Alternative Pairwise Nuclear Fusion Optimization Algorithm in Arbitrarily Weighted Particle-in-Cell Simulations

Authors: Qian Dong

Date: 2026-02-03T13:29:24+00:00

Abstract

An optimization algorithm for pairwise nuclear fusion with arbitrarily weighted macroparticles is proposed and demonstrated using particle-in-cell (PIC) simulations. Compared with the classical nuclear fusion algorithm, the present method enables a more efficient evaluation of the Coulomb collision operator, thereby allowing straightforward integration into existing PIC codes with only minor implementation effort. The algorithm is benchmarked for like-particle fusion, D(d,n)^{3}He, unlike-particle fusion, D(t,n)^{4}He, thermonuclear fusion, and beam–target fusion scenarios. Numerical results show that the CPU time can be substantially reduced without any loss of simulation accuracy.

Full Text

An Optimized Pairwise Nuclear Fusion Algorithm for Arbitrarily Weighted Particle-in-Cell Simulations

Qian Dong,[1],[2],† Chao-Zhi Li,[1],† Chen Xie,[1] Zhi-Dong Chen,[1],[2] De-Bin Zou,[1] Wen Luo,[2],‡ and Tong-Pu Yu[1],§

[1]Department of Physics, National University of Defense Technology, 410073 Changsha, China
[2]School of Nuclear Science and Technology, University of South China, 421001 Hengyang, China

Abstract. We propose and demonstrate an optimization algorithm for pairwise nuclear fusion with arbitrarily weighted macroparticles in particle-in-cell (PIC) simulations. This algorithm achieves faster computation of the Coulomb collision operator compared to classical nuclear fusion algorithms and can be implemented in PIC simulation codes with minimal effort. The algorithm is

benchmarked for like-particle D(d,n)^{3}He reactions, unlike-particle D(t,n)^{4}He reactions, thermonuclear fusion, and beam-target fusion scenarios. The results show that CPU time can be reduced significantly without sacrificing simulation accuracy.

Keywords: Particle-in-cell simulation; Nuclear fusion; Coulomb collision; Binary collision

I. INTRODUCTION

Particle-in-cell (PIC) simulations provide insights into the kinetic properties of particles (such as velocity and position) during plasma interactions by tracking charged particles under the influence of both self-consistent and external electromagnetic fields. This technique offers an intuitive understanding of field-particle interactions and serves as a vital tool for investigating various plasma physics problems [1–3], including inertial confinement fusion (ICF) [4–6], ion acceleration [7,8], laser–nuclear interactions [9,10], and positron generation [11–14]. Although PIC simulations are widely used for modeling plasma physical processes, two critical challenges must be addressed in practical applications: (1) numerical noise, which can be significantly suppressed using various methods [15] (e.g., variable weights [16], adaptive mesh refinement [17]), and (2) the more critical challenge of high computational cost, which results in long simulation times even on parallel computers [18]. Therefore, optimizing PIC codes is both important and necessary to enhance the capabilities of corresponding simulations [19–21].

In PIC simulations, the kinetic modeling of plasmas is usually dominated by collective effects, and most PIC codes omit short-range Coulomb collisions between particles. At high temperatures (1 keV) and relatively low plasma densities (10^{27} m^{-3}), collisional effects are typically considered marginal, and a collision-free approximation is valid in PIC simulations [22]. However, in plasmas with lower temperatures or significantly higher densities, the effect of short-range interactions between particles on system evolution must be addressed. Reducing the computational effort required to account for Coulomb collisions in PIC simulations has become a primary concern in research topics including ICF and strong-field ionization [23].

Initially, the particle distribution in plasma was determined by directly solving the Fokker–Planck kinetic equation. The key to this method is calculating the Rosenbluth potential to assess the contribution of Coulomb collisions to the scattering coefficient [24]. However, coupling this method with PIC simulations was initially complex, and the approach requires numerous simulated particles to accurately calculate the Rosenbluth potential. In practical PIC simulations, most models are based on the "binary collision" model proposed by Takizuka and Abe (TA77) [25]. This model replaces direct collisions by calculating the Coulomb cross-section between particles in the same cell and randomly selecting the appropriate scattering angle. Additionally, this methodology enables

