Null collision Monte Carlo simulation model for particle-in-cell method

V.A.Vshivkov, A.A.Soloviev, I.S.Chernoshtanov, A.A.Efimova
Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk, Russia.
E-mail: vsh@ssd.sscc.ru

Abstract. The article discusses the original null collision approach for modeling collisions in plasma for particle-in-cell method. A distinctive feature of the method is the simulation of particle collisions without the use of trigonometric functions and calculation of pairwise collisions (as is done in classical papers), which significantly increases the speed of the developed algorithms. The developed algorithms are used in other software systems for modeling particle motion in a magnetic field. Comparison of the results of system’s operation in the presence of collisions and in their absence is made.

1. Introduction
Simulating plasma collisions using the Monte Carlo method is a well-known problem. Many articles have been published on this topic with differing approaches to its solution.

For example, in the classical paper [1], pairwise collisions of particles are simulated based on principles of classical mechanics, which allows conserving the total energy and momentum of the entire system.

Unfortunately, for the particle-in-cell method, when the number of tracked particles and the number of iterations are large enough, this approach is excessively laborious, since it requires searching and sorting over the entire array of particles [2]. In addition, the calculation of pairwise collisions is rather slow because of the need to calculate the scattering angles and new velocities of the particles.

Various authors in their papers [3-4] have proposed a number of ideas that allow avoiding the calculation of pairwise or group collisions using the so-called ”Null collision” approach. In this approach, there is no calculation of the direct interaction (collision) of two or more particles, but the speed and trajectory of individual particle change. The parameters of these changes (collision frequency, deviation magnitude, new velocity) are determined from other parameters, for example, density, average thermal ion velocity, etc.

Despite the fact that this approach can significantly reduce the program execution time, it nevertheless has its drawbacks. One of the main disadvantages in the case of a three-dimensional problem is the fact that it is required to determine the new coordinates of the particle’s motion vector immediately after changing its trajectory, which cannot be done without the use of trigonometric functions (whatever method is used).

So, for example, there is a standard way to calculate deflection parameters in 3D space uses a rotation matrix in 3D space (see fig. 1):
Figure 1. 3D rotation

The rotation matrix has the form

\[
\begin{pmatrix}
\cos \alpha \cos \beta & \cos \alpha \sin \beta \sin \gamma - \sin \alpha \cos \gamma & \cos \alpha \sin \beta \cos \gamma + \sin \alpha \sin \gamma \\
\sin \alpha \cos \beta & \sin \alpha \sin \beta \sin \gamma + \cos \alpha \cos \gamma & \sin \alpha \sin \beta \cos \gamma - \cos \alpha \sin \gamma \\
-\sin \beta & \cos \beta \sin \gamma & \cos \beta \cos \gamma
\end{pmatrix}
\]

The calculation of such a number of trigonometric functions on a computer is a relatively slow procedure, which ultimately affects the final speed of program execution [5].

We have proposed and programmatically implemented an approach based on "Null collision", which does not use trigonometric functions at all.

2. Trigonometry-free null collision Monte Carlo simulation model

The proposed method is based on the fact that the interaction of any two particles in a cell can be represented as the interaction of this particle and the "average" particle in this cell. This eliminates the need to search and sort by an array of particles.

This approach uses some of the ideas described in [6-8].

The interaction of particles is represented as the scattering of every particle on a stationary "average" particle (for this, a transition to the corresponding coordinate system is done).

To determine the mean free path of a particle \(l_i\), a formula is used from [8]

\[
l_i = \frac{T_i^2}{4\pi e^4 N_k L_i},
\]

where \(T_i\) is thermal energy, \(N_k\) is density in particle’s cell, \(L_i\) is Coulomb logarithm with ionic components.

The essence of the method is easiest to demonstrate in the figure. The place of "impact", and, as a consequence, the angle of deflection of the particle from its initial trajectory is chosen randomly.

The maximum angle of deflection of a particle is taken as 30\% (in this case, the principles of classical mechanics are used, specifically, the solution of the problem when one ball hits a stationary other with the same mass. It is easy to show that the angle of deflection of a flying ball cannot exceed \(\pi/6\)).

