Performance improvement of particle-in-cell method for numerical modelling of open magnetic system

M.A. Boronina, I.G. Chernykh, E.A. Genrikh, V.A. Vshivkov
Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Lavrentiev avenue 6, Novosibirsk, 630090, Russia
E-mail: boronina@ssd.sscc.ru

Abstract. The work is connected with numerical simulation of plasma dynamics in open magnetic trap in diamagnetic regime. The hybrid particle-in-cell model we develop allows to perform numerical experiments for the high ratio of Larmor radius of the ions and the one of the electrons due to the combining of the kinetic description for the ions and MHD description for the electron plasma component. The disadvantage of the model is the stability condition and the corresponding requirements for the time step. In practice a doubling of the grid nodes in each direction leads to the decrease of the time step x6 times. For the characteristic times of the plasma processes $10^2$ reciprocal ion cyclotron frequencies the computations required few days for grids 100x500. The processing of the particle data takes more than 85% of the total computation time, thus its effective realization yields significant gain in performing. In our algorithm we combine the dynamic load balance and vectorized computations of densities and current densities. We present the results of numerical experiments on its basis including the highly non-uniform particle distribution in the domain and the increasing of the particle number due to the beam injection.

1. Introduction
An original concept of plasma confinement in a linear mirror trap is proposed in [1]. The idea is based on the expelling of the magnetic field of the mirror trap with the particles injected into the trap. The domain with low values of the magnetic field near the injection point has shape of bubble. In the thin bound of the bubble the electro-magnetic field gradient is high, the majority of the ions do not cross the boundary layer. The ratio of the plasma pressure to the pressure of the magnetic field $\beta$ may reach values $\sim 1$. The high values of $\beta$ ($\sim 0.6$ in GDL experiments [2]) and engineering simplicity turn the compact mirror traps into a promising alternative to tokamaks in the field of the plasma fusion. The questions of the bubble formation and equilibrium are considered theoretically, the construction of a new trap CAT (BINP SB RAS, Russia) is in process and expected to demonstrate the confinement in the diamagnetic “bubble” regime [3].

Numerical experiments for the study of the plasma dynamics in the magnetic field may be performed basing on hybrid model [4],[5] using particle-in-cell method [6]. Kinetic approach for electrons and ions both is difficult to be applied due to the significant differences in the spatial and time scales based on the mass ratio $m_i/m_e \sim 2 \cdot 10^9$: the electron Larmor radius requires proportionally finer grids and consequently increases computational costs. The hybrid description, which includes kinetic Vlasov equation for the ion component of the plasma
and the magneto-hydrodynamic equations for the electron component, allows to decrease the computation requirements. Another way of acceleration of getting the numerical solution is based on code parallelization and its application with large number of processes. Vectorization approach, when one operation is applied to few values at one time, enables exploitation of processors in intensive regime and improves the code performance.

In the paper we present the numerical algorithm based on the described approaches for performing simulations on the diamagnetic regime of plasma-field interaction in the open magnetic trap taking into account the non-uniform particle distribution in space and time.

2. Problem statement
We consider 2D-3V model of cylindrical open trap with background hydrogen plasma of density $n_0$ at the initial time moment. We use rectangular domain of sizes $[R_{\text{max}} \times L_{\text{max}}]$ and cylinder coordinate system. The magnetic field at the trap center $(0, L_{\text{max}}/2)$ is $H_0$ and is defined by two mirrors on the ends of the trap with the mirror ratio $R_m = 2$. The ion beam particles are permanently injected from the point $(0, L_{\text{max}}/2)$. The plasma is considered quasi-neutral, the densities of the ion and electron component coincide: $n = n_i = n_e$.

In dimensionless variables the phenomena can be described with characteristic equations of kinetic Vlasov equation for the ions:

$$\frac{d\vec{r}_i}{dt} = \vec{v}_i, \quad (1)$$

$$\frac{d\vec{v}_i}{dt} = \vec{F}_i, \quad (2)$$

where $\vec{v}_i$, $\vec{r}_i$ are velocities and coordinates of the ions, force $\vec{F}_i$ is combination of the Lorentz force and the ion-electron friction force:

$$\vec{F}_i = \vec{E} + \left[\vec{v}_i, \vec{H}\right] - \kappa \vec{j}/n, \quad (3)$$

where $\kappa = cm_e/eH_0\tau_{ei}$, $\tau_{ei}$ is characteristic ion-electron collision time [7].