the simultaneous consideration of multiple elastic and inelastic collision channels in a pairwise process. Lavell et al. [26] applied the corrected binary collision model, originally developed for charged-particle collisions, to all scattering channels considered, validating the accuracy and utility of their model across a wide range of plasma environments. However, in binary collision modeling, the computational efficiency (i.e., speed and accuracy) of high-dimensional Monte Carlo integration methods depends strongly on the sampling technique employed. Xie et al. [27] proposed a simple and efficient Monte Carlo method for computing arbitrary ion velocity distributions. A method similar to Monte Carlo collision (MCC) for simulating the transition region between collisionless and Coulomb collisional plasma is presented in Ref. [28], where a lattice-based "collision field" concept handles collisions between different particle types, offering significant advantages for high-dimensional simulations. Manheimer et al. [29] developed a Coulomb collision model for electron–electron and electron–ion collisions that was simplified to an isotropic scattering model. The N97 method [30] uses the same order-N binary-pairing procedure as TA77 but employs a different distribution function for the scattering angle. Cohen et al. [31] described a grid-based binary model for Coulomb collisions that defines the unique particle counterpart of a particle on the grid by randomly sampling the velocity distribution, thereby omitting the process of direct particle pairing in simulation cells and significantly reducing CPU runtime.

However, all these methods assume that macroparticles have identical weights, so that the real particle density is linearly proportional to the number of macroparticles in a cell. In many simulations, variable weights are necessary to render the computational task feasible. Using appropriate pairing statistics, Miller and Combi [32] applied the TA77 method to differentially weighted particles, preserving energy and momentum while producing suitable relaxation time scales compared with theoretical predictions. Subsequently, Nanbu and Yonemura [33] developed a Coulomb collision algorithm for weighted particles based on the cumulative properties of Coulomb collisions in plasma, verifying its effectiveness through several examples. Higginson et al. [34] detailed a method for applying fusion to PIC simulations with different particle weights and validated it using analytical benchmarks for thermonuclear and beam-target fusion reactions. Unlike the work of Higginson et al. [34], Wu et al. [35] presented an algorithm for pairwise fusion applicable to macroparticles of arbitrary weights at relativistic energies, compatible with the Coulomb scattering algorithm widely used by Nanbu [33] and Sentoku [36], and implementable in PIC simulation codes with Coulomb scattering with minimal effort. Recently, Angus et al. [37] described a method for adjusting particle velocities post-scatter to restore exact conservation of momentum and energy, illustrating its efficacy with various test problems.

In the present work, we describe an efficient optimization method for the nuclear fusion algorithm and its implementation within the existing PIC code Smilei [38]. Our aim was to develop collision-operator schemes with maximum efficiency while maintaining simulation accuracy. Based on the classical nuclear

fusion algorithm described in Ref. [25], our algorithm simplifies particle pairing and reduces computational overhead by eliminating the particle address randomization step. Collision particles are selected randomly, which significantly reduces CPU time and improves the computational efficiency of collision processes.

II. OPTIMIZATION OF THE NUCLEAR FUSION ALGORITHM

A. Theoretical Background

When two fusion-reactant macroparticles a and b undergo fusion, they create two products A and B with an energy gain Q. This process is described by Eq. (1):

$$P = n_{\max} v_{\mathrm{rel}} \sigma_{ab} \Delta t,$$

where $n_{\max}$ is the maximum density of the two particles a and b, v_{rel} is the relative velocity between the particles, and the fusion reaction cross-section associated with the kinetic energy of the incident particle (denoted as σ_{ab}) is expressed as follows:

$$\sigma(E_{\mathrm{cm}}) = S(E_{\mathrm{lab}}) \exp\left(-\sqrt{\frac{E_G}{E_{\mathrm{cm}}}}\right),$$

where E_G represents the Gamow energy given by

$$E_G = (\pi \alpha_f Z_1 Z_2)^2 m_r c^2.$$

Here, E_{cm} and E_{lab} denote the kinetic energies of incident particles in the center-of-mass (CM) and laboratory frames, respectively. $\alpha_f = e^2/\hbar c$ is the fine-structure constant, Z_i denotes the atomic number, $\hbar$ represents the reduced Planck constant, and c is the speed of light. m_r is the reduced mass, defined by $m_r = m_a m_b/(m_a + m_b)$. The exponential term in Eq. (2) denotes the Coulomb barrier penetration probability. $S(E_{\mathrm{lab}})$ is known as the astrophysical S-factor, which varies gradually with kinetic energy for most nuclear reactions. According to the parametric fitting formulae of Chulick et al. [39], the values of $\sqrt{E_G}$ for the reactions D(d,n)^{3}He and D(t,n)^{4}He are 31.40 and 34.37, respectively. The cross-section and S-factor are parameterized as functions of the reactant particle energy.

Note that coordinate system transformation during nuclear fusion implementation in the code is mandatory. When a fusion reaction occurs, the fusion products are generated in the CM frame, transformed into the laboratory frame, and finally transformed back into the simulation frame.

