Implicit electrostatic particle-in-cell/Monte Carlo simulation for the magnetized plasma: Algorithms and application in gas-inductive breakdown*

Wang Hong-Yu(王虹宇)[a)†], Sun Peng(孙 鹏)[a)], Jiang Wei(姜 巍)[b)], Zhou Jie(周 杰)[c)], and Xie Bai-Song(谢柏松)[d)]

[a)] School of Physics Science and Technology, Anshan Normal University, Anshan 114005, China
[b)] School of Physics, Huazhong University of Science and Technology, Wuhan 430074, China
[c)] Southwest Institute of Technical Physics, Chengdu 610041, China
[d)] College of Nuclear Science and Technology, Beijing Normal University, Beijing 100875, China

(Received 28 October 2014; revised manuscript received 23 January 2015; published online 10 April 2015)

An implicit electrostatic particle-in-cell/Monte Carlo (PIC/MC) algorithm is developed for the magnetized discharging device simulation. The inductive driving force can be considered. The direct implicit PIC algorithm (DIPIC) and energy conservation scheme are applied together and the grid heating can be eliminated in most cases. A tensor-susceptibility Poisson equation is constructed. Its discrete form is made up by a hybrid scheme in one-dimensional (1D) and two-dimensional (2D) cylindrical systems. A semi-coarsening multigrid method is used to solve the discrete system. The algorithm is applied to simulate the cylindrical magnetized target fusion (MTF) pre-ionization process and get qualitatively correct results. The potential application of the algorithm is discussed briefly.

Keywords: particle-in-cell/Monte Carlo, implicit simulation, discharging simulation

PACS: 52.80.Pi, 52.27.Aj, 52.65.Rr **DOI:** 10.1088/1674-1056/24/6/065207

1. Introduction

The magnetized discharging is a very common technology in the plasma device. In a typical scheme, the magnetic field can change the particle trajectory and confine the transverse transport. Therefore, the magnetizing can often increase the plasma densities and cause better performance of plasma sources.[1–4] However, in a few cases of plasma sources with stochastic heating, the magnetic field can reduce the heating rates and decrease the source performances. Therefore, the magnetic field parameters and configurations must be designed carefully.

The theoretical analysis of the magnetized discharging must work with numerical simulations. In the cases with low gas pressure, the most powerful simulating tool is the particle-in-cell/Monte Carlo (PIC/MC) technique.[5–13] In typical cases, this tool can trace the dominant physics process in the plasma and can be used to analyze the discharge mechanism. In the one-dimensional (1D) problems dominated by stochastic heating, the PIC/MC simulating works very well.

The magnetized discharges in nature are two dimensional (2D) or three-dimensional (3D) problems. Therefore, the simulating should be executed in two dimensions or three dimensions. However, multi-dimensional PIC/MC simulating costs plenty of computational resources. In fact, even 2D PIC/MC simulating with industrial size discharging has to use parallel computers, because the PIC/MC simulating must satisfy the constraints as[14]

$$\begin{aligned} \Delta t &< \frac{2}{\omega_{\mathrm{p}}}, \\ \Delta t &< \frac{\Delta x}{c}, \\ \Delta x &< \xi \lambda_{\mathrm{D}} \end{aligned} \tag{1}$$

In many of the cases, these constraints cause small space and time steps unpractically. As mentioned above, the plasma density can be increased in the magnetized discharge so that the simulation becomes more difficult. Some authors introduced implicit PIC algorithms to overcome this problem. Most of these implicit algorithms for magnetized plasma are based on an electromagnetic field solver.[15–17] However, it is very complex and difficult to apply these implicit electromagnetic PIC algorithms in the typical discharging simulation. The first problem is that the implicit PIC algorithms introduce (pseudo) dielectric and susceptibility functions in Maxwell equations. These functions cause failure of both the explicit finite difference time domain (FDTD) solver and the non-iterative ADI solver. One must develop new PDE solvers or introduce some reductions. The divergence equation (Gauss theorem of E and B) cannot be neglected in implicit electromagnetic PIC models and it causes more complexities. In addition, in many of these discharge problems, external circuit model and charge

*Project supported by the National Natural Science Foundation of China (Grant Nos. 11275007, 11105057, 11175023, and 11275039). One of the author (Wang H Y) is supported by Program for Liaoning Excellent Talents in University (Grant No. LJQ2012098).

