Efficient algorithm for high fidelity collisional charged particle beam dynamics

A. Al Marzouk,[*] H. D. Schaumburg, S. Abeyratne, and B. Erdelyi
Department of Physics, Northern Illinois University, DeKalb, Illinois 60115, USA

(Received 4 March 2021; accepted 22 June 2021; published 13 July 2021)

High fidelity charged particle beam dynamics, especially if collisional, require specialized numerical methods. The challenges include the ability to deal with very large particle numbers, long-range electromagnetic forces, and vast spatial and timescales. We developed a novel algorithm to address these challenges. The main characteristics of the algorithm are Strang splitting to separate near and far forces, the fast multipole method to lower the computational cost of the far region, and the Simò integrator to capture all close encounters efficiently in the near region. The algorithm is fully adaptive both in space and time, while maintaining symplecticity to machine precision. Its performance is illustrated with two challenging examples from nonlinear multiparticle beam dynamics, including the first electron cooling simulations based on first principles.

DOI: 10.1103/PhysRevAccelBeams.24.074601

I. INTRODUCTION

Charged particle beams are ensembles of close-by electrons or ions in directed motion. Their science and applications underlie vast areas of science, technology, and industrial processes [1]. Hence, it is important to be able to accurately and efficiently model, simulate, and optimize their behavior. The typical beam consists of a very large number of particles, but much smaller than Avogadro's number. Most beams contain less than one trillion particles and some contain just a few thousand. Also, particles within beams interact through long-range Coulomb forces and are subjected to externally imposed electromagnetic fields. Moreover, the typical application of beams requires understanding the qualitative and quantitative long-time dynamical behavior, including low levels of beam loss to their surroundings. Therefore, the associated numerical methods are very challenging due to the large particle numbers that interact pairwise, vast spatial and timescales present in the dynamics, and maintenance of the symmetries present in the time-continuous system, the most important being symplecticity.

The modeling and simulation of these complex systems can take several forms, based on the level of accuracy needed in the particular application. One way to categorize these methods is based on whether the method is collision-less or captures collisions. If warranted to use, the collision-less methods are less challenging and can deal with larger particle numbers and longer simulation times. The main idea is to coarse-grain a distribution of particles to be represented by their average fields. Then, this average field can be applied to a test particle as a smooth external field. Representatives of these methods are the Vlasov solvers, which model the single-particle phase space distribution's propagation in time by the Vlasov equation [2], the particle-in-cell (PIC) methods that solve the Poisson equation on a mesh [3], the softened N-body solvers, which use a modified Coulomb force to reduce the effect of the artificial close encounters [4], and the so-called basis function methods, which model the beam distribution as a series in some basis functions combined with the moment method [5]. These methods are called collision-less mean-field methods. Both the force computation and time stepping complexity are reduced for these methods.

On the other hand, methods that accurately capture collisions are much more challenging. Clearly, these methods need algorithms that take into account particle-to-particle interactions directly in full detail. If in addition one would like to preserve the geometric features of the time-continuous system in the discretization, the feat seems almost impossible. Indeed, the well-known Hamiltonian formulation of a single classical point-charge in external electromagnetic fields that satisfy Maxwell equations cannot be extended to a collection of relativistic, interacting particles [6]. There may be workaround solutions by reconsidering the self-fields as external, and approximating the dynamics by an iterative step-wise solution of fields and distribution advancement in time, respectively. Here, we are interested in collisional methods, but we take a different

[*]afnan.almarzouk1@gmail.com

approach, which works well for charged particle beams; namely, we work in a boosted frame where the relative motion is nonrelativistic [7]. This is often possible by the very definition of particle beams, and it simplifies an electrodynamics problem to a combined electrostatic problem accompanied by relativistic transformations. In this approach, the straightforward Newtonian method is ideal, where the particles interact merely by the Coulomb force. Equivalently, the instantaneous scalar potential computation is sufficient to describe the dynamics.

To this end, the naive, direct, particle-to-particle method is one solution. However, since it scales quadratically with the number of particles [8], it is unfeasible for large particle numbers. Modifications of some mean field methods are available that more or less *ad hoc* capture some effects of the collisions. Examples are the Boltzmann, Fokker-Planck or Langevin approaches [9–11], the particle-particle particle-mesh (P^3M) method and its variants [12], or the analytical binary collision model [13]. While being the most accurate method, indeed numerically exact, the direct method is also the most inefficient. In order to capture the collisional effects accurately and efficiently, new methods were needed. Currently, there are several so-called fast methods for force computation that scale better than quadratically with the number of particles. Most of these methods can be categorized as hierarchical space subdivision methods and include tree methods such as Barnes-Hut [14], cluster methods [15], and methods that combine both into fast multipole methods (FMM) [16]. The FMM achieves linear scaling and offers the optimal solution for the force computation, at least in the high accuracy regime [17]. We developed a novel fully adaptive FMM algorithm that is well suited to be incorporated in collisional dynamics problems and that is also applicable to softened Coulomb potentials [18].