In the laboratory frame, the coordinate origin is fixed at a point in the laboratory, similar to the target nucleus in a nuclear experiment. Particle a moves at a relative velocity of $v_{\rm rel}$. Therefore, the velocity in the simulation frame should first be transformed into the relative velocity in the laboratory frame by the transformation $v_{a,\rm lab} = |v_a - v_b|$.

[Figure 1: see original paper] (Color online) Relationship between center-of-mass velocity and incident particle velocity.

In the center-of-momentum (CM) frame, a macro-collision may be considered as being associated with two interacting macroparticles and the local density of their respective species. In general, the fusion probability P of this interaction in the CM frame can be expressed as the sum of the kinetic energy of the incident particle a and target nucleus b relative to the center of mass, typically referred to as the kinetic energy of their relative motion, denoted as $E_{\rm cm}$. As shown in Fig. 1, the distance between the incident particle a and target nucleus b is denoted by x, whereas that between the center of mass and target nucleus is denoted by x_c. According to the definition of the center of mass,

$$x_c = \frac{m_a}{m_a + m_b} x.$$

Here, dx/dt represents the velocity of incident particle a (denoted by v_a). Similarly, dx_c/dt represents the velocity of the center of mass (denoted by $v_{\rm cm}$). This equation indicates that particle a moves toward particle b with velocity v_a, whereas the center of mass moves toward particle *b$ with velocity $v_{\rm cm}$.

Therefore, the velocity of a relative to the center of mass (or equivalently, the velocity of a in the CM frame) is given by

$$v_{a,\rm cm} = v_{a,\rm lab} - v_{\rm cm} = \frac{m_b}{m_a + m_b} v_{a,\rm lab}.$$

The velocity of b relative to the center of mass (denoted by v_b in the CM frame) is equal in magnitude to the velocity $v_{\rm cm}$ of the center of mass relative to b, but opposite in direction. Hence, the following relationship is obtained:

$$v_{b,\rm cm} = -v_{\rm cm} = -\frac{m_a}{m_a + m_b} v_{a,\rm lab}.$$

The sum of the kinetic energies of particles a and b in the CM frame can be calculated using Eq. (5) and Eq. (6):

$$E_{\rm cm} = \frac{1}{2} m_a v_{a,\rm cm}^2 + \frac{1}{2} m_b v_{b,\rm cm}^2 = \frac{1}{2} m_r v_{a,\rm lab}^2.$$

Subsequently, the fusion products are generated in the CM frame with conservation of energy and momentum. The energy released in a reaction (generally

referred to as the Q-value) is given by the difference between the remaining mass energies of the reactants and products. Therefore, the total kinetic energy in the CM frame after the reaction is $Q + E_{\text{cm}}$. This energy is then distributed among product particles A and B in proportion to their masses.

In the CM frame, the total momentum of the particles is zero both before and after the reaction. This implies that $m_A v_{A,\text{cm}} = -m_B v_{B,\text{cm}}$. Therefore, the kinetic energy of the product particles can be expressed as follows (considering particle A as an example):

$$|v_{A,\text{cm}}| = \sqrt{\frac{2m_B(Q + E_{\text{cm}})}{m_A(m_A + m_B)}}.$$

To facilitate calculation of the fusion product, the momentum space is transformed such that the velocity of the center of mass is in the z-direction. The rotation matrix for this transformation is:

$$\mathbf{R} = \begin{pmatrix} \cos\theta\cos\phi & \cos\theta\sin\phi & -\sin\theta \\ -\sin\phi & \cos\phi & 0 \\ \sin\theta\cos\phi & \sin\theta\sin\phi & \cos\theta \end{pmatrix},$$

where θ represents the polar angle of the outgoing particle in the z-CM frame, and ϕ represents the azimuthal angle of the outgoing particle in this frame.

In the CM framework, particle emission is assumed to be isotropic, meaning it occurs uniformly in all directions with respect to polar angle θ. The azimuthal angle ϕ is selected from a uniform distribution between 0 and 2π. These angles are then used to calculate the velocity of the first product in the z-CM frame as follows:

$$v_{A,\text{cm-z}} = |v_{A,\text{cm}}|(\sin\theta\cos\phi\,\hat{x} + \sin\theta\sin\phi\,\hat{y} + \cos\theta\,\hat{z}).$$

From momentum conservation $m_A v_{A,\text{cm}} = -m_B v_{B,\text{cm}}$, the velocity of fusion product B in the CM frame is

$$v_{B,\text{cm-z}} = -\frac{m_A}{m_B} v_{A,\text{cm-z}}.$$

The particle velocity vector should then be transformed back into the CM frame using the inverse of the previous transformation:

$$v_{A,\text{cm}} = \mathbf{R}^T v_{A,\text{cm-z}}, \quad v_{B,\text{cm}} = \mathbf{R}^T v_{B,\text{cm-z}}.$$

Finally, after these coordinate transformations and velocity calculations, the particle velocity vector is rotated back to the simulation frame using an inverse

transformation, and the obtained fusion product is propagated as a conventional macroscopic particle in the simulation frame.

