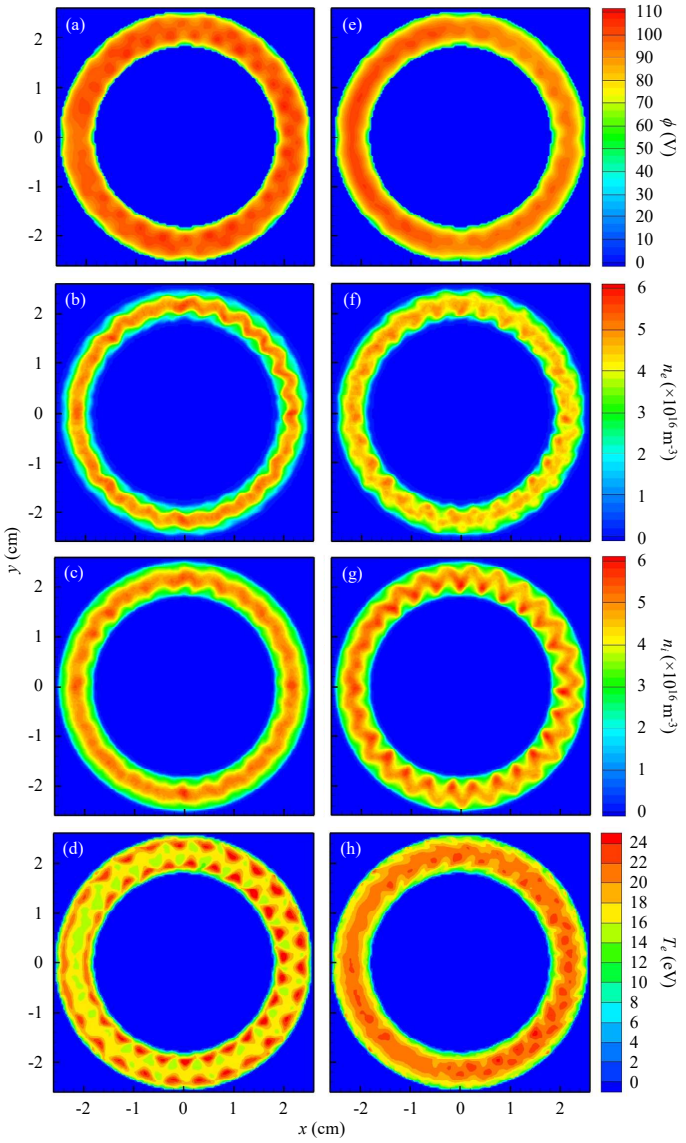

FIGURE 13. The results of potential, electron density, ion density, and electron temperature for two typical moments: (a)-(d) at 9.585 μs, and (e)-(h) at 10.034 μs.

in figure 10. The main geometric and physical parameters of this simulation are listed in table 3. The region larger than r_2 is regarded as the outer wall, and the region smaller than r_1 is the inner wall. Surface charge deposition at the wall is not considered in this simulation. We assume that both two walls have the fixed potential with $\phi_{out} = \phi_{in} = 0$V, thus the potential solved in this simulation is only a relative value. In addition, there is a fixed axial electric field $E_z = 2.0 \times 10^4$V/m that is used in the whole domain. The magnetic field distribution is the same as in the azimuthal direction but not uniform in the radial direction. The profiles are shown in figure 11.

Xenon gas is used as an unchanged background gas with the density of 2×10^{18} m^{-3} and the temperature of 640 K. Only the electrons and singly charged xenon

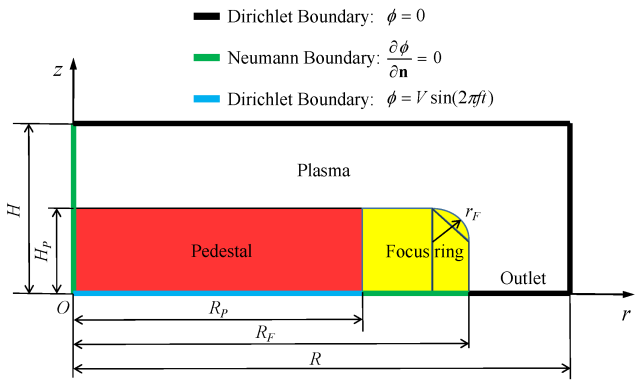

FIGURE 14. Schematic diagram of simulation domain for the CCP reactor.

TABLE 4. Geometric and physical parameters of the CCP reactor.

Parameters	Symbol	Value	Unit
Height of domain	H	4.8	cm
Radius of domain	R	16.8	cm
Height of pedestal	H_P	1.8	cm
Radius of pedestal	R_P	10.0	cm
Radius of focus ring	R_F	13.0	cm
Fillet radius of focus ring	r_F	0.5	cm
The voltage amplitude of RF source	V	200	V
The frequency of RF source	f	10	MHz
Argon gas pressure	p_g	100	mTorr
Argon gas temperature	T_g	300	K
Initial plasma density	n_0	1×10^{15}	m^{-3}
Electron temperature	T_e	2.585	eV
Ion temperature	T_i	0.2585	eV
Spatial step size	$\Delta r = \Delta z$	5×10^{-4}	m
Temporal step size	Δt	2.8571×10^{-11}	s