However, choosing the optimal time stepping method is still an outstanding issue, even if the force can be computed efficiently at each moment in time. There is a vast literature on numerical integrators [19], including for N-body problems [20], but how to pick the optimal integrator is still far from clear [21]. Based on a theorem of Simò [22] and aided by differential algebraic methods [23], we previously developed a collisional numerical integrator, the Simò integrator, with some provably optimal behavior [24]. This integrator is variable order and adaptive with automatic selection of particle-by-particle optimal order and time step to minimize computational cost given an *a priori* user-selected tolerance for error. It also has dense output, eliminating step rejection completely. The drawback of the integrator is its quadratic scaling with the particle number.

Therefore, it is natural to try to combine the FMM for the force computation with the Simò integrator for the time stepping. Indeed, this is straightforward. However, it would require an FMM call per smallest time step (the time step associated with the particle that has the smallest time step at each simulation time moment). The solution we propose is to use splitting [19] to alleviate the problem. The splitting is into the sum of two parts: near and far forces. The full simulation is integrated with a second order accurate Strang splitting [25] with a relatively long and constant time step. The far part is solved exactly, with the approximate forces given by the FMM. This is justified by the fact that relatively large position changes of particles in the far region modify the far fields little, and the differential equations of the far region can be solved analytically. This allows the reduction of the number of FMM calls by a significant margin. By definition, particles from the far region cannot have a close encounter with a given particle in the allotted time. Particles in the near region are integrated with the Simò integrator, and all possible close encounters are captured. This piece of the solution cannot be solved analytically exactly, but the error can be reduced to machine precision. Since the motion of charged particles in electromagnetic fields is Hamiltonian, another main advantage of applying Strang splitting is the preservation of symplecticity, a well-known geometrical property of the time-continuous flow that needs to be preserved in long-term simulations [19,26]. While there are many studies on the use of symplectic integrators for beam dynamics such as [27–32], it is exceedingly difficult to preserve symplecticity with adaptive integrators [33] and practically impossible for beams consisting of millions or billions of particles. In addition, it is well-known from the theory of splitting methods of symplectic integration that the combination of exact and numerical flows do not produce symplectic flows in general [19]. Hence, the numerical piece of the solution needs to be solved to machine precision, resulting in preservation of symplecticity of the total solution to machine precision. The total computational cost is reduced by a significant factor in this process. According to this discussion, splitting is still necessary for an efficient algorithm regardless of symplecticity. For applications where maintaining symplecticity is not essential, the accuracy of the Simò integrator can be set to lower than machine precision and give solutions that are not symplectic.

In this paper, we present our collisional numerical method, as described above, that we termed: particles' high-order adaptive dynamics (or PHAD). Our preliminary work toward the development of PHAD was presented in [34–37]. The main components of PHAD are: the Strang splitting method, the fast multipole method, and the Simò integrator. The resulting algorithm is an accurate and efficient collisional simulation of charged particle beams in external electromagnetic fields that is symplectic to machine precision and is provably the best in reaching an *a priori* set error level with minimized computational cost. We also illustrate its performance with applications to ultracold multibeamlet dynamics and the first electron cooling simulations based on a microscopic approach.

In the remaining sections of the paper, we describe the main components of the PHAD algorithm: the splitting method in Sec. II; the fast multipole method and the long-range interactions in Sec. III; the Simò integrator and short-range interactions in Sec. IV; and the algorithm's implementation in Sec. V. We also provide several challenging examples taken from beam physics in Sec. VI; we finally conclude with a brief summary in Sec. VII.

II. EQUATIONS OF MOTION AND SPLITTING METHOD

The motion of the charged particles in a beam is governed by the Coulomb force that arises from their mutual interactions (self-force) and Lorentz force of the external electromagnetic forces that guide and accelerate the beam in particle accelerators. Each particle in the beam experiences a force that is the sum of the self Coulomb force and the external Lorentz force.

Assuming that the longitudinal motion of the charged particle beam is along the z-direction and that the particles motion could be relativistic, the particles are observed to move at a velocity $v_z = \beta c$ in the laboratory (lab) frame. In this frame, the external electromagnetic forces are given and can be computed for an individual particle independently from other particles' positions. In the beam rest frame, the particles are regarded as stationary and the magnetic field is practically zero. Therefore, to derive the equations of motion, we calculate the electrostatic self-forces in the rest frame and use the Lorentz transformation to transform them to the lab frame to be combined with the external forces.

For a particle i of charge q_i and mass m_i, the resulting equations of motion for position and momentum are

$$\dot{\mathbf{x}}_i = \frac{c(p_{x_i}\mathbf{i} + p_{y_i}\mathbf{j} + p_{z_i}\mathbf{k})}{(m_i^2c^2 + p_{x_i}^2 + p_{y_i}^2 + p_{z_i}^2)^{1/2}} \tag{1}$$

and

$$\dot{\mathbf{p}}_i = q_i\left(\frac{1}{4\pi\epsilon_0}\sum_{\substack{j=1\\j\neq i}}^{N}\frac{q_j((x_i-x_j)\mathbf{i} + (y_i-y_j)\mathbf{j} + \gamma^2(z_i-z_j)\mathbf{k})}{\gamma[(x_i-x_j)^2 + (y_i-y_j)^2 + \gamma^2(z_i-z_j)^2]^{3/2}} + \mathbf{E} + \mathbf{v}_i\times\mathbf{B}\right), \tag{2}$$

where γ is the Lorentz relativistic factor, c is the speed of light in vacuum, ϵ_0 is the vacuum permittivity, and $\mathbf{E}$ and $\mathbf{B}$ are the external electric and magnetic fields, respectively. A detailed derivation of the equations of motion is provided in [37].