Although this calculation is performed for all binary pairs in each simulation cell, most of the computational time required by the binary collision operator is spent randomizing and pairing particles rather than computing the collision kinematics.

B. Optimizing the Algorithm

Most PIC codes (including Smilei [38]) with different collision models [30,40–42] were inspired by TA77 [25]. The time steps of these codes follow a similar procedure: (a) Particle groups are stored as a linked list within each simulation cell, a data structure that offers significant advantages for parallel computations; (b) Particle address indices are randomized to ensure randomness in subsequent particle pairing; (c) Pairs of like-particles are determined from the top of the address list; and (d) Pairs of unlike-particles are selected from the top of the address list. Finally, new particles are generated through the fusion of paired particles, the original linked list is updated, and product particle information is stored. The primary steps of this algorithm are summarized in Fig. 2 [Figure 2: see original paper].

The relaxation time used in Coulomb collision calculations is more accurate than standard approximations but is computationally expensive and requires substantial CPU time, presenting a significant challenge for large-scale PIC simulations. To improve computational efficiency while maintaining the accuracy of the Coulomb collision model, we implemented targeted optimizations to the classical binary collision algorithm.

First, we eliminated the particle address index randomization. Although macroparticles may collide multiple times during the simulation, collision pair selection is, on average, performed once per execution of the optimization algorithm. This would represent a significant time investment if particle address index randomization were performed at each time step. Instead, selection of colliding particle pairs is achieved using random numbers, which ensures the randomness of particle pairing and eliminates the need for a particle address randomization operation.

Second, we optimized the pairing of like-particles. As illustrated in Fig. 2(e), when particle pairs belong to the same species, let N_{num} denote the total number of particles in the simulation cell. Two random numbers R_a and R_b are generated (with $R_a, R_b \in (0, 1]$), and the indices of the paired particles are determined by $p_a = R_a N_{\mathrm{num}}$ and $p_b = R_b N_{\mathrm{num}}$. This step is performed in a loop until $N_{\mathrm{num}}/2$ particle pairs are formed. If $p_a = p_b$ and $p_a \neq N_{\mathrm{num}}$, then p_a is incremented by one; otherwise, it is decremented by one. These steps are applied recursively to the remaining simulation cells until all cells are paired. The pseudocode is provided in Algorithm 1.

Algorithm 1: Pairing of like-particles
Input: Information regarding all macroparticles in the simulation cell; Number of particles in the cell (N_{num})
Output: Information of $N_{\text{num}}/2$ particle collision pairs
1. repeat
2. Generate two random numbers $R_a, R_b \in (0, 1]$
3. // Determine the index of particle pairs
4. $p_a = N_{\text{num}} \times R_a$, $p_b = N_{\text{num}} \times R_b$
5. if $p_a = p_b$ and $p_a \neq N_{\text{num}}$ then
6. // If particle pairs have equal index within valid bounds
7. $p_a = p_a + 1$
8. else
9. // If particle pairs have equal boundary index
10. $p_a = p_a - 1$
11. end if
12. until $N_{\text{num}}/2$ pairs of colliding particles are selected
13. return $N_{\text{num}}/2$ particle colliding pairs

Third, we optimized the pairing of unlike-particles. For pairings between different particle types, the total numbers of particles with lower density (N_a) and higher density (N_b) should first be determined. Similar to the optimization for like-particles, two random numbers $R_a, R_b \in (0, 1]$ are generated, and the indices of the pairs are determined by $p_a = R_a N_a$ and $p_b = R_b N_b$. As shown in Fig. 2(f), this procedure is repeated until N_a particle pairings are completed. The process is then executed recursively across all simulation cells until all cells are paired. The pseudocode is given in Algorithm 2.