Input file 1: The parameter information of the domain specified in input file 'mesh.inp'	
0., 0.	! xmin(zmin), ymin(rmin)
96., 336.	! xmax(zmax), ymax(rmax)
97, 337	! nx(nz), ny(nr): total node number in each dimension
1., 1.	! hx, hy: spatial step size in each dimension

ions (Xe^+) are considered in the plasma. The main plasma parameters can be obtained in table 3. In this simulation, an equal amount of electrons and ions are randomly injected into the circular region at each step. The number of rejected particles can be calculated by

$$\Delta N = nSv_{\mathrm{d}}\Delta t/w, \tag{31}$$

where $S = \pi(r_2^2 - r_1^2)$ is the area of the annular region. The thermal velocities in each direction of electrons and ions are loaded by a full Maxwellian distribution, and the ions have an additional fixed drift velocity 1.7×10^4 m/s in the axial direction. When the particles collide with the annular walls, they will be removed. In addition, the particles also be tracked along the axial (or z) direction, and a virtual acceleration distance of 2.2 cm is assumed. When the particles cross the

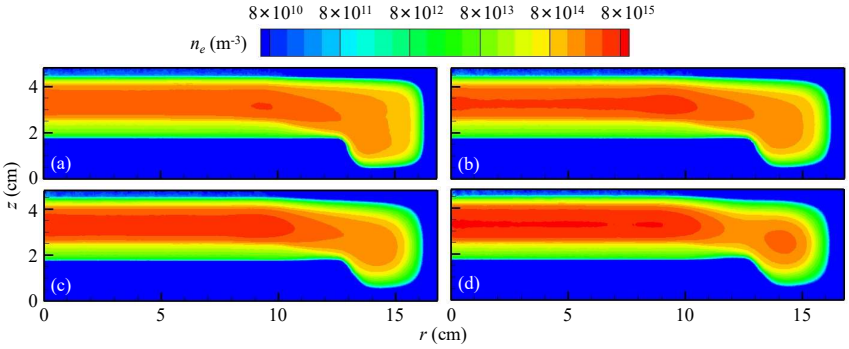

FIGURE 15. The distributions of electron density at different times. (a) 0.71μs, (b) 1.43μs, (c) 2.86μs, and (d) 8.57μs.

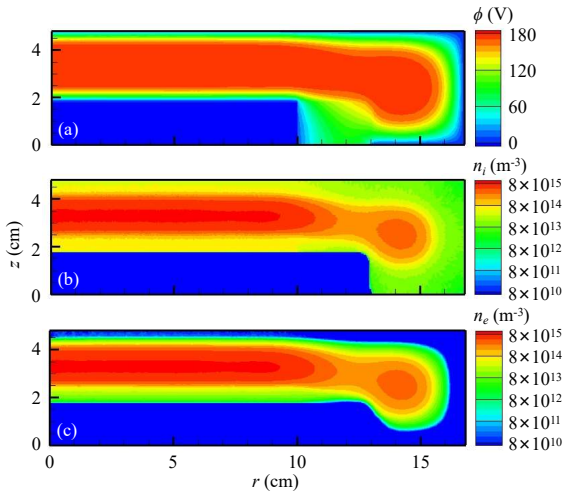

FIGURE 16. The time-averaged results of (a) potential, (b) ion density, and (c) electron density.

Input file 2: The parameter information of the objects specified in input file 'object.inp'	
4,-2	! the number of objects, index of the vacuum region
3	! objects(1)%Shape: 1-circle, 2-triangle, 3-box ...
0	! objects(i)%Axis: 0-no axis, 1-x-axis symmetry, 2-y-axis symmetry
36.0, 200.0	! objects(i)%Dimensions(1:2): length and width
60.0, 0.0	! objects(i)%Locations(1,:): lower corner coordinate
96.0, 200.0	! objects(i)%Locations(2,:): upper corner coordinate
-2, -1	! objects(i)%Regions(1:2): 1-inside, 2-outside
300	! objects(i)%Phi: potential
4.0	! objects(i)%Eps: dielectric constant
...	repeat the above process to define multiple objects

Input file 3: The parameter information of the IFE method specified in input file 'ife.inp'	
0	! delta: 0-Cartesian coordinate, 1-axisymmetric coordinate
1	! sparse linear solver: 1-PCG, 2-Gauss-Seidel, 3-Newton, 4-GMRES
500	! the maximum number of iterations
1.D-8	! the tolerance of solver

Input file 4: The PIC control options and plasma parameters specified in input file 'pic.inp'	
0, 0	! restart option (1 = yes), restart point
.True.	! IMPIC_index
1.0D15	! the reference value of plasma density (m^{-3})
2.585	! the reference value of temperature (eV)
300000, 0.1	! total steps, temporal step size (dimensionless)
1000	! the interval between output results: T_{out}
5000	! the interval between dumps: T_{dump}
2	! the total number of types of simulated particles
-1.6022D-19,9.1095D-31	! the charge and mass of an electron
1.6022D-19,6.64D-26	! the charge and mass of an ion
5.0D4,5.0D4	! particle weights: electron, ion
1	! index of the electron
1.0D15	! the density of the electron (m^{-3})
2.585	! the temperature of the electron (eV)
0.0, 0.0, 0.0	! drifting velocities in each dimension (m/s)
2	! index of the ion
1.0D15	! the density of the ion (m^{-3})
0.02585	! the temperature of the ion (eV)
0.0, -8.099D4, 0.0	! drifting velocities in each dimension (m/s)

Input file 5: The parameter information of the MCC method specified in input file 'mcc.inp'	
96.4d20	! the density of the gas (m^{-3})
300	! the temperature of the gas (K)
1,3	! index of the electron, the total collision types between electrons and atoms
180000	! elastic collision
180001	! excitation collision
180002	! ionization collision
2,2	! index of the ion, the total collision types between ions and atoms
180011	! elastic collision
180012	! charge exchange collision

lower and upper boundaries of the acceleration distance, they will be randomly re-injected into the domain from the other boundary with the same state as the initialization. Finally, the simulation is performed with the spatial step size of 3.7143×10^{-4} m and the temporal step size of 2.8027×10^{-11} s.