Since the equations of motion (1) and (2) deal with very small quantities such as particles' charges and masses, we scale the equations by some physical quantities in order to preserve numerical stability. We define the charge factor $n_i = q_i/q$ where q is the elementary charge, and the mass factor $f_i = m_i/m$ with m being the proton's mass. For the other variables, we employ the accent "$\hat{}$" to denote a scaled variable throughout as follow: $\hat{t} = tc$, $\hat{\mathbf{p}}_i = \mathbf{p}_i/mc$, and $\hat{\mathbf{v}}_i = \mathbf{v}_i/c$. Therefore, we arrive to the following scaled equations of motion

$$\dot{\mathbf{x}}_i = \frac{\hat{p}_{x_i}\mathbf{i} + \hat{p}_{y_i}\mathbf{j} + \hat{p}_{z_i}\mathbf{k}}{(f_i^2 + \hat{p}_{x_i}^2 + \hat{p}_{y_i}^2 + \hat{p}_{z_i}^2)^{1/2}}, \tag{3}$$

and

$$\frac{d\hat{\mathbf{p}}_i}{d\hat{t}} = \frac{qn_i}{mc^2}\left(\frac{q}{4\pi\epsilon_0}\sum_{\substack{j=1\\j\neq i}}^{N}\frac{n_j((x_i-x_j)\mathbf{i} + (y_i-y_j)\mathbf{j} + \gamma^2(z_i-z_j)\mathbf{k})}{\gamma[(x_i-x_j)^2 + (y_i-y_j)^2 + \gamma^2(z_i-z_j)^2]^{3/2}} + \mathbf{E} + c\hat{\mathbf{v}}_i\times\mathbf{B}\right). \tag{4}$$

A. Strang splitting

In a beam consisting of N particles, there are $6N$ equations to be solved. To write these $6N$ equations in a more compact form, we define the vector

$$\mathbf{Y} = [x_1 y_1 z_1 p_{x_1} p_{y_1} p_{z_1} \ldots x_N y_N z_N\ p_{x_N} p_{y_N}\ p_{z_N}]^{\mathrm{T}}$$

in $\mathbb{R}^{6N}$. The function $\mathbf{F}$ in the initial value problem (IVP)

$$\begin{cases}\dot{\mathbf{Y}} = \mathbf{F}(\mathbf{Y})\\ \mathbf{Y}(0) = \mathbf{Y}_0\end{cases} \tag{5}$$

is given by the right-hand sides of (3) and (4) above.

Our algorithm is based upon applying Strang splitting [19, pp. 47–50] [25] to the equations of motion (5).

Theorem 1: (Strang splitting) Suppose the initial value problems

$$\begin{cases}\dot{\mathbf{y}} = \mathbf{g}^{[1]}(\mathbf{y}) \\ \mathbf{y}(0) = \mathbf{y}_0\end{cases} \quad \text{and} \quad \begin{cases}\dot{\mathbf{y}} = \mathbf{g}^{[2]}(\mathbf{y}) \\ \mathbf{y}(0) = \mathbf{y}_0\end{cases}$$

have exact solutions $\phi_t^{[1]}(\mathbf{y}_0)$ and $\phi_t^{[2]}(\mathbf{y}_0)$ for any choice of $\mathbf{y}_0 \in \mathbb{R}^n$. Then

$$\phi_t = \phi_{t/2}^{[2]} \circ \phi_t^{[1]} \circ \phi_{t/2}^{[2]}(\mathbf{y}_0) \tag{6}$$

is a second order approximation of the solution to the initial value problem

$$\begin{cases}\dot{\mathbf{y}} = \mathbf{g}^{[1]}(\mathbf{y}) + \mathbf{g}^{[2]}(\mathbf{y}) \\ \mathbf{y}(0) = \mathbf{y}_0\end{cases}.$$