Algorithm 2: Pairing of unlike-particles
Input: Information regarding all macroparticles in the cell; Number of particles with lower density in the cell (N_a); Number of particles with higher density in the cell (N_b)
Output: Information of N_a particle collision pairs
1. repeat
2. Generate two random numbers $R_a, R_b \in (0, 1]$
3. // Determine different density particle index
4. $p_a = N_a \times R_a$, $p_b = N_b \times R_b$
5. until N_a pairs of colliding particles are selected
6. return N_a particle colliding pairs

The spatial grid size in this procedure is of the order of the Debye radius. Because it is unnecessary to consider Coulomb collisions between particles spaced farther apart than the Debye radius, interactions between particles in different simulation cells can be omitted, enabling parallel computation of the simulation procedures. A flowchart comparing the classic binary collision algorithm (TA77) with our optimization algorithm (NEW) is shown in Fig. 3 [Figure 3: see original paper].

III. BENCHMARKS

To evaluate the efficiency of the optimization algorithm, we present three typical simulations: D(d,n)^{3}He and D(t,n)^{4}He thermonuclear fusion, and D(d,n)^{3}He beam-target fusion. The results demonstrate that the optimization algorithm computes the Coulomb collision process with maximum efficiency without compromising physical accuracy. These benchmarks were implemented using the PIC code Smilei [38]. All tests were performed with fields disabled. Simulations can use cross-sectional data derived from experimental measurements or parametric fitting equations; in this work, the second approach was preferred to ensure direct comparison with theoretical predictions.

It is important to emphasize that the optimization algorithm operates on probability-driven random pairing rather than complete traversal of all particles. Although the number of pairings is limited by the lower-density species, the optimization algorithm leverages Monte Carlo methods to achieve robust sampling of high-density species. In nuclear fusion, the ion velocity distribution follows the Maxwell–Boltzmann distribution, wherein energetic particles (although sparse) contribute significantly to the fusion reaction [34]. Random pairing cumulatively covers the high-energy region through multiple independent samplings (cycling each time step) rather than relying on single-step traversal. Consequently, the statistical fluctuations of random pairing are attenuated by time-averaging effects in long-duration simulations, ensuring accurate kinetic evolution.

A. D(d,n)^{3}He Thermonuclear Fusion

In the first benchmark, we created a simulation box with dimensions $L_x = 40\pi c/\omega_r$ and $L_y = 40\pi c/\omega_r$, divided into a 100×100 grid, with each cell containing an equal number of D ions. The number density of D ions was 10^{20} cm^{-3}, and the particles were weighted unequally. Thermonuclear fusion occurs when ions attain high temperatures. Because the fusion cross-section depends strongly on the relative velocity of particles, most fusion events occur between ions in the tails of the distribution with kinetic energies several times higher than the mean plasma energy. Simulations were performed at different temperatures to evaluate fusion reactivity and neutron spectra.

The neutron energy spectra obtained at different temperatures are shown in Fig. 4 [Figure 4: see original paper]. Both TA77 and the optimization algorithm yielded similar neutron energy spectra. The full width at half maximum (FWHM) of the neutron energy spectra increased with initial ion temperature. To validate the accuracy of the energy spectra generated by the optimization algorithm, the number of particles per cell (PPC) was fixed at 1000 for all simulations. As illustrated in Fig. 5(a) [Figure 5: see original paper], by varying the initial temperature of the D ions, we compared the effective FWHM of the simulated energy spectra with the analytical fit of Ballabio [43] using relativistic kinematics. The results reveal remarkable agreement. The ultimate goal

of the optimization algorithm is to improve computational efficiency without altering the underlying physics. Figure 5(b) presents the CPU runtime for the D(d,n)^{3}He thermonuclear fusion simulation with an initial temperature of 10 keV. Compared with TA77, the CPU runtime was reduced by 30.5%. Although the improvement in computational efficiency may not be universal, it becomes more evident when simulating larger numbers of particles in complex collisions.

B. D(t,n)^{4}He Thermonuclear Fusion

To test the optimization algorithm for unlike-particles and illustrate the case of unequal weighting, we conducted a simulation for D(t,n)^{4}He thermonuclear fusion. The initial conditions were similar to the first benchmark, with a density of 10^{20} cm^{-3} for both ions and an initial temperature of 5 keV. The neutron energy spectra were benchmarked by varying the initial numbers of D and T ions in each cell. As shown in Fig. 6(a) [Figure 6: see original paper], when the initial number of D ions was less than that of T ions (N_D = 100 PPC, N_T = 1000 PPC), the optimization algorithm matched the neutron energy spectrum from the TA77 simulation. In the opposite case, with the particle numbers reversed (N_D = 1000 PPC, N_T = 100 PPC), the results were as anticipated (see Fig. 6(b)). Furthermore, when equal weights were maintained and the initial density ratios of D and T ions were varied, the neutron energy distribution was unaffected by variations in initial particle density, influencing only the peak of the energy spectrum.