The equation for the currents is:

$$\vec{j} = n_i\vec{V}_i - n_e\vec{V}_e, \quad (4)$$

where $\vec{V}_e$ is the electron velocity.

The mean ion velocity $\vec{V}_i$ and the ion density $n_i$ are integrals of distribution function $f_i(t, \vec{r}, \vec{v})$:

$$n_i(\vec{r}) = \int f_i(t, \vec{r}, \vec{v}) d\vec{v}, \quad (5)$$

$$\vec{V}_i(\vec{r}) = \frac{1}{n_i(\vec{r})} \int f_i(t, \vec{r}, \vec{v}) \vec{v} d\vec{v}. \quad (6)$$

We use the massless approximation of the electron component:

$$\vec{E} + \left[\vec{V}_e, \vec{H}\right] + \nabla p_e/2n_e - \kappa \vec{j}/n = 0 \quad (7)$$

The electron temperature $T_e$ is defined by the heating due to the ion-electron friction and the compression/expansion:

$$n \left( \frac{\partial T_e}{\partial t} + (\vec{V}_e \nabla) T_e \right) = (\gamma - 1) \left( 2\kappa \frac{\vec{j}^2}{n_e} - p_e \text{div} \vec{V}_e \right), \quad (8)$$
$p_e = n_e T_e$ is the pressure of the electrons, the adiabatic index $\gamma = 5/3$, $\kappa = \text{const}$.

We consider the displacement currents are negligible and for the computation of the electric field $\vec{E}$ and magnetic field $\vec{H}$ use Maxwell’s equations in the following form:

$$\frac{\partial \vec{H}}{\partial t} = -\text{rot} \vec{E},$$

(9)

$$\text{rot} \vec{H} = \vec{j}.$$

(10)

At the initial time moment the electric field $\vec{E} = 0$, magnetic field is determined using Poisson equation for the colis potential. We consider particles reflect from the walls of the trap and leave the trap through its ends. The perturbations of all values at the borders are considered negligibly small.

3. Algorithms

We apply particle-in-cell method to solve the equations: the real ions are grouped in model particles with coordinates, velocities and charges $r, z, v_r, v_x, v_z, q$ defined on Lagrangian mesh. All other variables from the equations are defined on the Eulerian mesh. The spatial grids are taken shifted for different functions: $H_r(i - 1/2, k), H_\phi(i, k), H_z(i, k - 1/2), E_r(i - 1/2, k - 1/2), E_\phi(i - 1/2, k), E_z(i - 1/2, k - 1/2), V_{r_r}(i, k - 1/2), V_{r_\phi}(i - 1/2, k - 1/2), V_{r_z}(i - 1/2, k), V_{z_r}(i - 1/2, k), V_{z_\phi}(i - 1/2, k), V_{z_z}(i - 1/2, k), j_r(i, k - 1/2), j_\phi(i - 1/2, k - 1/2), j_z(i - 1/2, k), p_e(i - 1/2, k - 1/2), n_e(i - 1/2, k - 1/2)$ [8]. On each time step (Fig. 1) the new ion velocities are calculated from the values of electro-magnetic fields in the corresponding grid nodes [9] and this stage represents the switching from the computation on the Eulerian mesh to the ones on the Lagrangian mesh. The switch from the particles to the grid is the ion density and the mean ion velocities computations.

The higher number of model particles in cell provides a better particle-in-cell method accuracy [10]. Apart from the beam ions the hybrid model implies computations of background ions, because the background density must be nonzero. An increase of the number of grid cells leads to higher computational costs not only due to the number of grid nodes, but due to the correspondingly increasing total number of particles. The conditional stability of the numerical algorithm limits the time step value and increases the computation times either. In practice, for the grid with $80 \times 240$ nodes the time step $\tau = 2.5 \cdot 10^{-6}$ was used. Theoretical estimates for the

![Algorithm block-scheme](image-url)
formation times of the stationary shape of the “bubble” may be achieved at times $\sim 10^4 \omega_{ci}^{-1}$, which is $10^4$ in dimensionless time units, and leads to prolonged computations.