The algorithm consists of the following steps (see fig. 2):

1. Around the end of the particle velocity vector (green vector), a ball is constructed, the radius of which is chosen so that the deflection angle (between green and red vectors) does not exceed \(\pi/6\).
2. A cube with sides 2R is built around the ball.
3. Then we choose 3 random uniformly distributed variables $a$, $b$ and $c$ (from -1 to 1), which are responsible for a random point inside this cube (normalized by length). We need them to be inside the ball, not inside the cube, so we check the condition $a^2 + b^2 + c^2 \leq 1$. If it fails (a point in the cube, but outside the ball), we re-generate $a$, $b$ and $c$ until the condition is met. The probability of choosing 3 suitable variables on the first try is approximately 52%, and the average number of steps is approximately 1.9. Given that choosing a random number is a very fast procedure, regeneration practically does not slow down the program.

4. We build a new vector (in the figure in yellow) with added variables $a$, $b$ and $c$.

At the end of the procedure for calculating the new parameters of motion for all particles, the energy and momentum of the entire system is normalized. As you can see, trigonometric functions are not used in the algorithm. However, this approach also has drawbacks, since it requires normalizing the parameters of the system at each step of the collision calculation.

Next, we briefly describe another collisional algorithm in order to compare the results of the proposed null collision algorithm.

3. Algorithm for collisional diffusion calculating

In this section, we will briefly talk about one of the collisional algorithms (then we will compare the results of this algorithm with the algorithm described above).

Let a monoenergetic beam of particles of type $a$ move through a stationary plasma. Due to the interaction with plasma particles, the average beam velocity $<v_\alpha>$ decreases in accordance with the equation (2)

$$\frac{d <v_\alpha >}{dt} = - <v_\alpha > \sum v_{s/\beta}^{\alpha/\beta}$$

In a plasma with a Maxwellian particle velocity distribution, the frequencies $v_{s/\beta}^{\alpha/\beta}$ are:

$$v_{s/\beta}^{\alpha/\beta} = \frac{4\pi e_\alpha^2 e_\beta^2 \Lambda_{\alpha\beta} n_\beta}{m_\alpha^2 v_\alpha^3} \left(1 + \frac{m_\alpha}{m_\beta}\right) \Psi \left(\frac{v_\alpha^2}{u_\beta^2}\right), \quad \Psi(x) = \frac{2}{\sqrt{\pi}} \int_0^x \sqrt{t} e^{-t} dt.$$  \hspace{0.5cm} (3)

Here $e_\alpha$ and $m_\alpha$ are charge and mass of $\alpha$-type particle, $\Lambda_{\alpha\beta}$ is Coulomb logarithm, $u_\beta = \sqrt{2T_\beta/m_\beta}$ is thermal velocity of $\beta$-type particles.

Due to angular scattering, the scatter in the transverse component (across the average beam velocity) and the longitudinal components of the velocities of the beam particles increases
Figure 3. Velocity distribution of particles as a result of the null collision algorithm

monotonically (4)-(5):

\[
\frac{d}{dt} \langle v^2_\perp \rangle = v^\alpha_\perp v^2_\perp,
\]

\[
v^\alpha_\perp = 2 \frac{4\pi e^2 e^2 \lambda_{\alpha\beta} n_\beta}{m_\alpha^2 v_\alpha^3} \left( 1 - \frac{u^2_\beta}{2v^2_\alpha} \right) \Psi \left( \frac{v^2_\alpha}{u^2_\beta} \right) + \Psi' \left( \frac{v^2_\alpha}{u^2_\beta} \right),
\] (4)

\[
\frac{d}{dt} \left( (v^\parallel_\alpha - \langle v_\alpha \rangle)^2 \right) = v^\alpha_\parallel v^2_\parallel,
\]

\[
v^\alpha_\parallel = 4\pi e^2 e^2 \lambda_{\alpha\beta} n_\beta \frac{u^2_\beta}{v_\alpha^2} \Psi \left( \frac{v^2_\alpha}{u^2_\beta} \right).
\] (5)

We will not dive into the specifics of the realization of this algorithm on a computer, we just note that it was implemented in software and then compared with the above-described null collision algorithm.