Because the nuclear reaction mechanism of the second benchmark differs from the first, we performed multiple simulation sets. The evolution of the neutron energy FWHM with ion temperature is shown in Fig. 7(a) [Figure 7: see original paper], again demonstrating remarkable agreement between the optimization algorithm (NEW) and theory (Ballabio). Additionally, we compared the computational efficiency of D(t,n)^{4}He thermonuclear fusion in both cases, as shown in Fig. 7(b), revealing a 25.5% reduction in CPU runtime for the optimized algorithm compared with TA77.

C. D(d,n)^{3}He Beam-Target Fusion

Two distinct tests were conducted to investigate the feasibility of D(d,n)^{3}He beam-target fusion using the optimization algorithm. In these simulations, D(d,n)^{3}He beam-target fusion occurred when a high-energy beam impacted a cold, stationary D ion target. In this scenario, the kinetic energy of the beam determined the magnitude of the fusion cross-section. One group of D ions was considered as a stationary background with a density of 10^{20} cm^{-3}, while another group with equal density served as the projectile. Two simulations were performed with projectile velocities of $0.05c$ and $0.1c$, and the resulting neutron spectra are presented in Fig. 8 [Figure 8: see original paper]. Figures 8(a) and 8(b) illustrate the simulation results with initial velocities of $0.05c$ and $0.1c$, respectively. As shown, the optimization algorithm accurately reproduced the neutron energy spectra.

Because the optimization algorithm partially removed particle traversal steps, we illustrate the relative error in total kinetic energy during the simulation in Fig. 9(a) [Figure 9: see original paper]. For the optimization algorithm, the peak of the relative error stabilized at $\pm 0.02\%$, an insignificant energy deviation considering statistical fluctuations during simulation. Additionally, as shown in Fig. 9(b), the optimization algorithm significantly reduced CPU runtime by 32.5%.

IV. CONCLUSION

In this study, we addressed the computational inefficiency of multiparticle complex collisions in PIC simulations by optimizing the classical PIC binary Coulomb collision algorithm. The particle-pairing process was simplified by removing the particle address randomization step, and collision particle pairs were selected using random numbers. This eliminated the need to traverse all particle address information within simulation cells and significantly reduced CPU runtime. The optimized algorithm was benchmarked for like-particle D(d,n)^{3}He, unlike-particle D(t,n)^{4}He, thermonuclear fusion, and beam-target fusion scenarios. The results show that CPU runtime can generally be reduced by approximately 30% while maintaining physical accuracy. The minimal errors in energy and momentum enable the optimized algorithm to minimize its effect on the underlying physics.

For plasmas with significantly high density (10^{27} m^{-3}) or significantly low temperature (1 keV), the collision frequency may increase substantially, which would amplify the statistical error in the optimization algorithm (NEW). Under such conditions, particle correlation effects (e.g., collective oscillations or strong coupling) may exceed the algorithm' s assumptions for independent collision events. We plan to further validate the robustness of the optimization algorithm in future studies. The generalization for parallel computations is straightforward because the optimization algorithm operates cell-by-cell, and particles are "sorted" throughout the simulation. Therefore, the performance of the optimization algorithm is likely to improve substantially as the number of particles and simulation system complexity increase. More importantly, the optimization algorithm enables evaluation of problems that would otherwise be inaccessible due to computational resource limitations, providing technical support for subsequent large-scale particle collision simulations, including ICF and laser-driven neutron sources.

[1] E.P. Alves, W.B. Mori, F. Fiuza et al., Numerical heating in particle-in-cell simulations with Monte Carlo binary collisions. Phys. Rev. E 103, 013306 (2021). doi: 10.1103/PhysRevE.103.013306
[2] J.S. Ross, D.P. Higginson, D. Ryutov et al., Transition from collisional to collisionless regimes in interpenetrating plasma flows on the national ignition facility. Phys. Rev. Lett. 118, 185003 (2017). doi: 10.1103/PhysRevLett.118.185003
[3] X.Y. An, M. Chen, J.L. Liu et al., Modeling of axion and electromagnetic