4.1.2. Results and discussions. Fig. 12 shows the number of simulated particles from the beginning to 13 μs. The number of simulated particles increases sharply at the beginning with the particle injection and ionization. After about 7 μs, the number of particles has reached equilibrium. That is, the increased particles are approximately equal to the lost particles at the walls. It indicates that the simulation has stabilized, but the number of particles fluctuates periodically, and the fluctuation period is approximately 1 μs.

From the above, we know the system has an oscillation between two typical states, and the two states are developing alternately when the simulation is stable. The first one is seen at $t = 9.585$ μs, and the results can be found in Fig. 13 (a)-(d). The azimuthal oscillations of the potential, electron and ion density distribution are obvious, and the wavelength of the oscillation is approximately 5 mm. The azimuthal oscillation characteristic of ion density is weaker than the electron density, and the ion density is mainly concentrated in the middle of the channel. The second state is seen at $t = 10.034$ μs, and the results can be found in Fig. 13 (e)-(h). In contrast, the ion density exhibits more perfect oscillation characteristics, and the peak density near the inner wall is higher than that near the outer wall.

At this time, the periodic oscillation structure of electron density and potential is disrupted.

The difference between the two states is likely caused by the periodic heating of electrons. As shown in Fig. 13 (f), the electron temperature near the walls is higher than that inside the channel and exhibits periodic peaks along the walls at $t = 9.585$ μs. It indicates that the electrons will collide with the walls more intensely, and the loss of electrons on the wall will increase, leading to a decrease in the number of particles. As the number of electrons near the wall decreases, electrons within the channel dominate, which leads to the periodic structure of electrons being weakened, like Fig. 13 (h). Since the mass of the ion is much greater than that of the electron, the response time of ions to the electric field changes lags behind that of electrons. This leads to inconsistency in the periodic characteristics of ions and electrons over time. This example demonstrates the applicability of the CFRIM package to the low-temperature plasma problem with complex interfaces in a two-dimensional Cartesian coordinate system.

4.2. Application 2: capacitively coupled plasma reactor.

4.2.1. Description and setup of the CCP problem. The capacitively coupled plasma reactor simulation, which is established in a two-dimensional axisymmetric coordinate system simulated, is set up as shown in Fig. 14. A pedestal disk is located in the middle of the computational domain (red region), and a focus ring (yellow region) surrounds it. The focus ring is divided into three simple geometric regions that are read through the input file, namely rectangle, quadrilateral, and arch, then they are merged into one object in the code. The lower boundary of the pedestal is connected to a radio-frequency source, thus the potential in this boundary is $\phi = V\sin(2\pi ft)$, where V is voltage amplitude, f is frequency, and t is time. The left boundary is the axis of symmetry, hence treated as a Neumann boundary with $\partial\phi/\partial\boldsymbol{n} = 0$. The focus ring is a dielectric ($\varepsilon = 4\varepsilon_0$, ε_0 is the permittivity of vacuum), and its lower boundary is also set as a Neumann boundary with $\partial\phi/\partial\boldsymbol{n} = 0$. The remaining boundaries are the Dirichlet boundary with $\phi = 0$. Table 4 lists the important geometric and physical parameters of the CCP reactor.

Argon gas is used with the pressure of 100 mTorr and the temperature of 300 K, and uniformly distributed throughout the whole computational domain as a background gas. The initial plasma density is uniform at 1×10^{15} m^{-3} for all mesh elements. Only the electrons and singly charged argon ions (Ar^+) are considered, and their temperatures are 2.585 eV and 0.2585 eV respectively. The particles will be reflected on the left boundary, and absorbed on other boundaries. However, the secondary electron emission (SEE) is considered on the upper boundary of the pedestal, which includes the induced SEE by electron or ion. And the induced SEE by electron consists of inelastic or elastic reflected electron and real secondary electron.

4.2.2. Results and discussions. The simulation is calculated with the spatial step size of $\Delta r = \Delta z = 5\times10^{-4}$ m and the temporal step size of $\Delta t = 2.8571\times10^{-11}$ s. Fig. 15 shows the distributions of electron density at different simulating times. At the beginning, the electrons diffuse along the radial direction, but they are constrained by the focus ring. Then the electron density gradually increases at the edge of the pedestal, such as the result of 0.71μs. Next, the electron density inside the pedestal also increases and finally forms a uniform density distribution. The distribution of ion density is similar to the electron density because the ions are created through the ionization between the electrons and the gas.

The time-averaged results over ten periods are shown in Fig. 16 when the simulation is stable. The potential is relatively uniform in the center of the reactor,

Algorithm 8: Pseudocode of the MCC method in the CFIRM code.