†Corresponding author. E-mail: why@btitgroup.com

© 2015 Chinese Physical Society and IOP Publishing Ltd

deposition effects should be modeled. The coupling of discharge solver and circuit solver is also more complex in an electromagnetic model than that in an electrostatic model because the electrostatic model is based on potential (voltage). Moreover, the Monte Carlo procedure must be merged into the algorithms to describe the collisions, which causes some other complexities in electromagnetic PIC methods, as the increasing noises introduced by MC. All these problems cause the electromagnetic PIC/MC discharging simulating more difficulties than electrostatic cases. By now, the implicit electromagnetic PIC/MC simulating for the gas discharging problem is still an open problem. On the other hand, many cases of direct-current or radio-frequency (DC/RF) discharges are essentially electrostatic problems. In these cases, the electromagnetic wave affects the physics little. For example, the electromagnetic wave effects in a common 13.56 MHz RF-driven capacity coupled plasma can be neglected and the electrostatic solver is enough.[18,19] Theoretically, magnetized capacity coupled plasma (CCP),[1] magnetic-biased inductive coupled plasma (ICP),[20] magnetron,[5] and the pre-ionization of magnetized target fusion (MTF)[21] are of these cases and can be considered as electrostatic plasmas plus driving fields. Besides, we can expect that electrostatic implicit algorithms can give a better simulating speed and convergency in these cases because this algorithm can get rid of instabilities relative to electromagnetic waves. So an electrostatic implicit particle-in-cell/Monte Carlo simulating algorithm is useful for these problems.

In this paper, we discuss an implicit and energy-conservation electrostatic PIC/MC algorithm. In our previous work,[22] we showed that the PIC/MC code merged with implicit and energy-conservation algorithms could get rid of the grid heating even if $DX \gg \lambda_{\mathrm{D}}$. Then we generalized the combined algorithm to magnetized plasmas. The generation includes developing the differential form and solver of the new field equations, developing the two-body collision algorithms, revising particle pushing codes for magnetized plasmas. The parallelization of this algorithm and the particle rezoning technologies are also needed. An example of gas-inductive breakdown simulating is also shown briefly.

This paper is organized as follows. In Section 2, we describe the skeleton framework of this implicit PIC algorithm. The dielectric potential equation and particle pushing formula is generated. The parallelization is also discussed in this section. In Section 3, the finite difference form of the field equation is discussed and the iteration solver of the linear system is shown. In Section 4, miscellaneous parts of the algorithm as Monte–Carlo coupling and inductive coupling driven forces modeling are described. In the final section, we show basic simulation results for the magnetized target fusion pre-ionize (MTF-PI) stage and discuss some other potential applications of the algorithm.

2. The algorithm framework

Our major algorithm framework is similar to Welch's[23] except that our plasma is electrostatically-modeled. In the non-relativistic condition, we can write the particle mover as

$$\begin{aligned} \boldsymbol{v}^{n+1/2} &= \boldsymbol{v}^{n-1/2} + \Delta t[\boldsymbol{a}^n + (\boldsymbol{v}^{n-1/2} + \boldsymbol{v}^{n+1/2}) \\ &\quad \times q\boldsymbol{B}^n(\mathbf{x}_n)/(2m)] + \frac{q}{m}\boldsymbol{E}_{\mathrm{T}}{}^n\Delta t, \end{aligned} \tag{2}$$

where $\boldsymbol{a}^n = q\boldsymbol{E}^n/m$ and we introduce field $\boldsymbol{E}_{\mathrm{T}}$ to note the non-potential field (i.e, induced electric field or ponderomotive force).

By introducing a tensor $\vec{\vec{T}}$, this mover can be written to

$$\boldsymbol{v}^{n+1/2} = \langle \vec{\vec{T}} \rangle \cdot \boldsymbol{S}$$

where

$$\langle \vec{\vec{T}} \rangle = \frac{1}{1+\Omega^2} \begin{bmatrix} 1+\Omega_1^2 & \Omega_1\Omega_2+\Omega_3 & \Omega_1\Omega_3-\Omega_2 \\ \Omega_1\Omega_2-\Omega_3 & 1+\Omega_2^2 & \Omega_2\Omega_3+\Omega_1 \\ \Omega_1\Omega_3+\Omega_2 & \Omega_2\Omega_3-\Omega_1 & 1+\Omega_3^2 \end{bmatrix} \tag{3}$$

with $\boldsymbol{\Omega} = q\boldsymbol{B}\Delta t/2m$ and the vector $\boldsymbol{S}$ is the pushing vector. The algorithm can be summarized by a standard DIPIC template:

let $\boldsymbol{a}_n = \frac{1}{2}(\bar{\boldsymbol{A}}^{n-1} + \boldsymbol{a}^{n+1})$ and $\boldsymbol{S} = S_1 + S_2$:

$$\begin{aligned} \boldsymbol{S}_1 &= \boldsymbol{v}^{n-1/2} + \frac{1}{2}\bar{\boldsymbol{A}}^{n-1}\Delta t + \boldsymbol{v}^{n-1/2} \times \Omega + \frac{q}{m}\boldsymbol{E}_{\mathrm{T}}{}^n\Delta t, \\ \boldsymbol{S}_2 &= \frac{q}{m}\boldsymbol{E}^{n+1}(x^{n+1})\Delta t/2. \end{aligned} \tag{4}$$

Then

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \langle \vec{\vec{T}} \rangle \cdot (\boldsymbol{S}_1 + S_2)\Delta t = \tilde{\boldsymbol{x}}^{n+1} + \delta\boldsymbol{x}^{n+1},$$

where $\delta\boldsymbol{x}^{n+1} = \langle \vec{\vec{T}} \rangle \cdot \boldsymbol{S}_2\Delta t$.