For a fixed time step h, iterating (6) gives an approximate solution

$$\underbrace{\phi_{h/2}^{[2]} \circ \phi_h^{[1]} \circ \phi_{h/2}^{[2]} \circ \phi_{h/2}^{[2]} \circ \phi_h^{[1]} \circ \phi_{h/2}^{[2]} \circ \cdots \circ \phi_{h/2}^{[2]} \circ \phi_h^{[1]} \circ \phi_{h/2}^{[2]} \circ \phi_{h/2}^{[2]} \circ \phi_h^{[1]} \circ \phi_{h/2}^{[2]}}_{k \text{ compositions of } \phi_{h/2}^{[2]} \circ \phi_h^{[1]} \circ \phi_{h/2}^{[2]}}(\mathbf{y}_0) \tag{7}$$

at time $t = kh$. However, the differential equations may be split in different ways at each time step. We employ an additional superscript to indicate different splittings

$$\phi_{h/2}^{[2,k]} \circ \phi_h^{[1,k]} \circ \phi_{h/2}^{[2,k]} \circ \phi_{h/2}^{[2,k-1]} \circ \phi_h^{[1,k-1]} \circ \phi_{h/2}^{[2,k-1]} \circ \cdots \circ \phi_{h/2}^{[2,2]} \circ \phi_h^{[1,2]} \circ \phi_{h/2}^{[2,2]} \circ \phi_{h/2}^{[2,1]} \circ \phi_h^{[1,1]} \circ \phi_{h/2}^{[2,1]}(\mathbf{y}_0). \tag{8}$$

In order to employ Strang splitting to the IVP in (5) according to Theorem 1, the right-hand sides of (3) and (4) must be expressed as sums. The right-hand side of (3) may be expressed as the sum of

$$\dot{\mathbf{x}}_i^{[1]} = \frac{\hat{p}_{x_i}\mathbf{i} + \hat{p}_{y_i}\mathbf{j} + \hat{p}_{z_i}\mathbf{k}}{(f_i^2 + \hat{p}_{x_i}^2 + \hat{p}_{y_i}^2 + \hat{p}_{z_i}^2)^{1/2}} \tag{9}$$

and

$$\dot{\mathbf{x}}_i^{[2]} = 0. \tag{10}$$

We choose to express the right-hand side of (4) as a sum which depends upon how our algorithm partitions the configuration space. Described in Sec. III, we used the fast multipole method (FMM) to partition the space containing the N particles into boxes (or cubes). Each box and all those adjacent to it form a neighborhood. In the FMM, expansions are calculated for each neighborhood that give the fields at each point within the neighborhood due to the particles outside the neighborhood. To make use of this expansion, the sum over particle indices on the right-hand side of (4) may be split among those indices inside and outside the neighborhood containing the particle i. Throughout, we denote the indices of particles inside the neighborhood whose central box contains particle i by N_i, and the indices of particles outside that neighborhood by N_i^c. With this notation, (4) is expressible as the sum of

$$\frac{d\hat{\mathbf{p}}_i^{[1]}}{d\hat{t}} = \frac{qn_i}{mc^2}\left(\frac{q}{4\pi\epsilon_0}\sum_{\substack{j\in N_i\\ j\neq i}}\frac{n_j((x_i - x_j)\mathbf{i} + (y_i - y_j)\mathbf{j} + \gamma^2(z_i - z_j)\mathbf{k})}{\gamma[(x_i - x_j)^2 + (y_i - y_j)^2 + \gamma^2(z_i - z_j)^2]^{3/2}} + \mathbf{E} + c\hat{\mathbf{v}}_i \times \mathbf{B}\right) \tag{11}$$

and

$$\frac{d\hat{\mathbf{p}}_i^{[2]}}{d\hat{t}} = \frac{qn_i}{mc^2}\frac{q}{4\pi\epsilon_0}\sum_{j\in N_i^c}\frac{n_j((x_i - x_j)\mathbf{i} + (y_i - y_j)\mathbf{j} + \gamma^2(z_i - z_j)\mathbf{k})}{\gamma[(x_i - x_j)^2 + (y_i - y_j)^2 + \gamma^2(z_i - z_j)^2]^{3/2}}. \tag{12}$$

Thus, the splitting our algorithm employs is $\mathbf{F} = \mathbf{F}^{[1]} + \mathbf{F}^{[2]}$, where $\mathbf{F}^{[1]}$ is given by the right-hand sides of (9) and (11), and $\mathbf{F}^{[2]}$ is given by the right-hand sides of (10) and (12).

Note that $\mathbf{F}^{[1]}$ depends on the particles whose positions are near particle i, while $\mathbf{F}^{[2]}$ depends on the particles whose positions are relatively far from particle i. For this reason, we call $\dot{\mathbf{Y}} = \mathbf{F}^{[1]}(\mathbf{Y})$ and $\dot{\mathbf{Y}} = \mathbf{F}^{[2]}(\mathbf{Y})$ the near and far equation, respectively. To apply Strang splitting, several initial value problems of the form

$$\begin{cases}\dot{\mathbf{Y}} = \mathbf{F}^{[1]}(\mathbf{Y}) \\ \mathbf{Y}(0) = \mathbf{Y}_0\end{cases} \quad \text{and} \quad \begin{cases}\dot{\mathbf{Y}} = \mathbf{F}^{[2]}(\mathbf{Y}) \\ \mathbf{Y}(0) = \mathbf{Y}_0\end{cases} \tag{13}$$

must be solved at discrete values of t.