Parallelization is a gainful way to accelerate the computations. In our code we apply mixed decomposition, dividing the computational domain along z-axis and assigning each subdomain and the particles in the subdomain to a group of processes (Fig. 2) [11]. Within each group the particles are divided almost uniformly due to the uniform distribution during injection and uniform particle exchanges among neighbour processes. In every group we mark one main process responsible for the Eulerian stage computations (currents, electron velocities, electric and magnetic fields, electron temperature), sending the grid data within their group and gathering the ion density and mean velocities from other groups processes. Other processes in group are marked slaves and perform computations of the Lagrangian stage and the both transitions from the grid to the particles and from the particles to the grid. The algorithm allowed to achieve the computation times smaller few times in comparison to a decomposition with one process per group, and tens times smaller in comparison with monoprocessor algorithm. In standard computations the central groups have the maximal number of processes, the number falls with distance from the center and the configuration is very effective for the initial stage of the evolution.

However, any static distribution of the processes among groups cannot represent an optimal configuration for the non-linear changing ion distribution: the distribution of the background ions in space is uniform at the initial moment, but the injected ions expel the magnetic field and the background ions, the “bubble” radius arises, the particles move to the mirrors and may reflect from them, and they have to be assigned to another processes. We introduce a dynamic balance load based on reassigning of process to another group depending on the average number of particles in group.

Another kind of parallelization is vectorization, when a single instruction is applied to multiple data providing highly efficient computations. Using explicit numerical schemes allows automatic loop vectorization for the Eulerian stage computations. However, this stage is not a time consuming one and thus its vectorization insignificantly decreases the total computation time. The computation of the particle velocities and coordinates (particle push) together with the density and mean velocities computation take major portion of total computation time [11].

The transition from the Eulerian to Lagrangian stage usually includes a loop through particles for the computation of the forces, acting on each particle $j$, using bilinear interpolation to the particle position from nearest grid nodes $(i, k), (i, k + 1), (i + 1, k), (i + 1, k + 1)$. The dependence of indexes $(i, k)$ on $j$ leads to the indirect indexing and overheads higher than the speed-up due to vectorization.

Figure 2. Decomposition scheme.
to the vectorization. The standard computation of the ion velocities and densities includes a loop through particles, calculation of the four numbers of the nearest grid nodes and accumulating in the nodes the ion densities and the current densities on the corresponding grids and further division of the current densities on the densities [11]. In this case two particles may deposit in the same grid node and cause data dependence preventing the loop vectorization. Both cases require special algorithms to avoid the described problems [12], [13].

![Figure 3. Force interpolation.](image1)

![Figure 4. Density/charge deposition.](image2)

We draw attention to the ion density and current density computation. A standard algorithm includes a loop on the particles $j=1..N_p$, where $p$ is process number. For example, for the radial component, on every step of the loop the minimal indexes $i,k$ of the nodes of the cell containing the particle $j$, the distances between the particle position and the grid coordinates $s_r, s_z$, define the weights of the charge $q(j)$, so the values of the density $n_r$ and the currents $n_v r$ at the grid nodes are updated in the following way:

$$n_r(i, k) = n_r(i, k) + (1 - s_r)(1 - s_z)q(j)$$
$$n_r(i + 1, k) = n_r(i + 1, k) + s_r(1 - s_z)q(j)$$
$$n_r(i, k + 1) = n_r(i, k + 1) + (1 - s_r)s_zq(j)$$
$$n_r(i + 1, k + 1) = n_r(i + 1, k + 1) + s_r s_zq(j)$$

$$n_v r(i, k) = n_v r(i, k) + (1 - s_r)(1 - s_z)q(j)v_r q(j)$$
$$n_v r(i + 1, k) = n_v r(i + 1, k) + s_r(1 - s_z)q(j)v_r q(j)$$
$$n_v r(i, k + 1) = n_v r(i, k + 1) + (1 - s_r)s_z q(j)v_r q(j)$$
$$n_v r(i + 1, k + 1) = n_v r(i + 1, k + 1) + s_r s_z q(j)v_r q(j)$$