Considering the relation between charge density and dipole moment

$$\delta\rho = -\nabla\cdot\boldsymbol{P} = -\nabla\cdot\frac{\sum q_i\delta\boldsymbol{x}_i}{\Delta V} = \nabla\cdot\sum_s \rho_s\delta\boldsymbol{x}_s,$$

where $\sum_s$ denotes summing over all of the particles species, one can get

$$\rho^{n+1} = \tilde{\rho}^{n+1} - \nabla\cdot P \simeq \tilde{\rho}^{n+1} - \nabla\cdot\left(\sum_s \tilde{\rho}_s^{n+1}\delta\boldsymbol{x}_s^{n+1}\right).$$

Put the expression of $\delta\boldsymbol{x}$ into the Poisson equation, the electric field at the next time step $\boldsymbol{E}^{n+1}$ can be worked out by

$$\begin{aligned} &\nabla\cdot\left[\varepsilon_0(1+\vec{\vec{\chi}})\cdot\nabla\phi\right] = \nabla\cdot\left(\vec{\vec{\varepsilon}}\cdot\nabla\phi\right) = -\tilde{\rho}^{n+1}, \\ &\vec{\vec{\chi}} = \left[\langle T\rangle\sum_s \frac{q_s\rho_s\Delta t^2}{2\varepsilon_0 m_s}\right]. \end{aligned} \tag{5}$$

After solving the Poisson equation, the acceleration $\boldsymbol{a}^{n+1}$ and the δx^{n+1} can be figured out.

Finally, the algorithm can be written as follows:

(i) "First push" step. In this step, the particle will be pushed to intermediate position and velocity:

$$\begin{aligned}
&\boldsymbol{S}_1 = \boldsymbol{v}^{n-1/2} + \frac{1}{2}\bar{\boldsymbol{A}}^{n-1}\Delta t + \frac{q}{m}\boldsymbol{E}_{\mathrm{T}}{}^{n}\Delta t + \boldsymbol{v}^{n-1/2}\times\boldsymbol{\Omega}^{n},\\
&\tilde{\boldsymbol{v}}^{n+1/2} = \langle T\rangle\cdot\boldsymbol{S}_1,\\
&\tilde{\boldsymbol{x}}^{n+1} = \boldsymbol{x}^{n} + \tilde{\boldsymbol{v}}^{n+1/2}\Delta t.
\end{aligned} \tag{6}$$

(ii) "Field solver" step. In this step, the charge density will be accumulated by a 2-order weighting scheme and the susceptibility tensor $\vec{\vec{\chi}}$ is generated:

$$\tilde{\rho}(x_g) = \sum_{\mathrm{p}} q_{\mathrm{p}} S_2(x_g - \tilde{x}_{\mathrm{p}}). \tag{7}$$

Then the Poisson equation (5) will be solved.

(iii) "Final push" step. In this step, the electric field will be interpolated by 1-order scheme:

$$\boldsymbol{E}(x_{\mathrm{p}}) = \sum_{g} \boldsymbol{E}_g S_1(x_{\mathrm{p}} - x_{g+1/2}), \tag{8}$$

where $x_{g+1/2} = \frac{1}{2}\{x_g + x_{g+1}\}$.

The present PIC algorithms need the 2-order weighting plus 1-order interpolation scheme (2–1) to get rid of the grid heating. If the normal 1–1 scheme is applied instead, the grid heating will take place while $DX \gg \lambda_D$ and dt is small. In pure PIC simulation, one can let dt increase following DX to damp the grid heating. However, in PIC/MC code for discharge simulation, the increased damping causes too large an error and cannot be applied in many cases. The detail of the error is discussed in Ref. [22].

Then the second part of the particle mover will be executed:

$$\begin{aligned}
&\boldsymbol{a}^{n+1} = \frac{q}{m}\boldsymbol{E}^{n+1}(\tilde{x}^{n+1}),\\
&\boldsymbol{S}_2 = \frac{q}{m}E^{n+1}(x^{n+1})\Delta t/2,\\
&\delta\tilde{\boldsymbol{v}}^{n+1/2} = \langle T\rangle\cdot\boldsymbol{S}_2,\\
&\boldsymbol{v}^{n+1/2} = \tilde{\boldsymbol{v}}^{n+1/2} + \delta\tilde{\boldsymbol{v}}^{n+1/2},\\
&\boldsymbol{x}^{n+1} = \tilde{\boldsymbol{x}}^{n+1} + \delta\tilde{\boldsymbol{v}}^{n+1/2}\Delta t,\\
&\bar{\boldsymbol{A}}^{n} = \frac{\theta}{2}\boldsymbol{a}^{n+1} + (1-\frac{\theta}{2})\bar{\boldsymbol{a}}^{n-1},\\
&\bar{\boldsymbol{a}}^{n} = \left(1-\frac{\theta}{2}\right)\boldsymbol{a}^{n+1} + \frac{\theta}{2}\bar{\boldsymbol{a}}^{n-1}.
\end{aligned} \tag{9}$$

(iv) "MCC step". In this step, the Monte Carlo collisions are executed.