A very important implication of the splitting method described above is related to the preservation of geometric properties of the time-continuous system. From the point of view of an arbitrary particle within the charged particle beam, the full problem can be described as a Hamiltonian system with an explicitly time-dependent Hamiltonian. The time-dependence comes from the self-fields, which are not known *a priori*, and possibly the externally prescribed electromagnetic fields. The coordinates used, position and mechanical momenta, are canonical in regions of vanishing vector potential. We can make the assumption that the dynamics' endpoints are in such regions in order to make the dynamics symplectic. Otherwise, the canonical momenta should be used instead of the mechanical momenta, but even in this case the symplectic flow will be merely sandwiched between two simple kick maps that do not alter the characteristics of the dynamics. It is well-known that discretization with a relatively large, constant time step of this flow in general preserves important properties of the continuous system over long-time integrations by not introducing nonphysical effects and having an effective Hamiltonian close to the exact one. Splitting methods are one way of performing symplectic integration with all the advantages described above. However, all components of the splitting must be symplectic in order for their composition to maintain the advantages. In our case, there are two component maps. One can be evaluated exactly, giving exactly symplectic maps by construction. The other component is not possible to solve exactly in closed form and it is difficult to approximate numerically efficiently with symplectic integrators. Therefore, we chose to develop the most efficient general integrator that allows us to attain numerical precision and hence numerical symplecticity with a minimum of computational effort. These developments are summarized in the following sections.

III. FAST MULTIPOLE METHOD AND THE LONG-RANGE INTERACTIONS

The celebrated fast multipole method (FMM) was developed in 1987 by Greengard and Rokhlin [16,38]. It allows for an efficient computation of the Coulomb potential of discrete charges that can be evaluated at the location of a test charge. Our particular choice of the FMM is due to some of its main properties [17] that makes it the most suitable for our PHAD algorithm. The FMM can achieve the best possible asymptotic scaling in terms of runtime and memory which is linear with the number of particles. It has a tunable, high accuracy with a guaranteed priori error bounds. Moreover, it has a high arithmetic intensity that allows for efficient parallelization now and in the exascale era. Another feature important for PHAD is the structure of the FMM that can be efficiently combined with a time propagator for dynamics simulations. To be efficiently combined with the Simò integrator, we have employed our fast multipole method that is described in detail in [37,18]. It is an adaptive multilevel FMM method using a Cartesian coordinate system with optimal data structures. One of its key features is that the translation operators are treated as function compositions through the use of Differential Algebra.

Our adaptive multilevel FMM uses an octree type data structure to partition the spatial domain—each box is subdivided into eight children. The adaptivity of the FMM is achieved by subdividing each box (parent) into child boxes when the number of particles it contains exceeds a specified value q, which we call the clustering parameter. These subdivisions are done after scaling the space containing the entire particle distribution into a unit cube, the root box. Considering the root box to be at level $l = 0$, the eight congruent boxes resulting from the first subdivision will be at level $l = 1$ and so on for the following subdivisions. Any boxes that do not contain any particles are discarded.

In the FMM context, sources and targets refer to the set of charged particles that generate the electric potential and the set of particles (locations) where this potential is evaluated, respectively. The collection of a target box and the boxes at the same level that share a side or a vertex with it (its neighbors) is called a neighborhood. At the same level, any two boxes that are separated by at least one box are regarded as well separated. The interaction list of a target box does not include its own neighbors but it includes the child boxes of the neighbors of its parent.

Computationally, the subdivisions are described by an octree type data structure where a collection of boxes form a tree if they are linked by parent-child relationships. The D-tree is a rooted tree that describes the hierarchy between parent and child boxes containing targets. On the other hand, the C-trees describe the hierarchy between boxes containing sources, the interaction lists of the target boxes. The C-trees are formed by removing the root box (level $l = 0$) and the $l = 1$ boxes from the D-tree; the boxes of each C-tree must contain at least one source. Hence, the root box for each of these trees is its level $l = 2$ box. The process of selecting the D-boxes makes them appropriately fall into a single tree, while the process of selecting the C-boxes usually results into a set of disconnected trees. Since the collection of trees of the same type is called a forest, a C-forest is formed by the collection of the C-trees while there is no D-forest. We refer the reader to [37] for visual illustrations of the FMM subdivisions and data structure.

The electric potential generated by a source particle at (x_0, y_0, z_0) and evaluated at a target point (x, y, z) is of the form

$$V(x,y,z) = \frac{q}{4\pi\epsilon_0}\frac{1}{\|\mathbf{x}-\mathbf{x}_0\|} = \frac{q}{4\pi\epsilon_0}\frac{1}{\sqrt{(x-x_0)^2+(y-y_0)^2+\gamma^2(z-z_0)^2}}.$$

Assuming that there are a set of sources N_i^c in a given box, their electric potential at the center of the box (x_0, y_0, z_0) is

$$V(x_0, y_0, z_0) = \frac{q}{4\pi\epsilon_0}\sum_{j\in N_i^c}\frac{n_j}{\sqrt{(x_j-x_0)^2+(y_j-y_0)^2+\gamma^2(z_j-z_0)^2}}.$$