The subscript 0.5 of $s_z$ reminds that the grid is half step shifted in direction $z$. For enabling the loop vectorization we used algorithm, proposed in [14]. In this case an additional two-dimensional density $n_{jr}$ and currents $n_{vjr}$ arrays are introduced. The first index defines the cell vertex number $m$, the second index $icell_r$ represents the value in the cell with number $icell_r$. Since the particles can not deposit their charge to different cells, the data dependency is absent and vectorization may be enabled. In order to accommodate the loop into cache and accelerate computations the particles are divided into blocks of $lvect$ length. The loop through the particle number $jloc$ in each block includes the computation of one index $icell_r$ of the cell (not the nodes indexes), the distances $s_r(jloc), s_{z0.5}(jloc)$, the charge $Q(jloc)$ and velocity $Vr(jloc)$, and the vectorized loop through the cell vertex number $m$: 
Figure 5. Absolute values of the magnetic field.

\[ \Delta n_r = (I_r(m) - s_r(j)) \cdot (I_z(m) - s_{z0.5}(j)) \cdot S(m) \cdot q(j) \]
\[ \Delta n v_r = (I_r(m) - s_r(j)) \cdot (I_z(m) - s_{z0.5}(j)) \cdot S(m) \cdot q(j) \cdot v_r(j) \]
\[ n_{j_r}(m, icell_r) = n_{j_r}(m, icell_r) + \Delta n_r \]
\[ n_{v j_r}(m, icell_r) = n_{v j_r}(m, icell_r) + \Delta n v_r \]

enddo

Here \( I_r = (1, 0, 1, 0) \), \( I_z = (1, 1, 0, 0) \), \( S = (1, -1, -1, 1) \), and the arrays \( s_r, s_{z0.5}, Q, V r \) are of length \( lvect \). The loops for \( m=1..4 \) for assigning the values in the nodes \( n_r(i, k), n v_r(i, k) \) based on the values \( n_{j_r}(m, icell) \), \( n_{v j_r}(m, icell) \) in cell vertices require the computations on the grid and do not take much time for cases with high number of particles compared with the number of grid nodes.

For the \( \varphi \) and \( z \) components the shifted grids are used, and the arrays \( n_{j_r}, n_{v j_r}, n_z, n_{v j_z}, s_{r0.5}, s_z \) are introduced and computed by analogy with the described component. The algorithm does not require particle sorting.

4. Numerical results

For the numerical experiments the magnetic field \( H_0 = 2kG \), the background ion density \( n_0 = 4 \cdot 10^{12} cm^{-3} \) were used. The time is measured in units \( t_0 = 1/\omega_{ci} = 5.2 \cdot 10^{-8} sec \), \( \omega_{ci} = eH_0/cm_i \) is the ion gyrofrequency. The length unit is \( L_0 = c/\omega_{pi} = 11.4cm \), \( \omega_{pi} = \sqrt{4\pi n_0 e^2/m_i} = 2.6 \cdot 10^{10} sec^{-1} \) is the ion plasma frequency. The domain sizes are \( R_{max} = 4 \), \( Z_{max} = 12 \). The velocities are measured in Alfvén velocity \( V_A = H_0/\sqrt{4\pi m_i n_0} = 2.2 \cdot 10^8 cm/sec \), the average speed of the injecting ions is 0.5, the thermal beam speed is \( 8.9 \cdot 10^{-2} \).

Fig. 5 demonstrates the magnetic field values \(|H|\) in the domain at time moment \( t = 8 \). In the central region of the trap the cavity in the magnetic field is observed, within the bubble radius \( r=0.6 \) the absolute values of the magnetic field \(|H| \leq 1\%H_0 \). In Fig. 6 the background ion density at the same time moment \( t = 8 \) is presented. The high concentration of the background ions is located outside of the bubble boundary with high gradient of the magnetic field. The bubble layer corresponds to the high density of the injected ions. The formation of the diamagnetic bubble
corresponds to the the expelling of the background ions from central region and generating the cavity of crescent shape in the background ion density.