The particle movers and the MCC step can be parallelized easily. In our current version, the parallelized code is formed by a broadcast/reduce scheme.[24] That is, all of the processes share the field information and every process works on some particles. After the pushing step, every process weights its own particles and broadcast/reduce the ρ data of each other. Then the Poisson equation is solved serially and the solution is broadcast in the whole processes set. Finally, the final push and MCC step will be executed by processes on each of the owned particles. Anyway, the revising to put code to domain decomposition mode will not affect the movers and MCC procedures.

3. The field solver

In the magnetized implicit electrostatic particle-in-cell algorithms, the Poisson equation (5) must be solved at every step. As the above description, the pushing step and MCC step are the same as the standard PIC scheme in essence. Then the only difference appears in the Poisson equation, discrete and solving.

3.1. The discrete form

Equation (5) can be discreted with the finite volume method. The integral form of the equation is

$$\begin{aligned}
&\oint\{\vec{\vec{\varepsilon}}\cdot\boldsymbol{E}\}\cdot\boldsymbol{n}\,\mathrm{d}S = \int\rho\,\mathrm{d}zV,\\
&\vec{\vec{\varepsilon}} = \vec{\vec{I}}\,I + \vec{\vec{\chi}}\,.
\end{aligned} \tag{10}$$

In a 1D uniform grid, the discrete form is very simple.[25] In this paper, we discuss the 2D r–z cylindrical system with $E_\phi = D_\phi = 0$ and $B_\phi = 0$. This is the most common case in discharging device simulation. Introduce the $\boldsymbol{D} = \vec{\vec{I}}\,\varepsilon\cdot\boldsymbol{E}$ and set the grid to $r = \{0, \Delta r, 2\Delta r, \ldots, j\Delta r, \ldots\}$, $z = \{0, \Delta z, \ldots, i\Delta z, \ldots\}$, then the discrete form can be worked out on a ring with $r_{j-1/2} < r < r_{j+1/2}, z_{i-1/2} < z < z_{i+1/2}$:

$$\begin{aligned}
&-2\pi r_{j-1/2}\Delta z D^r_{i,j-1/2} + 2\pi r_{j+1/2}\Delta z D^r_{i,j+1/2}\\
&-2\pi r_j\Delta r D^r_{i-1/2,j} + 2\pi r_j\Delta r D^r_{i+1/2,j} = 2\pi r_j\Delta r\Delta z\rho_{i,j}.
\end{aligned}$$

In this equation, the half-point value of the electric field and dielectric tensor must be made up. The half-point dielectric tensor can be made up by a simple arithmetic average. The half-point electric field needs to be calculated in two forms. First of all, the standard half integer point discrete scheme of potential gives out

$$\begin{aligned}
E^r_{i,j+1/2} &= \frac{\phi_{i,j} - \phi_{i,j+1}}{\Delta r},\\
E^z_{i+1/2,j} &= \frac{\phi_{i,j} - \phi_{i+1,j}}{\Delta z}.
\end{aligned}$$

Secondly, the E^r in z direction half integer points and the E^z in r direction half integer points can be interpolated by the central difference of potential:

$$\begin{aligned}
E^r_{i+1/2,j} &= \frac{1}{2}(E^r_{i,j} + E^r_{i+1,j})\\
&= \frac{1}{2}\left\{\frac{\phi_{i,j-1} - \phi_{i,j+1}}{2\Delta r} + \frac{\phi_{i+1,j-1} - \phi_{i+1,j+1}}{2\Delta r}\right\},\\
E^z_{i,j+1/2} &= \frac{1}{2}(E^r_{i,j} + E^r_{i,j+1})
\end{aligned}$$

$$= \frac{1}{2}\left\{\frac{\phi_{i-1,j}-\phi_{i+1,j}}{2\Delta z}+\frac{\phi_{i-1,j+1}-\phi_{i+1,j+1}}{2\Delta z}\right\}.$$

Put all the ansatz together, we can get a nine-point differential scheme:

$$\begin{aligned}&s_{i,j}\phi_{i-1,j-1}+a_{i,j}\phi_{i-1,j}+t_{i,j}\phi_{i-1,j+1}+d_{i,j}\phi_{i,j-1}\\&+b_{i,j}\phi_{i,j}+e_{i,j}\phi_{i,j+1}+p_{i,j}\phi_{i+1,j-1}\\&+c_{i,j}\phi_{i+1,j}+q_{i,j}\phi_{i+1,j+1}=-\rho_{i,j},\end{aligned}\tag{11}$$

where $s_{i,j}$ to $q_{i,j}$ are nine coefficients and must be calculated at each step.

We should point out when the magnetic field is not very strong, one can apply five-term approximation as Welch's.[15] In this approximation, we can neglect the coefficients s, t, p, and q (anyway, the B field is still considered fully in the particle pusher). Then equation (11) can be solved by some convenient method.[26] The accuracy of this approximation should be checked carefully.

In the Descartes system or the 3D cases, the discrete can be done with similar steps.