Using differential algebra (DA), described in Appendix A, the multipole expansion of this potential at the center of the box (x_0, y_0, z_0) can be easily calculated and has to be performed only once at the highest level. The best-known DA engine in accelerator physics is COSY INFINITY [39]; we used it as the basis for our implementation and performance tests. In the framework of DA, the multipole expansion of the potential is represented by a truncated Taylor series that converges if the distance between the source and the target $r = \|\mathbf{x}_j - \mathbf{x}_0\|$ is larger than the box's side length. We introduce the following DA variables

$$d_x = \frac{x-x_0}{r^2}, \qquad d_z = \frac{z-z_0}{r^2},$$
$$d_y = \frac{y-y_0}{r^2}, \qquad d_r = \frac{1}{r}, \tag{14}$$

which have to be small in order for the expansion to be valid in a region far from the neighborhood of the box. The potential becomes

$$V(x,y,z) = \frac{qd_r}{4\pi\epsilon_0}\sum_{j\in N_i^c}\frac{n_j}{\sqrt{1+\|\mathbf{x}_j-\mathbf{x}_0\|^2 d_r^2 - 2(\mathbf{x}_j-\mathbf{x}_0)\cdot\mathbf{d}}} = \frac{q}{4\pi\epsilon_0}d_r V_m, \tag{15}$$

where $\mathbf{d} = (d_x, d_y, d_z)$ and the multipole expansion of the potential is described by V_m.

Once the octree subdivisions are performed and the data structuring is done, the FMM performs an upward pass and a downward pass to translate the multipole expansion V_m. In the upward pass, the multipole expansion of the potential is calculated at the center of each box at the highest level in each C-tree. Moving these expansions toward the root of each C-tree is accomplished by recursively applying the multipole-to-multipole (M2M) operator $\mathcal{T}_{M2M}$ which translates V_m centered in the child box to the multipole expansion V_m' centered in the parent box. By the use of DA, this translation is computed as the composition: $V_m' = V_m \circ \mathcal{T}_{M2M}$. At each subsequent level, the translated multipole expansions from each child box are summed to give the total multipole expansion of their parent box. Recursive application of the M2M operator is performed until all expansions have been collected at level $l = 2$ for each C-tree.

The downward pass starts from level $l = 2$ to the highest level through the D-tree. At each successive level, the accumulated V_m' is translated to a well-separated box at the same level (interaction list) via the multipole-to-local (M2L) operator $\mathcal{T}_{M2L}$ to get V_l', computed as the composition: $V_l' = V_m' \circ \mathcal{T}_{M2L}$. This expansion V_l' is added to the box's existing local expansion. Next, the total local expansion of a box is translated to its child boxes in the D-tree. Using the local-to-local (L2L) operator $\mathcal{T}_{L2L}$, this translation is achieved via the composition: $V_l = V_l' \circ \mathcal{T}_{L2L}$. Recursive application of the L2L operator is performed through the D-tree until the highest level is reached. Additional description and illustration of the FMM operators can be found in [37,18].

As a result of the hierarchical subdivision and the categorization of the computational domain, the contributions to the electric potential is divided into near and far regions. The near region contains potential contributions from the neighborhood boxes, and the far region includes contributions from the interaction list boxes. The electric field is calculated as the gradient of the potential $\mathbf{E} = -\boldsymbol{\nabla}V(\mathbf{x})$, and thus the far field local expansions can be evaluated at each target point after the translation of all local expansions to the highest level of the D-tree. This is called the local-to-point (L2P) evaluation and it is an elementary operation in DA. At the end of the downward pass, all the contributions to each particle from the multipole expansion of the far region is calculated. Then, the field at each target is modified by adding the point-to-point (P2P) field evaluations due to the sources in the neighborhood of the target (or the regular pairwise field of the near region).

In PHAD, the FMM is employed specifically in the computation of the far forces, as described above (hence the P2P operator is skipped). The far equation is exactly solvable as the spatial components (x_i, y_i, z_i) are constants, and hence the full solution is a kick. Indeed, the exact solution of the components of the far equation (10) and (12) are

$$\mathbf{x}_i^{[2]} = \mathbf{x}_i^0 \tag{16}$$

and

$$\hat{\mathbf{p}}_i^{[2]} = \hat{\mathbf{p}}_i^0 + t\frac{q^2 n_i}{4\pi\epsilon_0 mc^2}\sum_{j\in N_i^c}\frac{n_j((x_i-x_j)\mathbf{i}+(y_i-y_j)\mathbf{j}+\gamma^2(z_i-z_j)\mathbf{k})}{\gamma[(x_i-x_j)^2+(y_i-y_j)^2+\gamma^2(z_i-z_j)^2]^{3/2}}, \tag{17}$$

respectively. Thus, the only hurdle in solving the far equation is computing the expansion of the electric field vector

$$\tilde{\mathbf{E}}_{N_i^c} = \frac{q^2 n_i}{4\pi\epsilon_0 mc^2} \sum_{j\in N_i^c} \frac{n_j((x_i^0 - x_j^0)\mathbf{i} + (y_i^0 - y_j^0)\mathbf{j} + \gamma^2(z_i^0 - z_j^0)\mathbf{k})}{\gamma[(x_i^0 - x_j^0)^2 + (y_i^0 - y_j^0)^2 + \gamma^2(z_i^0 - z_j^0)^2]^{3/2}} \tag{18}$$

for each particle i. The fast multipole method clears this hurdle with an effort on the order of $\mathcal{O}(N)$ flops [16].