fields interaction in particle-in-cell simulations. Matter Radiat. Extrem. 9, 067204 (2024). doi: 10.1063/5.0226159
[4] Y.D. Lu, J.X. Zheng, X.F. Liu et al., An improved analysis method for assessing the nuclear-heating impact on the stability of toroidal field magnets in fusion reactors. Nucl. Sci. Tech. 35, 96 (2024). doi: 10.1007/s41365-024-01459-5
[5] E.M. Campbell, V.N. Goncharov, T.C. Sangster et al., Laser-direct-drive program: promise, challenge, and path forward. Matter Radiat. Extrem. 2, 37–54 (2017). doi: 10.1016/j.mre.2017.03.001
[6] X.R. Jiang, F.Q. Shao, D.B. Zou et al., Energetic deuterium-ion beams and neutron source driven by multiple-laser interaction with pitcher-catcher target. Nucl. Fusion 60, 076019 (2020). doi: 10.1088/1741-4326/ab91f9
[7] A. McIlvenny, D. Doria, L. Romagnani et al., Selective ion acceleration by intense radiation pressure. Phys. Rev. Lett. 127, 194801 (2021). doi: 10.1103/PhysRevLett.127.194801
[8] T. Minami, C.M. Chu, O. McCusker et al., Ion acceleration with an intense short-pulse laser and large-area suspended graphene in an extremely thin target regime. High-energy density Phys. 55, 101195 (2025). doi: 10.1016/j.hedp.2025.101195
[9] J.H. Cheng, Q. Xiao, J.G. Deng et al., Laser-assisted decay of the deformed odd-A nuclei. Nucl. Sci. Tech. 36, 69 (2025). doi: 10.1007/s41365-024-01610-2
[10] Q. Xiao, J.H. Cheng, Y.Y. Xu et al., Decay in extreme laser fields within a deformed Gamow-like model. Nucl. Sci. Tech. 35, 27 (2024). doi: 10.1007/s41365-024-01371-y
[11] H Chen, S.C. Wilks, D.D. Meyerhofer et al., Relativistic quasi-monoenergetic positron jets from intense laser-solid interactions. Phys. Rev. Lett. 105, 015003 (2010). doi: 10.1103/PhysRevLett.105.015003
[12] J. Zhao, Y.T. Hu, Y. Lu et al., All-optical quasi-monoenergetic GeV positron bunch generation by twisted laser fields. Commun. Phys. 5, 15 (2022). doi: 10.1038/s42005-021-00797-9
[13] F. Wan, C. Lv, K. Xue et al., Simulations of spin/polarization-resolved laser–plasma interactions in the nonlinear QED regime. Matter Radiat. Extrem. 8, 064002 (2023). doi: 10.1063/5.0163929
[14] T.P. Yu, K. Liu, J. Zhao et al., Bright X/ -ray emission and lepton pair production by strong laser fields: a review. Rev. Mod. Plasma Phys. 8, 24 (2024). doi: 10.1007/s41614-024-
[15] J.D. Blahovec, L.A. Bowers, J.W. Luginsland et al., 3-D ICEPIC simulations of the relativistic klystron oscillator. IEEE Trans. Plasma Sci. 8, 24 (2024). doi: 10.1109/27.887733
[16] Q. Dong, B.L. Wang, X.J. Duan et al., A dynamical particle merging and splitting algorithm for particle-in-cell simulations. Comput. Phys. Commun. 294, 108913 (2024). doi: 10.1016/j.cpc.2023.108913
[17] S.M. Robert, J.L. Cambie, Octree particle management for DSMC and PIC simulations. J. Comput. Phys. 327, 943–966 (2016). doi: 10.1016/j.jcp.2016.01.020
[18] J P Verboncoeur, Particle simulation of plasmas: review and advances.