3.2. The linear solver

Equation (11) is a linear equation system. In order to make effective and practical simulation for a real discharging system, the solver must have good efficiencies. In addition, the field solver is the major difficulty of parallelization. In the 1D case, the problem is a tridiagonal system and can be solved by a traditional TDMA.

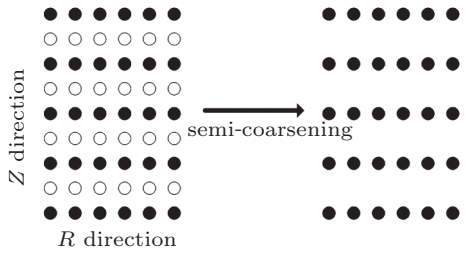


Fig. 1. The concept of semi-coarsening.

In the 2D case, the semi-coarsening multigrid method in Ref. [26] is generalized to solve the equations. This method can work robustly in both Descartes and cylindrical coordinate systems, even in the cases of $L_z \ll L_r$ or $L_r \ll L_z$. Its major concepts can be understood by Fig. 1: Equation (11) can be written as $Au = f$, where the coefficient matrix A is the matrix formed by the coefficients in Eq. (11) and determined by $\vec{\chi}$. Then the initial step of semi-coarsening multigrid solver is

init step :

$$\begin{aligned}&\chi^L = \chi,\\&A^L = A(\chi^L),\\&l = L,\end{aligned}$$

do while $l > 1$,

$$\begin{aligned}&u^l = 0; \chi^{l-1} = I_l^{l-1}\chi^l; A^{l-1} = A(\chi^l),\\&l = l-1,\end{aligned}$$

enddo.

Here the project operator I_l^{l-1} is read as

$$I_{l-1}^l = [1/4\ 1/2\ 1/4]^{\mathrm{T}}.$$

At the solver initialization, the χ in every tier is calculated and the coefficients are made out with χ in every tier. Then the matrix A is assembled in all tiers. After the initialization, a $MG(0,1)$ cycle is executed to solve the linear system:

solving step :

until converge

$f^L = f$,

do $l = L-1, 2$,

$f^{l-1} = I_l^{l-1}(f^l - A^l u^l)$,

enddo

solve $A^1 u^1 = f^1$,

do $l = 2, L$,

$u^l = u^l + I_{l-1}^l u^{l-1}$,

linesmooth(u^l, f^l),

enddo,

loop.

In this algorithm, linesmooth(u^l, f^l) is the line smooth procedure. It means that the equation $AX = b$ is

$$\begin{aligned}&a_{i,j}\phi_{i-1,j}+b_{i,j}\phi_{i,j}+c_{i,j}\phi_{i+1,j}+d_{i,j}\phi_{i,j-1}+e_{i,j}\phi_{i,j+1}\\&+s_{i,j}\phi_{i-1,j-1}+t_{i,j}\phi i-1,j+1+p_{i,j}\phi_{i+1,j-1}\\&+q_{i,j}\phi_{i+1,j+1}=h^2 f_{i,j}.\end{aligned}$$

Then the line smooth worked in r direction: for every i, solving the tridiagonal system

$$\begin{aligned}&\cdots\\&d_{i,j-1}\phi_{i,j-2}^{\mathrm{new}}+b_{i,j-1}\phi_{i,j-1}^{\mathrm{new}}+e_{i,j-1}\phi_{i,j}^{\mathrm{new}}\\&=h^2 f_{i,j-1}-(a_{i,j-1}\phi_{i-1,j-1}^{\mathrm{old}}+c_{i,j-1}\phi_{i+1,j-1}^{\mathrm{old}}+s_{i,j-1}\phi_{i-1,j-2}^{\mathrm{old}}\\&+t_{i,j-1}\phi_{i-1,j}^{\mathrm{old}}+p_{i,j-1}\phi_{i+1,j-2}^{\mathrm{old}}+q_{i,j-1}\phi_{i+1,j}^{\mathrm{old}})\\&d_{i,j}\phi_{i,j-1}^{\mathrm{new}}+b_{i,j}\phi_{i,j}^{\mathrm{new}}+e_{i,j}\phi_{i,j+1}^{\mathrm{new}}\\&=h^2 f_{i,j}-(a_{i,j}\phi_{i-1,j}^{\mathrm{old}}+c_{i,j}\phi_{i+1,j}^{\mathrm{old}}+s_{i,j}\phi_{i-1,j-1}^{\mathrm{old}}\\&+t_{i,j}\phi_{i-1,j+1}^{\mathrm{old}}+p_{i,j}\phi_{i+1,j-1}^{\mathrm{old}}+q_{i,j}\phi_{i+1,j+1}^{\mathrm{old}})\\&d_{i,j+1}\phi_{i,j}^{\mathrm{new}}+b_{i,j+1}\phi_{i,j+1}^{\mathrm{new}}+e_{i,j+1}\phi_{i,j+2}^{\mathrm{new}}\\&=h^2 f_{i,j+1}-(a_{i,j+1}\phi_{i-1,j+1}^{\mathrm{old}}+c_{i,j}\phi_{i+1,j+1}^{\mathrm{old}}+s_{i,j+1}\phi_{i-1,j}^{\mathrm{old}}\\&+t_{i,j+1}\phi_{i-1,j+2}^{\mathrm{old}}+p_{i,j}\phi_{i+1,j}^{\mathrm{old}}+q_{i,j+1}\phi_{i+1,j+2}^{\mathrm{old}})\end{aligned}$$

the i-th line smoothing is to solve the above equation set with i. The whole linesmooth procedure in one tier is to execute all of the N lines smoothing by the red–black order in this tier.