$n_g \leftarrow$ the gas density ; $\sigma \leftarrow$ the cross-section ; $N \leftarrow$ the total number of electrons or ions ;
$\mathbf{v} = (vx, vy, vz) \leftarrow$ the velocity of electrons or ions before collision ;
$\mathbf{v}' = (v'x, v'y, v'z) \leftarrow$ the velocity of electrons or ions after collision ;
$\mathbf{g} = (gx, gy, gz) \leftarrow$ the relative velocity of two colliding particles ;
for isp $\leftarrow 1$ **to** isp_tot **do**
  $\omega_m = max\{n_g\sigma(E)v_r\}$; $P_m = 1 - e^{-\omega_m \Delta t}$; $N_m = \mathrm{INT}(N(\mathrm{isp}) * P_m)$;
  for $i \leftarrow 1$ **to** N_m **do**
    $R_1, R_2, R_3, R_4, R_5 \leftarrow$ random numbers belong to [0, 1] ;
    i_part $= 1 + \mathrm{INT}(N(\mathrm{isp}) * R_1)$;
    if isp $= 1$ **then** `/* Collisions between electrons and atoms */`
      if $R_2 <= \omega_1/\omega_m$ **then** `/* Elastic collision */`
        $cos\theta = E(\mathrm{i\_part}) + 2 - 2 * (1 + E(\mathrm{i\_part}))^{R_3}/E(\mathrm{i\_part})$;
        $sin\theta = \sqrt{(1 - cos^2\theta)}$; $\varphi = 2 * \pi * R_4$;
        $g = \sqrt{gx * gx + gy * gy + gz * gz}$;
        $hx = \sqrt{gy * gy + gz * gz} * cos\varphi$;
        $hy = -(gx * gy * cos\varphi + g * gz * sin\varphi)/\sqrt{gy * gy + gz * gz}$;
        $hz = -(gx * gz * cos\varphi - g * gy * sin\varphi)/\sqrt{gy * gy + gz * gz}$;
        $v'x = vx - (gx * (1 - cos\theta) + hx * sin\theta) * M/(M + m)$;
        $v'y = vy - (gy * (1 - cos\theta) + hy * sin\theta) * M/(M + m)$;
        $v'z = vz - (gy * (1 - cos\theta) + hz * sin\theta) * M/(M + m)$;
      else if $\omega_1/\omega_m < R_2 <= (\omega_1 + \omega_2)/\omega_m$ **then** `/* Excitation collision */`
        $E^e_{old} = E(\mathrm{i\_part}) - E^{ex}_{Threshold}$; $\mathbf{v} = \sqrt{E^e_{old}/E(\mathrm{i\_part})} * \mathbf{v}$;
        $v'x, v'y, v'z \leftarrow$ obtain the velocity in the same way as 'Elastic collision' ;
      else if $(\omega_1 + \omega_2)/\omega_m < R_2 <= (\omega_1 + \omega_2 + \omega_3)/\omega_m$ **then** `/* Ionization collision */`
        $E' = E(\mathrm{i\_part}) - E^{ion}_{Threshold}$; $E^e_{old} = 10 * tan(R_5 * tan^{-1}(E')/20)$;
        $\mathbf{v}_0 = \mathbf{v}$; $\mathbf{v} = \sqrt{E^e_{old}/E(\mathrm{i\_part})} * \mathbf{v}_0$;
        $v'x, v'y, v'z \leftarrow$ obtain the velocity in the same way as 'Elastic collision' ;
        $E^e_{new} = E' - E^e_{old}$; $\mathbf{v} = \sqrt{E^e_{new}/E(\mathrm{i\_part})} * \mathbf{v}_0$; `/* A new electron is created */`
        $vx^e_{new}, vy^e_{new}, vz^e_{new} \leftarrow$ obtain the velocity in the same way as 'Elastic collision' ;
        $vx^i_{new}, vy^i_{new}, vz^i_{new} \leftarrow$ initialization using the atom parameters ;
        `/* A new ion is created */`
      else `/* Null collision */`
        return ;
    else if isp $= 2$ **then** `/* Collisions between ions and atoms */`
      if $R_2 <= \omega_1/\omega_m$ **then** `/* Elastic collision */`
        $v'x, v'y, v'z \leftarrow$ the velocity is obtained in the same way as above ;
      else if $\omega_1/\omega_m < R_2 <= (\omega_1 + \omega_2)/\omega_m$ **then** `/* Charge Exchange collision */`
        $vx^i_{new}, vy^i_{new}, vz^i_{new} \leftarrow$ initialization using the atom parameters ;
        `/* A new ion is created */`
      else `/* Null collision */`
        return ;
  end for
end for

and a plasma sheath is formed near each boundary. The distributions of ion and electron density are basically the same in the center of the reactor, indicating that the plasma has good quasi-neutrality. However, at the sheath regions, the electron density is significantly lower than the ion density. The number of electrons in the sheath region is scarce. But it has a considerable distribution only near the upper surface of the pedestal because the secondary electron is emitted in here. This example shows the computational ability in the axisymmetric coordinate system of the CFIRM code for dealing with multiple objects, complex interfaces, and gas discharge problems.

5. Conclusions

In this paper, we presented a most recently developed two-dimensional fully kinetic low-temperature plasma simulation code package, namely CFIRM, which has strong integration and versatility. The CFIRM code is designed based on the continuous Galerkin IFE-PIC-PPR-MCC framework, which can be used to handle complex interface problems on the Cartesian meshes in both Cartesian and axisymmetric coordinate systems, while tracking the motion trajectory of charged particles by the explicit or implicit PIC schemes and simulating the collisions between the plasma and neutral gas. In addition, the PPR technology is used to ensure the accuracy of the electric field in the interface elements. The variable weights and adaptive particle management algorithms are added to reduce the memory utilization rate. Two validation experiments were carried out and the results showed good consistency with the benchmark cases. These tests effectively validated the CFIRM package based on the existing works.

Two typical low-temperature plasma problems were simulated by the CFIRM code, including the azimuthal oscillation of a hall thruster and the capacitively coupled plasma reactor. The results demonstrated the applicability and computational capability of the CFIRM package for handling complex interfaces, multiple objects, and gas discharge problems in both Cartesian and axisymmetric coordinate systems. Based on the performance of the current package, it is an interesting and important future work to update it for parallel computation.

Appendix A. Input files

In this appendix, we introduce the input files that are used to perform the interaction between the user and the CFIRM package, as shown in the following tables. The simulation domain and the mesh size are defined in 'mesh.inp', where the domain is determined by the coordinates of the vertices in the lower and upper corners of a rectangular area. Other physical structures (objects) in the domain are defined in 'object.inp'. We can define multiple simple objects that combine into a complex object by Boolean operations in the code. One general data structure is to use up to 8 parameters to describe the properties of an object. Users can apply this data structure to define various geometries such as circles, triangles, rectangles, arches, etc. The important information about the IFE method can be defined in 'ife.inp', including the selection of the coordinate type and linear system solver.