Plasma Phys. Control. Fusion 47 A231 (2005). doi: 10.1088/0741-3335/47/5A/017
[19] E. Kawamura, C.K. Birdsall, V. Vahedi, Physical and numerical methods of speeding up particle codes and paralleling as applied to RF discharges. Plasma Sources Sci. Technol. 9, 413 (2000). doi: 10.1088/0963-0252/9/3/319
[20] R. Li, J.L. Jiao, H. Luo et al., Two-dimensional particle-in-cell modeling of blow-off impulse by X-ray irradiation. Nucl. Sci. Tech. 35, 53 (2024). doi: 10.1007/s41365-024-01412-6
[21] S.H. Ren, H.Y. Li, J.R. Shi et al., Calculation algorithm for the space charge force of a train with infinite bunches. Nucl. Sci. Tech. 36, 96 (2025). doi: 10.1007/s41365-025-01673-9
[22] M. Ramsay, Dissertation, University of Warwick, 1975
[23] D. Tskhakaya, R. Schneider, Optimization of PIC codes by improved memory management. J. Comput. Phys. 225, 829–839 (2007). doi: 10.1016/j.jcp.2007.01.002
[24] R.J. Procassini, B.I. Cohen, A comparison of particle-in-cell and fokker-planck methods as applied to the modeling of auxiliary-heated mirror plasmas. J. Comput. Phys. 102, 39–48 (1992). doi: 10.1016/S0021-9991(05)80003-8
[25] T. Takizuka, H. Abe, A binary collision model for plasma simulation with a particle code. J. Comput. Phys. 25, 205–219 (1977). doi: 10.1016/0021-9991(77)90099-7
[26] M.J. Lavell, A.J. Kish, A.T. Sexton et al., Verification of a Monte Carlo binary collision model for simulating elastic and inelastic collisions in particle-in-cell simulations. Phys. Plasmas 31, 043902 (2024). doi: 10.1063/5.0190352
[27] H.S. Xie, A simple and fast approach for computing the fusion reactivities with arbitrary ion velocity distributions. Comput. Phys. Commun. 292, 108862 (2023). doi: 10.1016/j.cpc.2023.108862
[28] M.E. Jones, D.S. Lemons, R.J. Mason et al., A grid-based Coulomb collision model for PIC codes. J. Comput. Phys. 123, 169–181 (1996). doi: 10.1006/jcph.1996.0014
[29] W.M. Manheimer, M. Lampe, G. Joyce, Langevin representation of Coulomb collisions in PIC simulations. J. Comput. Phys. 138, 563–584 (1997). doi: 10.1006/jcph.1997.5834
[30] K. Nanbu, Momentum relaxation of a charged particle by small-angle Coulomb collisions. Phys. Rev. E. 56, 7314–7314 (1997). doi: 10.1103/PhysRevE.56.7314
[31] B.I. Cohen, A.M. Dimits, D.J. Strozzi, A grid-based binary model for Coulomb collisions in plasmas. J. Comput. Phys. 234, 33–43 (2013). doi: 10.1016/j.jcp.2012.08.046
[32] R.H. Miller, M.R. Combi, A Coulomb collision algorithm for weighted particle simulations. Geophys. Res. Lett. 21, 1735–1738 (1994). doi: 10.1029/94GL01835
[33] K. Nanbu, S. Yonemura, Weighted particles in Coulomb collision simulations based on the theory of a cumulative scattering angle. J. Comput. Phys. 145, 639–654 (1998). doi: 10.1006/jcph.1998.6069
[34] D.P. Higginson, A. Link, A. Schmidt, A pairwise nuclear fusion algorithm

for weighted particle-in-cell plasma simulations. J. Comput. Phys. 388, 439–453 (2019). doi: 10.1016/j.jcp.2019.03.020
[35] D. Wu, Z.M. Sheng, W. Yu et al., A pairwise nuclear fusion algorithm for particle-in-cell simulations: weighted particles at relativistic energies. AIP Adv. 11, 075003 (2021). doi: 10.1063/5.0051178
[36] Y. Sentoku, A.J. Kemp, Numerical methods for particle simulations at extreme densities and temperatures: weighted particles, relativistic collisions and reduced currents. J. Comput. Phys. 227, 6846–6861 (2008). doi: 10.1016/j.jcp.2008.03.043
[37] J.R. Angus, Y.C. Fu, V. Geyko et al., Moment-preserving Monte-Carlo Coulomb collision method for particle codes. J. Comput. Phys. 531, 113927 (2025). doi: 10.1016/j.jcp.2025.113927
[38] J. Derouillat, A. Beck, F. Pérez et al., Smilei: a collaborative, open-source, multi-purpose particle-in-cell code for plasma simulation. Comput. Phys. Commun. 222, 351–373 (2018). doi: 10.1016/j.cpc.2017.09.024
[39] G.S. Chulick, Y.E. Kim, R.A. Rice et al., Extended parameterization of nuclear-reaction cross sections for few-nucleon nuclei. Nucl. Phys. A 551, 255–268 (1993). doi: 10.1016/0375-9474(93)90481-C
[40] V. Vahedi, 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). doi: 10.1016/0010-4655(94)00171-W
[41] A.J. Christlieb, R. Krasny, J.P. Verboncoeur, A treecode algorithm for simulating electron dynamics in a Penning–Malmberg trap. Comput. Phys. Commun. 164, 306–310 (2004). doi: 10.1016/j.cpc.2004.06.076
[42] P.T. Wang, X.S. Geng, G. Q. Zhang et al., Laser-driven micro-pinch: a pathway to ultra-intense neutrons. Nucl. Sci. Tech. 36, 108 (2025). doi: 10.1007/s41365-025-01738-9
[43] L. Ballabio, J. Källne, G. Gorini, Relativistic calculation of fusion product spectra for thermonuclear plasmas. Nucl. Fusion 38, 1723 (1998). doi: 10.1088/0029-5515/38/11/310

Note: Figure translations are in progress. See original paper for figures.