When χ is smooth and not very large, the solver converges after about 10–30 iterations and works well for all grid sizes. If the χ becomes very large (say, $\max(\chi) > 1000$), the solver could break down. Thus we use the solver only in the case of $\omega_p \Delta t < 20$. We must claim that the solver can break down also while Ω becomes too large. Anyway, this problem will not occur in our all discharging simulations.

In the current version, this SMG solver is realized as a serial code, so that every computing node must broadcast and reduce the field information. Anyway, following the idea of pipeline algorithms in Ref. [24], this solver can be parallelized. This work is tedious but straightforward: i) split the simulating region to a series of bands: $0 < r < r_1, r_1 < r < r_2, \ldots$ and dispatch every band to nodes; ii) semi-coarsening the matrix in the z direction for each node synchronously; iii) run the linesmooth procedure. Because the procedure is a series of tridiagonal system solving, it can be worked out by the pipeline parallel algorithm. We will use this algorithm for large device simulation or 3D simulation.

4. MCC, particle rezone, and some other tricks

The MCC step is similar to our previous work.[27] In common cases, the electron versus background gas collision can be modeled by null collision methods. We use the MCC subroutine as previous works except that the particle recombine collisions. Because we do not use the particle sorting and the particle grouped by cell, the recombine procedure must be processed by some trick.

In the current version, the N-body ($N > 1$) collision is executed by cell probability accumulating. For example, the electron–ion recombinations are processed with the following steps: a) calculate the collision probability for all cells (i, j) and get the recombinated charge $q_c(i, j)$ in one step. b) All of the electrons execute a Monte Carlo recombination with a recombining charge data copy of $q_{ce} = q_c$. If one electron in the (i, j) cell with charge $q_e < q_{ce}(i, j)$ are recombined, we can set $q_{ce}(i, j) = q_{ce}(i, j) - q_e$. If the recombined electron charge $q_e > q_{ce}(i, j)$, we set $q_e = q_e - q_{ce}(i, j)$ and $q_{ce}(i, j) = 0$. The electron macro-particle, which is recombined, is deleted from the particle lists and the neutral particle density in this cell is increased by the combined charge data. The running on particles is executed by a wheel-rolling order: the traversing of the particle running from k to N and then 1 to k, where $k = \mathrm{rand}(N)$ and N is the electron macro-particle numbers. c) All of the ions execute the recombination procedure with data copy $q_{ci} = q_c$ except the gas density increasing. The ionized exhausting of the gas is considered in a similar way.

Besides the N-body collision, the particle self-rezoning is considered. The three to two combinations and trivial splitting of macro-particles by Lapenta[28] are applied in our codes. There are two reasons to apply rezoning procedures. The first case is in the breakdown avalanche process, for example, which takes place in the pulsed ICP, where the electron charges increase 6–8 orders and the combination procedure is applied to overcome the memory exhaust. The second case is in axisymmetric simulating, the rezoning procedures are applied to control the noise. In the second case, when a macro-particle runs close to the axis, it can be split to several smaller macro-particles to decrease the stochastic noise. The split will cause the macro-particle numbers to increase continually, therefore we will combine very small macro-particles to control the particle number after some steps. We find the energy–momentum conservation is important in the rezoning, thus the three to two combination is applied.

Finally, the driving field $\boldsymbol{E}^{\mathrm{T}}$ should be calculated independently. In typical problems, the driving fields are inductive electric fields. If the plasma density is very low, we can ignore the plasma self-generated current. Then the inductive electric field can be solved by the applied external magnetic field. On the problems as ICP, the plasma skin effect must be considered. We solved these problems as Takekida did:[29] firstly, the inductive field and the plasma transverse in two dimensions can be written as harmonic form $E^{\mathrm{T}} = E^{\theta} \exp\{\mathrm{i}\omega t\}$, $j^{\mathrm{T}} = j^{\theta} \exp\{\mathrm{i}\omega t\}$ and the inductive field equation is

$$\nabla^2 E^{\theta} + \frac{\omega^2}{c^2} E^{\theta} = \mathrm{i}\omega\mu_0 (j^{\theta} + J^{\mathrm{ext}}), \tag{12}$$

where ω is the RF driving frequency and J^{ext} is the current density amplitude of the external coil. The plasma current density j^{θ} is the average in one RF period:

$$J^{\theta} = \langle \sum_i \frac{1}{V_g} \sum_j q_j v_j^{\theta} \rangle_{\mathrm{T}}.$$

Then equation (12) needs to be solved once per RF cycle.