The PIC control options and plasma parameters are specified in 'pic.inp'. It is first decided whether to restart and use the implicit PIC scheme, and the restart point needs to be specified if restarting is performed. Then the reference value of plasma density and temperature are defined for the nondimensionalization. The total number of iteration steps and the temporal step size are provided, and the output intervals of the result and the interrupt files are also defined. Finally, the total number of particle species in the simulated plasma and the charge and mass of each species are also provided. For each type of particle, it is defined by the index, density, temperature, and drift velocities. The parameter information of the MCC

method is defined in 'mcc.inp'. The density and temperature of the gas are provided first. Then, the types of collisions between electrons or ions and neutral atoms are determined separately. For each collision type, we can provide a string of numerical code that can be used to read the corresponding cross-section information from the database.

Through the above five input files, users can easily achieve human-machine interaction with the CFIRM package. Different low-temperature plasma problems can be conveniently modeled in the CFIRM package by modifying the parameter values in these input files.

Acknowledgments

The authors would like to thank Wenzhe Zheng and Siyu Wu for their partial contribution to coding and Zhitang Zhong for his partial contribution to testing the code. Jinwei Bai is supported by the China Postdoctoral Science Foundation (No. 2022M710977) and Guangdong Basic and Applied Basic Research Foundations (No. 2022A1515110215). Wei Jiang is supported by National Natural Science Foundation of China (No. 12275095). Yong Cao is supported by Guangdong Basic and Applied Basic Research Foundations (No. 2023A1515010137) and Science, Technology, and Innovation Commission of Shenzhen Municipality (Nos. ZDSYS201707280904031 and GJHZ20220913143010019).

References

[1] M. A. Lieberman and A. J. Lichtenberg. Principles of Plasma Discharges and Materials Processing. John Wiley & Sons, 2005.
[2] I. Adamovich, S. Agarwal, E. Ahedo, and et al. The 2022 plasma roadmap: low temperature plasma science and technology. J. Phys. D: Appl. Phys., 55(37):373001, 2022.
[3] V. M. Donnelly and A. Kornblit. Plasma etching: yesterday, today, and tomorrow. J. Vac. Sci. Technol. A, 31(5):050825, 2013.
[4] F. Taccogna and L. Garrigues. Latest progress in hall thrusters plasma modelling. Rev. Mod. Plasma Phys., 3(12):1–63, 2019.
[5] J. Faure, Y. Glinec, A. Pukhov, S. Kiselev, S. Gordienko, E. Lefebvre, J. P. Rousseau, F. Burgy, and V. Malka. A laserplasma accelerator producing monoenergetic electron beams. Nature, 431(7008):541–544, 2004.
[6] A. Anders. Plasma and ion sources in large area coating: A review. Surf. Coat. Tech., 200(5):1893–1906, 2005.
[7] S. Li, Y. Liu, C. Liu, and Y. Fang. Numerical study of the effect of kinetic damping on resistive wall modes with plasma toroidal rotation in CFETR. Phys. Plasmas, 29(4):042109, 2022.
[8] K. Hara. Non-oscillatory quasineutral fluid model of cross-field discharge plasmas. Phys. Plasmas, 25(12):123508, 2018.
[9] R. E. Heath, I. M. Gamba, P. J. Morrison, and C. Michler. A discontinuous Galerkin method for the Vlasov-Poisson system. J. Comput. Phys., 231(4):1140–1174, 2012.
[10] H. Liu, F. Shi, J. Wan, X. He, and Y. Cao. Discrete unified gas kinetic scheme for a reformulated BGK-Vlasov-Poisson system in all electrostatic plasma regimes. Comput. Phys. Commun., 255:107400, 2020.
[11] C.K. Birdsall. Particle-in-cell charged-particle simulations, plus Monte Carlo collisions with neutral atoms, PIC-MCC. IEEE Trans. Plasma Sci., 19(2):65–85, 1991.
[12] C. K. Birdsall and A. B. Langdon. Plasma Physics via Computer Simulation (Series in Plasma Physics). Institute of Physisc Publishing, 1991.
[13] J. Wang, J. Anderson, and J. Polk. Three-dimensional particle simulations of ion optics plasma flow. In 34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. AIAA, 1998-3799.
[14] Y. Fu, J. Yang, H. Mou, R. Tan, X. Xia, and Z. Gao. Integrative simulation of a 2 cm electron cyclotron resonance ion source with full particle-in-cell method. Comput. Phys. Commun., 278:108395, 2022.
[15] H. Li and A. Sun. Issues in the numerical modeling of positive ion extraction. Comput. Phys. Commun., 259:107629, 2021.
[16] R. Pan, J. Ren, H. Tang, S. Cao, and J. Cao. Application of the view factor model on the particle-in-cell and Monte Carlo collision code. Phys. Rev. E, 102(3):033311, 2020.