IV. THE SIMÒ INTEGRATOR AND THE SHORT-RANGE INTERACTIONS

An N-body numerical integrator can approximately solve the components of the near equation appearing in (9) and (11). General N-body numerical integration methods face efficiency limitations due to close encounters as Coulomb forces change rapidly. Aiming to accurately resolve close encounters with high efficiency, we developed the Simò N-body numerical integrator. A detailed description of the Simò integrator was published in [24] and we include here a short summary of it. The framework of the Simò integrator consists of two main components. The first component is an adaptive, variable order integrator with dense output; the second component is a strategy for an optimal selection of the particle-by-particle order and time step size.

For the first component we have developed a Picard iteration-based integrator [40] which is based on the Taylor method and uses Picard iterations to generate Taylor polynomials of the solution. The Taylor method straightforwardly provides a variable order, adaptive scheme giving high precision solutions. In addition, this integrator is implemented using the DA techniques in COSY INFINITY. In this context, evaluating a function in DA gives its truncated Taylor series at once, and thus our Picard iteration-based integrator is readily adaptable. Denoting the nth Taylor polynomial of a function $\mathbf{z}(t)$ in t centered at zero by $\mathcal{T}^n_{t,0}[\mathbf{z}(t)]$, the following Theorem 2 provides the theoretical basis of the integrator [40].

Theorem 2: For the initial value problem $\mathbf{y}'(t) = \mathbf{h}(\mathbf{y}, t)$ with $\mathbf{y}_0 = \mathbf{y}(0)$, let $\mathbf{z} = [\mathbf{y}^{\mathrm{T}}\ t]^{\mathrm{T}}$ with $\mathbf{z}_0 = \mathbf{z}(0)$ and $\mathbf{z}'(t) = \mathbf{f}(\mathbf{z})$. Suppose $\mathbf{z}$ has a Taylor series centered at 0 with nonzero radius of convergence ρ and $t < \rho$, then

$$\mathcal{T}^n_{t,0}[\mathbf{z}(t)] = \mathbf{z}_0 + \int_0^t \mathcal{T}^{n-1}_{s,0}[\mathbf{f}(\mathcal{T}^{n-1}_{s,0}[\mathbf{z}(s)])]ds. \tag{19}$$

Another main advantage of this integrator is that it gives a high-order dense output directly. Since the solution is expressed as a Taylor polynomial up to any order, the polynomial can be evaluated at any time and the trajectory of the particle can be fully traced within the interval $[0, \rho]$, and hence the dense output. Indeed, the order n may be arbitrarily selected and it may be evaluated for any t in the interval $[0, \rho]$ unlike other methods that allow evaluations at only discrete points of time. These features are essential for the efficiency of the N-body integrator where the N-body dynamics requires varying the time step frequently, especially when close encounters are present.

For the second component, we employed a unique strategy proposed by Simò [22] to determine each particle's optimal order p and optimal time step h_s. Simò's theorem, Theorem 3 [22], imposes two requirements for the optimal selection of p and h_s in order to achieve a given accuracy: minimizing the computational cost and a tolerance of the truncation error [22,41].

Theorem 3: Suppose that the Taylor expansion of the function $\mathbf{y}(t)$ has the radius of convergence $\rho(t)$ around some given initial conditions, and that the Taylor coefficients y_k satisfy $A_1 < \rho^k|y_k| < A_2$ for some fixed real numbers A_1 and A_2 with $0 < A_1 < A_2$. Then, a given relative error ε_r can be achieved at each step when the error tends to zero by the optimal time step size

$$h_s = \frac{\rho}{\exp(2)}. \tag{20}$$

The corresponding optimal order guarantees the required relative error bound ε_r and is given by $p = -\frac{1}{2}\ln\varepsilon_r$. As described in [24], we determine the optimal order using a different approach that is based on the estimation of the remainder of the Taylor series within its convergence interval. Once the estimated remainder term at the optimal time step h_s satisfies the user's required accuracy at a given order, that order is set to the optimal order p. Moreover, the optimal selection of h_s and p is performed for each particle at each time step. Additional characteristics of the Simò integrator are detailed in [24] and a short overview is given in Appendix B.

In PHAD, the Simò integrator is employed in the computation of the near dynamics, as described above. The near equation, contrary to the far equation, is not exactly solvable. Indeed, the Simò integrator is used to numerically approximate (up to machine precision) the solutions of (9) and (11). It will capture all close encounters, without skipping any collisions or degrading the accuracy of the solutions. Here, we also capture the effects of external fields.

V. IMPLEMENTATION AND PARALLELIZATION OF THE ALGORITHM

Our PHAD algorithm is illustrated in Fig. 1 and it works through the compositions described by (8).