The simulations were performed on the grid with $120 \times 480$ nodes and required $1.6 \cdot 10^7$ time steps for $T=8$ ($h_r = 2.5 \cdot 10^{-2}$, $h_z = 2.5 \cdot 10^{-2}$, $\tau = 5 \cdot 10^{-7}$). In order to demonstrate the influence of the dynamic load balancing and the code vectorization the same physical parameters, spatial and time steps are taken. At the initial time moment the number of background ions per cell is 4, each time time step 1 particle appears at the domain. At the time $t=2$ there are $N = 4.3 \cdot 10^6$ beam particles, the number of background ions is $N_{bg} = 3 \cdot 10^6$. The particles do not leave the trap on the stage and at $t=8$ the number of them is $N = 16.3 \cdot 10^6$.

The tests were performed on Intel Xeon Phi 7290 processors, the code written in Fortran using MPI was compiled with Intel compiler, the option -axCORE-avx2 was used to enable the vectorization. Each process corresponds to one core of the processor, $np = 72$ cores were divided among $N_{groups} = 20$ and $N_{groups} = 48$ groups. In Fig. 7 and Fig. 8 the core distribution is shown for the static and the dynamic load balance at time $t=2$ with black and blue lines respectively.

| $N_{groups}$ | $N_{bg}$ | $N$ |
|--------------|----------|-----|
| 20           | 3 \cdot 10^6 | 16.3 \cdot 10^6 |
| 48           | 3 \cdot 10^6 | 16.3 \cdot 10^6 |

Table 1 demonstrates the dependence of maximal number of particles in core $N_{p_{max}}$ on the number of subdomains and balancing method for time moments $t=1$ and $t=2$.

In Fig. 9 and Fig. 10 the graphs demonstrate the dependence of the run times measured in hours of each step $T^* = 2 \cdot 10^5 \tau = 0.1$, on the time $t$. The rebalancing occurred every $T^*$ steps. The plain lines correspond to the computations without vectorization, the lines with points correspond to the vectorized computations, the red and green color correspond to the tests with
Table 1. Maximal number of particles in a core at \( t = 1 \) and \( t = 2 \).

| \( N_{\text{groups}} \) | Load balance | \( J_{p_{\text{max}}} \) at \( t = 1 \) | \( J_{p_{\text{max}}} \) at \( t = 2 \) |
|-----------------|-------------|-----------------|-----------------|
| 20 static       | 0.93 \( \cdot 10^5 \) | 1.68 \( \cdot 10^5 \) |
| 20 dynamic      | 0.38 \( \cdot 10^5 \)  | 0.74 \( \cdot 10^5 \) |
| 48 static       | 1.22 \( \cdot 10^5 \) | 1.73 \( \cdot 10^5 \) |
| 48 dynamic      | 0.82 \( \cdot 10^5 \)  | 1.37 \( \cdot 10^5 \) |

the dynamic load balance. The blue lines on the graph correspond to the static and the dynamic load balances. The times for the density computations are marked with green and purple colors. The total of the simulations are marked with black and red colors.

From the results of the numerical experiments the following observations can be made:

- The computations of the particle velocities and coordinates together with the density and currents computations take 87 – 92% of the total time for \( N_{\text{groups}} = 48 \) and 85 – 90% for \( N_{\text{groups}} = 20 \). The smaller \( N_{\text{groups}} = 20 \) leads to the bigger subdomains and higher run times of the Eulerian stage, the central groups contain higher number of cores (see Fig. 7, 8) and exchanges within the central groups take more time.

- The times for the case of 48 groups grow slower due to the faster particle leaving the thinner domains.

- The run time of the particle pusher for the dynamic load balance is 2.2 times smaller than for the case of the static one for \( N_{\text{groups}} = 20 \) and 1.5 times smaller for \( N_{\text{groups}} = 48 \), it is almost proportional to the ratio of the maximal numbers of particles in a core \( N_{p_{\text{max}}} \).