[17] B. I. Cohen, A. B. Langdon, and A. Friedman. Implicit time integration for plasma simulation. J. Comput. Phys., 46:15–38, 1982.
[18] A. B. Langdon, B. I. Cohen, and A. Friedman. Direct implicit large time-step particle simulation of plasmas. J. Comput. Phys., 51:107–138, 1983.
[19] A. Friedman. A second-order implicit particle mover with adjustable damping. J. Comput. Phys., 90:292–312, 1990.
[20] H. Y. Wang, W. Jiang, and Y. N. Wang. Implicit and electrostatic particle-in-cell/Monte Carlo model in two-dimensional and axisymmetric geometry: I. Analysis of numerical techniques. Plasma Sources Sci. Technol., 19(4):045023, 2010.
[21] S. Mattei, K. Nishida, M. Onai, J. Lettry, M. Q. Tran, and A. Hatayama. A fully-implicit Particle-In-Cell Monte Carlo collision code for the simulation of inductively coupled plasmas. J. Comput. Phys., 350:891–906, 2017.
[22] J. Bai, Y. Cao, X. He, and E. Peng. An implicit particle-in-cell model based on anisotropic immersed-finite-element method. Comput. Phys. Commun., 261(6):107655, 2020.
[23] Z. Li. The immersed interface method using a finite element formulation. Appl. Numer. Math., 27(3):253–267, 1997.
[24] R. E. Ewing, Z. Li, T. Lin, and Y. Lin. The immersed finite volume element methods for the elliptic interface problems. Math. Comput. Simulation, 50(1-4):63–76, 1999.
[25] S. Adjerid, I. Babuška, R. Guo, and T. Lin. An enriched immersed finite element method for interface problems with nonhomogeneous jump conditions. Comput. Methods Appl. Mech. Engrg., 404:#115770, 2023.
[26] S. Adjerid, T. Lin, and Q. Zhuang. Error estimates for an immersed finite element method for second order hyperbolic equations in inhomogeneous media. J. Sci. Comput., 84(2):1–25, 2020.
[27] T. Lin and X. Zhang. Linear and bilinear immersed finite elements for planar elasticity interface problems. J. Comput. Appl. Math., 236(18):4681–4699, 2012.
[28] T. Lin, Y. Lin, and X. Zhang. Partially penalized immersed finite element methods for elliptic interface problems. SIAM J. Numer. Anal., 53(2):1121–1144, 2015.
[29] R. Guo. Solving parabolic moving interface problems with dynamical immersed spaces on unfitted meshes: Fully discrete analysis. SIAM J. Numer. Anal., 59(2):797–828, 2021.
[30] R. Guo and T. Lin. A higher degree immersed finite element method based on a Cauchy extension for elliptic interface problems. SIAM J. Numer. Anal., 57(4):1545–1573, 2019.
[31] R. Guo and X. Zhang. Solving three-dimensional interface problems with immersed finite elements: A-priori error analysis. J. Comput. Phys., 441:#110445, 2021.
[32] C. He, S. Zhang, and X. Zhang. Error analysis of Petrov-Galerkin immersed finite element methods. Comput. Methods Appl. Mech. Engrg., 404:#115744, 2023.
[33] H. Ji, F. Wang, J. Chen, and Z. Li. A new parameter free partially penalized immersed finite element and the optimal convergence analysis. Numer. Math., 150:1035–1086, 2022.
[34] X.-M. He, T. Lin, Y. Lin, and X. Zhang. Immersed finite element methods for parabolic equations with moving interface. Numer. Methods Partial Differential Equations, 29(2):619–646, 2013.
[35] Y. Chu, Y. Cao, X.-M. He, and M. Luo. Asymptotic boundary conditions with immersed finite elements for interface magnetostatic/electrostatic field problems with open boundary. Comput. Phys. Commun., 182(11):2331–2338, 2011.
[36] Y. Cao, Y. Chu, X. Zhang, and X. Zhang. Immersed finite element methods for unbounded interface problems with periodic structures. J. Comput. Appl. Math., 307(1):72–81, 2016.
[37] C. Lu, Z. Yang, J. Bai, Y. Cao, and X.-M. He. Three-dimensional immersed finite element method for anisotropic magnetostatic/electrostatic interface problems with non-homogeneous flux jump. Int. J. Numer. Meth. Eng., 121(10):2107–2127, 2020.
[38] J. Wang, X. Zhang, and Q. Zhuang. An immersed Crouzeix-Raviart finite element method for Navier-Stokes equations with moving interfaces. Int. J. Numer. Anal. Mod., 19:563–586, 2022.
[39] Y. Chen, Z. Deng, and Y. Huang. Recovery-based a posteriori error estimation for elliptic interface problems based on partially penalized immersed finite element methods. Int. J. Numer. Anal. Mod., 19:126–155, 2022.
[40] R. Kafafy. Immersed finite element Particle-In-Cell simulations of ion propulsion. Ph.D. dissertation, Virginia Polytechnic Institute and State University, 2005.
[41] R. Kafafy, T. Lin, Y. Lin, and J. Wang. Three-dimensional immersed finite element methods for electric field simulation in composite materials. Int. J. Numer. Meth. Eng., 64(7):940–972, 2005.
[42] R. Kafafy, J. Wang, and T. Lin. A hybrid-grid immersed-finite-element particle-in-cell simulation model of ion optics plasma dynamics. Dyn. Contin. Discrete Impuls. Syst. Ser. B Appl. Algorithms, 12:1–16, 2005.

[43] Y. Cao, Y. Chu, X.-M. He, and T. Lin. An iterative immersed finite element method for an electric potential interface problem based on given surface electric quantity. J. Comput. Phys., 281:82–95, 2015.

[44] Y. Chu, D. Han, Y. Cao, X. He, and J. Wang. An immersed-finite-element particle-in-cell simulation tool for plasma surface interaction. Int. J. Numer. Anal. Model., 14(2):175–200, 2017.

[45] H. J. Cao, Y. Cao, Y. C. Chu, X.-M. He, and T. Lin. A huygens immersed-finite-element particle-in-cell method for modeling plasma-surface interactions with moving interface. Commun. Nonlinear Sci. Numer. Simul., 59:132–148, 2018.