5. Application and discussion

We have used this algorithm to simulate the MTF pre-ionize procedure. In this device, a power inductive RF pulse applied on a cylinder gas tube with an external axis-direction magnetic field (biasing field) existing. In the pre-ionize stage, the plasma self-generated magnetic field can be ignored, then the electrostatic field plus the driving field model can be applied. The detailed discussion and the physical problem will be shown in another paper. Here we only describe the running features. The discharging cube is a cylinder column with gas filled. The driving field is an inductive field from a solenoid coil with sinusoidal current. The biasing field is a uniform axis-direction static magnetic field. At the beginning of the

simulation, a seed plasma with density about $10^{14}/m^3$ is put in the cube. When the driving field is applied, the discharge occurs.

The simulation is executed with grid spacing $DX \sim 0.1$ mm and 4–8 CPU processes. Under our parameters, the breakdown takes place in dozens of nanoseconds. All the simulation worked on about 4 μms with several days. After about 100 ns, a torus profile plasma is built up and the maximum plasma densities become larger than $10^{19}/m^3$. The typical discharge profiles are shown in Fig. 2.

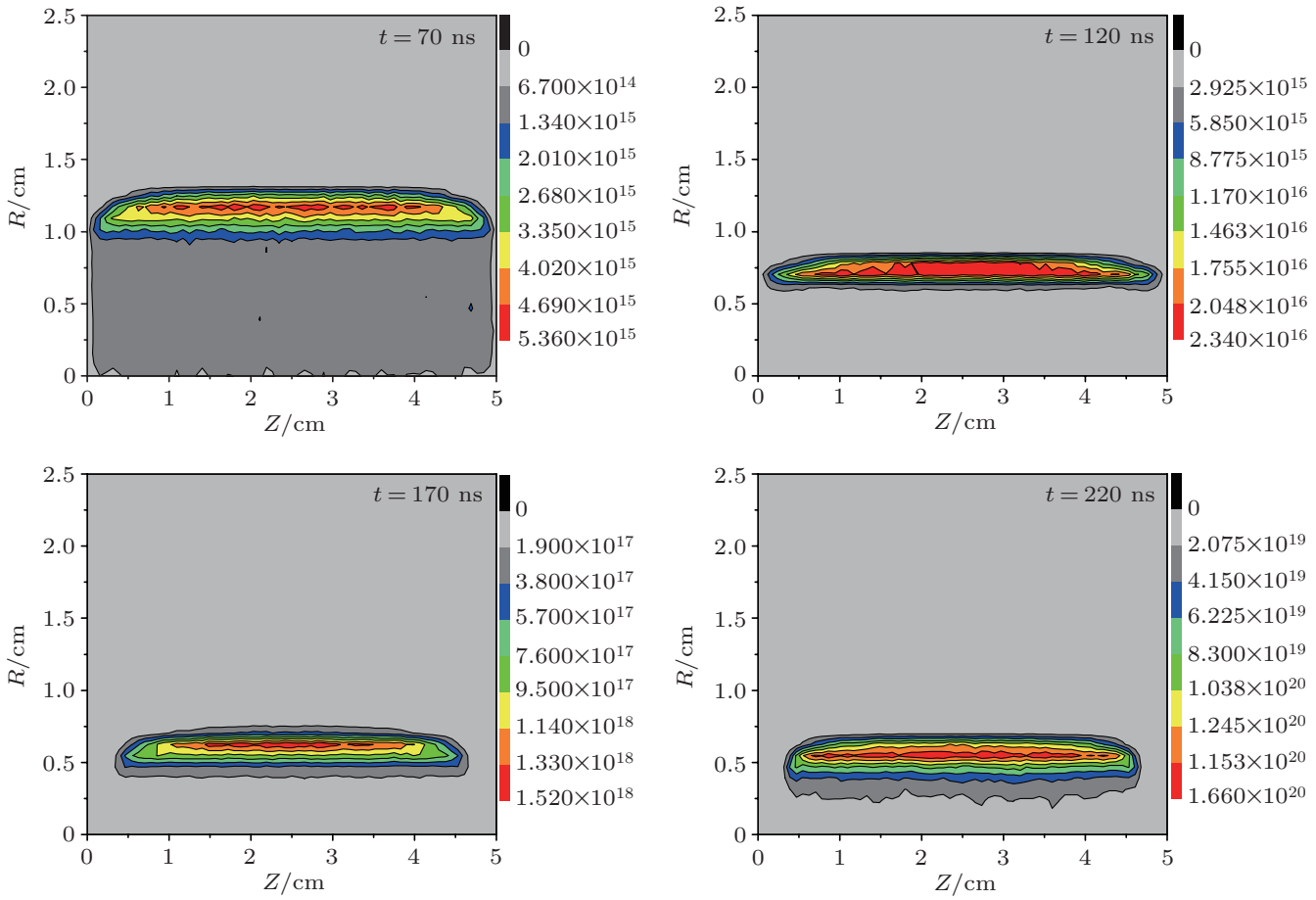


Fig. 2. (color online) A typical discharging profile evolution of the MTF pre-ionize stage, here bias magnetic field $B = 0.2$ T, Ar gas pressure = 20 mTorr (1 Torr = 1.33322×10^2 Pa) and the driving field $E^{\mathrm{T}} = E_0 \times r/R_0$ by $R_0 = 2.5$ cm and $E_0 = 50$ kV/m. After breakdown in the initial stage, the plasma ring runs closer to the axis.