4. Results
First, let’s compare the operation of the null collision and collisional algorithms, and in this case we do not pay attention to the speed of operation, but to the obtained distributions.

4.1. Comparison of the distributions of null collision and collisional algorithms
Both algorithms were implemented in computer software. The simulation showed that both approaches lead to the Maxwellian distribution of particles (see figures 3 and 4).

It can be seen that the distributions are similar, so the null collision algorithm can be used as a faster alternative.

4.2. Comparison of speed with other null collision algorithms
In the classical paper [4], the scattering of a particle is realized in the form of a deviation of its trajectory. In this case, the length of the vector is conserved, which does not require normalization.

This approach was implemented as a test program in order to compare the performance of these two algorithms.
Simulation showed that for a system containing 5000 particles, the time gain reaches 10% due to the absence of trigonometric functions (even taking into account the normalization procedure) compared to the approach described in [4]. However, it should be noted that the result may be different when calculating on specialized computers, orientated on calculation of trigonometric functions.

4.3. Application of the procedure in modeling plasma confinement systems
The developed null collision method was used in modelling plasma confinement systems to estimate the motion of particles in a magnetic field taking ion-ion collisions into account [9]. The most significant difference is the distribution of background ions during the operation of such systems, which can be seen in the figures 5 and 6.
As can be seen in the figures, ion-ion collisions affect the results obtained; therefore, the development of high-speed null collision methods are very important for this kind of research.

5. Conclusion
In this work, a high-speed algorithm was proposed for simulating ion-ion collisions in plasma for the particle-in-cell method (for one type of ions). The main advantage of the developed algorithms is a very high speed of work due to the absence of the need to calculate trigonometric functions. The developed algorithms were used in plasma confinement modeling systems. The results showed that taking into account ion-ion collisions significantly affects many parameters in the system, such as the distribution of background ions and ion velocities at the cavity boundaries.

Acknowledgments
This work was supported by the Russian Science Foundation, project 19-71-20026.

References
[1] Takizuka T., Abe H. A binary collision model for plasma simulation with a particle code // Journal of Computational Physics. 1977. V. 25. No. 3. pp. 205-219.
[2] Wang C., Lin T., Califsh R., Cohen B. Particle simulation of Coulomb collisions: Comparing the methods of Takizuka & Abe and Nanbu // Journal of Computational Physics. 2008. V. 227. No. 9. pp. 4308-4329.
[3] Galtier M., Blanco S., Caliot C., Coustet C., Dauchet J. Integral formulation of null-collision Monte Carlo algorithms // Journal of Quantitative Spectroscopy and Radiative Transfer. 2013. V. 125. pp. 57-68.
[4] Birdsall C. K. Particle in cell Monte Carlo collision codes (PIC-MCC); Methods and Applications to Plasma Processing // Plasma Processing of Semiconductors. 1997. pp. 277-289.
[5] Foerster P. A. Algebra and Trigonometry: Functions and Applications. New Jersey: Savvas Learning Co. 2005. 1058 p.
[6] Vahedi V., Surendra M. A Monte Carlo collision model for the particle-in-cell method: applications to argon and oxygen discharges // Computer Physics Communications. 1995. V. 87. No. 1–2. pp. 179-198.
[7] Freidberg J. Electromagnetic Interactions. Massachusetts Institute of Technology: MIT OpenCourseWare. 2005. 22.105. https://ocw.mit.edu.
[8] Leontovich M. A. Reviews of Plasma Physics. New-York: Springer. 1995. 525 p.
[9] Schwab I., Astrelin V., Burdakov A., Vshivkov V., Vshivkov K., Medvedev S., Yakunkin N. Numerical modeling of plasma dynamics in non-uniform magnetic field // Fusion Science and Technology. 2011. V. 59. No. 1. pp. 313-315.