[46] D. Han, J. Wang, and X.-M. He. PIFE-PIC: A 3-D parallel immersed finite element particle-in-cell framework for plasma simulations, #AIAA-2018-2196. Proceeding of 2018 AIAA Aerospace Sciences Meeting, Kissimmee, Florida, January 8-12, 2018.

[47] H. Cao, Y. Chu, E. Wang, Y. Cao, and G. Xia. Numerical simulation study on barrel erosion of ion thruster accelerator grid. J. Propul. Power, 31(6):1785–1792, 2015.

[48] H. Cao, Q. Li, K. Shan, Y. Cao, and L. Zheng. Effect of preionization on the erosion of the discharge channel wall in a hall thruster using a kinetic simulation. IEEE Trans. Plasma Sci., 43(1):130–140, 2015.

[49] H. Jian, Y. Chu, H. Cao, Y. Cao, X.-M. He, and G. Xia. Three-dimensional IFE-PIC numerical simulation of background pressure's effect on accelerator grid impingement current for ion optics. Vacuum, 116:130–138, 2015.

[50] C. Lu, J. Wan, Y. Cao, and X. He. A fully decoupled iterative method with three-dimensional anisotropic immersed finite elements for kaufman-type discharge problems. Comput. Method. Appl. M., 372:113345, 2020.

[51] C. Lu, Z. Yang, J. Bai, Y. Cao, and X. He. Three-dimensional immersed finite-element method for anisotropic magnetostatic/electrostatic interface problems with nonhomogeneous flux jump. Int. J. Numer. Meth. Eng., 121(10):2107–2127, 2020.

[52] L. Quan, Y. Cao, Y. Li, H. Liu, and B. Tian. Influence of the axial oscillations on the electron cyclotron drift instability and electron transport in Hall thrusters. Phys. Plasmas, 30(4):043510, 2023.

[53] Y. Han, G. Xia, C. Lu, and X. He. Trilinear immersed-finite-element method for three-dimensional anisotropic interface problems in plasma thrusters. AIAA J., 2023.

[54] J. Bai, Y. Cao, X.-M. He, H. Liu, and X. Yang. Modeling and an immersed finite element method for an interface wave equation. Comput. Math. Appl., 76(7):1625–1638, 2018.

[55] J. Bai, Y. Cao, Y. Li, K. Wang, B. Tian, and Y. Hu. Numerical study of the radio-frequency biased accelerating system in ion thrusters. Plasma Sci. Technol., 25(8):085502, 2023.

[56] C. Lu, J. Wan, Y. Cao, and X.-M. He. A fully decoupled iterative method with three-dimensional anisotropic immersed finite elements for Kaufman-type discharge problems. Comput. Meth. Appl. Mech. Eng., 372:#113345, 2020.

[57] D. Depew, D. Han, J. Wang, X.-M. He, and T. Lin. Immersed-Finite-Element Particle-In-Cell simulations of lunar surface charging, #199. Proceedings of the 13th Spacecraft Charging Technology Conference, Pasadena, California, June 23-27, 2014.

[58] D. Han, P. Wang, X.-M. He, T. Lin, and J. Wang. A 3D immersed finite element method with non-homogeneous interface flux jump for applications in particle-in-cell simulations of plasma-lunar surface interactions. J. Comput. Phys., 321:965–980, 2016.

[59] D. Han, J. Wang, and X.-M. He. A non-homogeneous immersed-finite-element particle-in-cell method for modeling dielectric surface charging in plasmas. IEEE Trans. Plasma Sci., 44(8):1326–1332, 2016.

[60] D. Han, J. Wang, and X.-M. He. Immersed-finite-element particle-in-cell simulations of plasma charging at lunar terminator. J. Spacecr. Rockets, 55(6):1490–1497, 2018.

[61] D. Han, X. He, D. Lund, and X. Zhang. PIFE-PIC: parallel immersed finite element particle-in-cell for 3-D kinetic simulations of plasma-material interactions. SIAM J. Sci. Comput., 43(3):C235–C257, 2021.

[62] J. Zhao, X. Wei, X. Du, X. He, and D. Han. Photoelectron sheath and plasma charging on the lunar surface: semianalytic solutions and fully-kinetic particle-in-cell simulations. IEEE Trans. Plasma Sci., 49(10):3036–3050, 2021.

[63] D. Lund, X. He, X. Zhang, and D. Han. Weak scaling of the parallel immersed-finite-element particle-in-cell (PIFE-PIC) framework with lunar plasma charging simulations. Comput. Part. Mech., 9(6):1279–1291, 2022.

[64] J. Wang, X.-M. He, and Y. Cao. Modeling spacecraft charging and charged dust particle interactions on lunar surface. Proceedings of the 10th Spacecraft Charging Technology Conference, Biarritz, France, 2007.