When the discharge continues, the plasma generating exhausts the background gas and the plasma densities become the theoretical limit by gas densities. Before the gas is exhausted, the plasma electron temperature keeps in a low value (less than 10 eV). After the gas is exhausted, the electron and ion temperature increase rapidly. The biasing field and the gas pressure will modify the breakdown time and the plasma final parameters. In our cases, the quality behaviors of the discharging are all the same. In these parameters, the grid spacing is much larger than the Debye length but the simulating seems to be robust. We find the simulating results agree with the previous experiments[21] qualitatively well. We should point out that the five-term approximation gives out very close results with nine-term equation (11) in most of our cases.

In addition, we are running the magnetized CCP simulation with the present algorithm. It seems that the algorithm can be used to estimate the effects of magnetic field on CCP discharging. In our future study, we will apply the code on more problems, such as magnetized CCP, magnetized ICP, and others.

References

[1] Rauf S 2003 *IEEE Trans. Plasma Sci.* **31** 471
[2] Lee S H, You S J, Chang H Y and Lee J K 2007 *J. Vac. Sci. Technol. A* **25** 455
[3] Lee J K, Yang I D and Whang K M 1996 *Plasma Source Sci. Tech.* **5** 383
[4] Fan Y, Zhou Y, Sun J Z, Thomas S and Wang D Z 2013 *Phys. Plasma* **20** 103507
[5] Lieberman M A and Lichtenberg A J 2005 *Principles of Plasma Discharges and Materials Processing*, 2nd edn. (New York: Wiley)
[6] Turner M M, Derzsi A, Donkó Z, Eremin D, Kelly S J, Lafleur T and Mussenbrock T 2013 *Phys. Plasma* **20** 013507
[7] Vender D and Boswell R W 1990 *IEEE Trans. Plasma Sci.* **18** 725
[8] Surendra M, Graves D B and Morey I J 1990 *Appl. Phys. Lett.* **56** 1022
[9] Birdsall C K 1991 *IEEE Trans. Plasma Sci.* **19** 65

[10] Zhao H Y and Mu Z X 2008 *Chin. Phys. B* **17** 1475
[11] Shi F, Zhang L L and Wang D Z 2009 *Chin. Phys. B* **18** 1674
[12] Liu C S, Han H Y, Peng X Q, Chang Y and Wang D Z 2010 *Chin. Phys. B* **19** 035201
[13] Wang H H, Liu D G, Meng L, Liu L Q, Yang C, Peng K and Xia M Z 2013 *Acta Phys. Sin.* **62** 015207 (in Chinese)
[14] Verboncoeur J P 2005 *Plasma Phys. Control. Fusion* **47** A231
[15] Welch D R, Rose D V, Clark R E, Genoni T C and Hughes T P 2004 *Comput. Phys. Commun.* **164** 183
[16] Petrov G M and Davis J 2011 *Phys. Plasma* **18** 073102
[17] Markidis S, Lapenta G and Mathe R 2010 *Math. Comput. Simul.* **80** 1509
[18] Lieberman M A, Booth J P, Chabert P, Rax J M and Turner M M 2002 *Plasma Sources Sci. Technol.* **11** 283
[19] Zhang Y R, Xu X, Zhao S X, Bogaerts A and Wang Y N 2010 *Phys. Plasma* **17** 113512
[20] Kim J Y, Lee H C, Kim Y D, Kim Y C and Chung C W 2013 *Phys. Plasma* **20** 101612
[21] Taccetti J M, Intrator T P, Wurden G A 2003 *Rev. Sci. Instrm.* **74** 4314
[22] Wang H Y, Jiang W, Sun P and Kong L B 2014 *Chin. Phys. B* **23** 035204
[23] Welch D R, Rose D V, Cuneo M E, Campbell R B and Mehlhorn T A 2006 *Phys. Plasma* **13** 063105
[24] Wang H Y, Jiang W and Wang Y N 2009 *Comput. Phys. Commun.* **180** 1305
[25] Vahedi V, DiPeso G and Birdsall C K, *et al.* 1993 *Plasma Source Sci. Technol.* **2** 21
[26] Wang H Y, Jiang W and Wang Y N 2010 *Plasma Source Sci. Technol.* **19** 045023
[27] Jiang W, Xu X, Dai Z L and Wang Y N 2008 *Phys. Plasma* **15** 033502
[28] Lapenta G 2002 *J. Comput. Phys.* **181** 317
[29] Takekida H and Nanbu K 2005 *J. Phys. D: Appl. Phys.* **38** 3461