- Since the push procedure is not vectorized, the gain in total time achieved from the vectorization gain depends on the ratio of computation times for the pusher and the deposition procedure.
The non-vectorized density deposition takes 41% of the time needed for the particle motion procedure. The vectorized density deposition run time is 2.7 times smaller than the non-vectorized one in case of $N_{\text{groups}} = 48$ and static load balance. The acceleration reflects the relatively high number of particles for the grid of sizes. In case of $N_{\text{groups}} = 20$ the grid of each core has almost doubled size, the dynamic balance leads to a small number of particles in cores and the vectorized algorithm becomes less effective: the ratio of run times for the vectorized and usual deposition procedures is 1.64. The red and green lines in Fig. 9 demonstrate the same effect: the vectorization leads to smaller times only for times ($T^* \geq 1$) when the number of particles is high enough. However, despite of such small effectiveness of the vectorization for rebalanced cases, the total runtime of the deposition until $t = 2$ for $N_{\text{groups}} = 20$ is only 1.1 times higher than for $N_{\text{groups}} = 48$ due to a good load balance.

The combination of the dynamic load balance and the vectorization for $T^*$ at $t = 2$ provides $\times2.6$ speed-up in deposition procedure for $N_{\text{groups}} = 20$ and $\times3.5$ for $N_{\text{groups}} = 48$.

The combination provides $\times2.2$ speed-up of the program runtime $T = 20T^*$ for 20 groups and $\times1.6$ for 48 groups, the total computation times are 20 and 29 hours for 20 and 48 subdomains respectively.

5. Conclusion

The algorithm for numerical modelling of the plasma dynamics in open magnetic trap and the results of numerical experiments for the problem are presented. The algorithm is based on MHD approach for the electron plasma component and kinetic approach for the ion one. The particle-in-cell method is used together with dynamic load balancing depending on the number of particles in a process for the domain-particle decomposition. The density and current density computation is a time consuming procedure and its vectorization demonstrated $\sim 1.5-3$ times speed-up. Although the vectorization effectiveness decreases with a good load balancing due to the decreasing of number of particles in process, the load rebalancing is an effective way to decrease the computations times in case of highly non-uniform particle distribution in space and time. The combination of vectorization and load rebalance allows significantly to accelerate computations and is more effective for higher number of particles per process. The simulation results demonstrate the formation of the diamagnetic “bubble” with the expelling of the background ions and the magnetic field from their original area.

Acknowledgements

The work is supported by Russian Science Foundation (project N 19-71-20026). The computations we re performed on Siberian Supercomputer Center cluster (ICM&MG SB RAS, Novosibirsk).

References

[1] Beklemishev A 2016 Phys. Plasmas 23 082506
[2] Ivanov A A and Prikhodko V V 2017 Phys.-Usp 60 509
[3] Bagryansky P A et al. 2016 AIP Conf. Proc. 1771 030015
[4] Caprioli D, Pop A R and Spitkovsky A 2015 Astrophys. J. Lett. 798 L28
[5] Vshivkova L and Dudnikova G 2017 Numerical Analysis and Its Applications: 6th International Conference, NAA 2016, Revised Selected Papers Lecture Notes in Computer Science 10187 pp 737–743
[6] Hockney R W and Eastwood J W 1988 Computer simulation using particles (Bristol: Hilger, 1988)
[7] Braginskii S 1965 Rev. Plasma Phys. 1 205–311
[8] Birdsal C and Langdon A 1985 Plasma physics via computer simulation (McGraw-Hill)
[9] Boris J 1970 Proc. Fourth Conference on Numerical Simulations of Plasmas, Washington D.C., USA 3–67
[10] Lotov K V, Timofeev I V, Mesyats E, Snytnikov A and Vshivkov V 2014 Phys. of Plasmas 22
[11] Boronina M A, Chernykh I G, Genrikh E A and Vshivkov V A 2019 J. Phys.: Conf. Ser. 1336 012017
[12] Nishiguchi A, Orii S and Yabe T 1985 J. Comput. Phys. 61 519 – 522 ISSN 0021-9991
[13] Horowitz E J 1987 J. Comput. Phys. 68 56 – 65 ISSN 0021-9991
[14] Vincenti H, Lobet M, Lehe R, Sasanka R and Vay J L 2017 Comp. Phys. Comm. 210 145 – 154