[65] D. Lund, X.-M. He, X. Zhang, and D. Han. Weak scaling of the parallel immersed finite element particle-in-cell (PIFE-PIC) framework with lunar plasma charging simulations. Comput. Part. Mech., 9:1279–1291, 2022.
[66] D. Lund, X.-M. He, and D. Han. Charging of irregularly-shaped dust grains near surfaces in space, #AIAA 2023-2616. AIAA SciTech 2023 Forum, National Harbor, Maryland & Virtual Conference, January 23-27, 2023.
[67] D. Lund, X.-M. He, and D. Han. Kinetic particle simulations of plasma charging at lunar craters under severe conditions. J. Spacecraft Rockets, 60(4):1176–1187, 2023.
[68] A. Naga and Z. Zhang. The polynomial-preserving recovery for higher order finite element methods in 2D and 3D. Discrete Contin. Dyn. Syst. - Ser. B, 5(3):769–798, 2005.
[69] L. Song and Z. Zhang. Polynomial preserving recovery of an over-penalized symmetric interior penalty Galerkin method for elliptic problems. Discrete Contin. Dyn. Syst. - Ser. B, 20(5):1405–1426, 2015.
[70] H. Guo, Z. Zhang, R. Zhao, and Q. Zou. Polynomial preserving recovery on boundary. J. Comput. Appl. Math., 307:119–133, 2016.
[71] H. Guo and X. Yang. Gradient recovery for elliptic interface problem: II. Immersed finite element methods. J. Comput. Phys., 338:606–619, 2017.
[72] R. Zhao, W. Du, F. Shi, and Y. Cao. Recovery based finite difference scheme on unstructured mesh. Appl. Math. Lett., 129:107935, 2022.
[73] J. Bai, Y. Cao, Y. Chu, and X. Zhang. An improved immersed finite element particle-in-cell method for plasma simulation. Comput. Math. Appl., 75(6):1887–1899, 2018.
[74] T. Lin, Y. Lin, R. C. Rogers, and L. M. Ryan. A rectangular immersed finite element method for interface problems. In P. Minev and Y. Lin, editors, Advances in Computation: Theory and Practice, Vol. 7, pages 107–114. Nova Science Publishers, Inc., 2001.
[75] X.-M. He. Bilinear immersed finite elements for interface problems. Ph.D. Dissertation, Virginia Polytechnic Institute and State University, 2009.
[76] X.-M. He, T. Lin, and Y. Lin. Approximation capability of a bilinear immersed finite element space. Numer. Methods Partial Differential Equations, 24(5):1265–1300, 2008.
[77] X.-M. He, T. Lin, and Y. Lin. A bilinear immersed finite volume element method for the diffusion equation with discontinuous coefficients, dedicated to Richard E. Ewing on the occasion of his 60th birthday. Commun. Comput. Phys., 6(1):185–202, 2009.
[78] X.-M. He, T. Lin, and Y. Lin. Interior penalty bilinear IFE discontinuous Galerkin methods for elliptic equations with discontinuous coefficient, dedicated to David Russell's 70th birthday. J. Syst. Sci. Complex., 23(3):467–483, 2010.
[79] X.-M. He, T. Lin, and Y. Lin. The convergence of the bilinear and linear immersed finite element solutions to interface problems. Numer. Methods Partial Differential Equations, 28(1):312–330, 2012.
[80] X.-M. He, T. Lin, and Y. Lin. A selective immersed discontinuous Galerkin method for elliptic interface problems. Math. Methods Appl. Sci., 37(7):983–1002, 2014.
[81] X.-M. He, T. Lin, and Y. Lin. Immersed finite element methods for elliptic interface problems with non-homogeneous jump conditions. Int. J. Numer. Anal. Model., 8(2):284–301, 2011.
[82] W. Feng, X.-M. He, Y. Lin, and X. Zhang. Immersed finite element method for interface problems with algebraic multigrid solver. Commun. Comput. Phys., 15(4):1045–1067, 2014.
[83] M. J. Kushner. Mechanisms for power deposition in Ar/SiH4 capacitively coupled RF discharges. IEEE Trans. Plasma Sci., 14(2):188–196, 1986.
[84] D. Vender and R.W. Boswell. Numerical modeling of low-pressure RF plasmas. IEEE Trans. Plasma Sci., 18(4):725–732, 1990.
[85] V. Vahedi and M. Surendra. A Monte Carlo collision model for the particle-in-cell method: applications to argon and oxygen discharges. Comput. Phys. Commun., 87:179–198, 1995.
[86] K. Nanbu, T. Morimoto, and M. Suetani. Direct simulation Monte Carlo analysis of flows and etch rate in an inductively coupled plasma reactor. IEEE Trans. Plasma Sci., 27(5):1379–1388, 1999.
[87] K. Nanbu. Probability theory of electron-molecule, ion-molecule, molecule-molecule, and coulomb collisions for particle modeling of materials processing plasmas and cases. IEEE Trans. Plasma Sci., 28(3):971–990, 2000.
[88] G. Lapenta. Particle rezoning for multidimensional kinetic particle-in-cell simulations. J. Comput. Phys., 181(1):317–337, 2002.
[89] W. Villafana, F. Petronio, A. C. Denig, M. J. Jimenez, D. Eremin, L. Garrigues, and O. Vermorel. 2D radial-azimuthal particle-in-cell benchmark for $\mathbf{E} \times \mathbf{B}$ discharges. Plasma Sources. Sci. Technol., 30(7):075002, 2021.
[90] T. Charoy, J. P. Boeuf, A. Bourdon, J. A. Carlsson, P. Chabert, B. Cuenot, and W. Villafana. 2D axial-azimuthal particle-in-cell benchmark for low-temperature partially magnetized plasmas. Plasma Sources. Sci. Technol., 28(10):105010, 2019.

School of Science, Harbin Institute of Technology(Shenzhen), Shenzhen, Guangdong 518055, P. R. China; Center for Advanced Material Diagnostic Technology, College of Engineering Physics, Shenzhen Technology University, Shenzhen 518118, P. R. China
E-mail: baijinwei@stu.hit.edu.cn and baijinwei@sztu.edu.cn

Department of Mathematics, Center for Mathematical Plasma Astrophysics, KU Leuven, Leuven, 3001, Belgium
E-mail: hongtao.liu@kuleuven.be

Department of Mathematics & Statistics, Missouri University of Science & Technology, Rolla, MO 65401, USA
E-mail: hex@mst.edu

School of Physics, Huazhong University of Science and Technology, Wuhan, Hubei 430074, P. R. China
E-mail: weijiang@hust.edu.cn

Department of Mechanical Engineering & Automation, Harbin Institute of Technology(Shenzhen), Shenzhen, Guangdong 518055, P. R. China
E-mail: yongc@hit.edu.cn