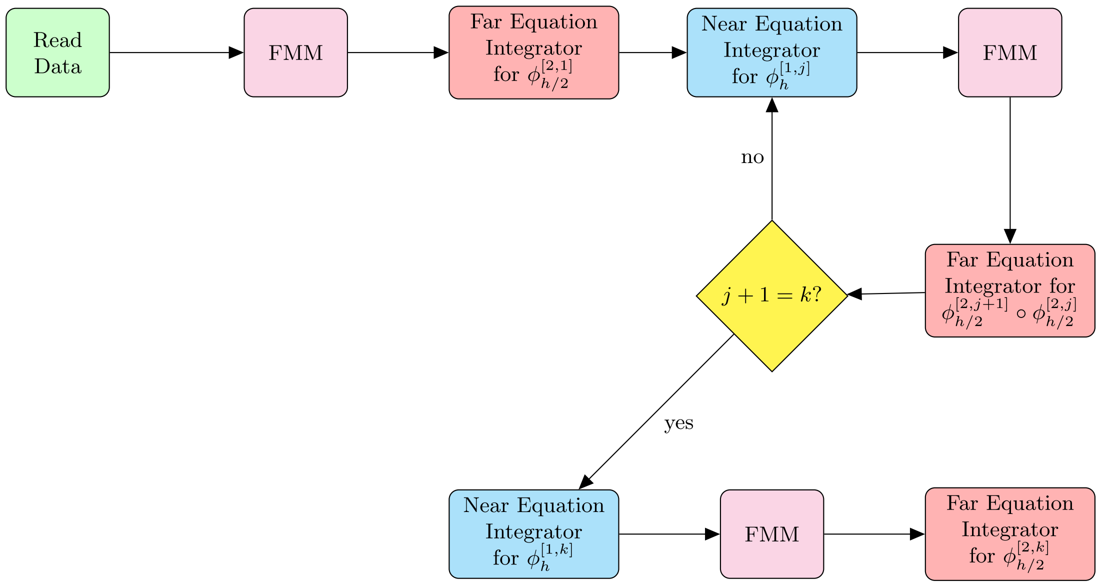

Read Data
FMM
Far Equation Integrator for $\phi_{h/2}^{[2,1]}$
Near Equation Integrator for $\phi_h^{[1,j]}$
FMM
no
Far Equation Integrator for $\phi_{h/2}^{[2,j+1]} \circ \phi_{h/2}^{[2,j]}$
$j+1=k$?
yes
Near Equation Integrator for $\phi_h^{[1,k]}$
FMM
Far Equation Integrator for $\phi_{h/2}^{[2,k]}$

FIG. 1. A flow chart of the PHAD algorithm.

The first exact solution to the far equation, $\phi_{h/2}^{[2,1]}(\mathbf{Y}_0)$, is approximated according to (16) and (17) using the first FMM evaluation. In order to capture all collisional effects, the time step of PHAD has to be chosen such that it is smaller than any timescale relevant to the specific application. Next, the first numerical solution to the near equation, $\phi_h^{[1,1]} \circ \phi_{h/2}^{[2,1]}(\mathbf{Y}_0)$, is computed using the Simò integrator. While doing this, the Simò integrator may on a particle-by-particle basis take several appropriate subtimesteps of h (an example is illustrated in Fig. 2) and utilize different orders.

Afterwards, each composition $\phi_h^{[1,j]}$ is computed using the Simò integrator, and each pair of compositions

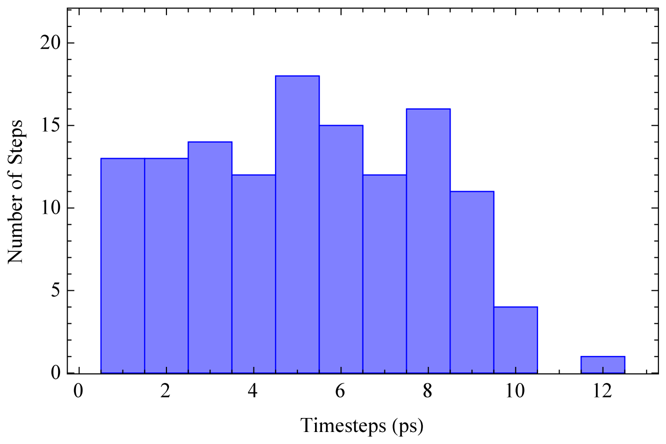


FIG. 2. An example showing the distribution and frequency of the time steps of a particle taken by the Simò integrator as subtimesteps of a PHAD time step $h = 1.7$ ns.

$\phi_{h/2}^{[2,j+1]} \circ \phi_{h/2}^{[2,j]}$ may be computed using a single FMM evaluation, as long as γ changes little between time steps. The reason this is possible is that the particle positions remain fixed through these compositions of solutions to the far equation. However, the FMM-based splitting may be different for $\phi_{h/2}^{[2,j]}$ and $\phi_{h/2}^{[2,j+1]}$. This observation sometimes makes a certain reconciliation process necessary.

A. Splitting reconciliation

Reconciling the current splitting with the previous splitting boils down to computing the total field contribution of the particles outside the neighborhood containing particle i in the previous FMM using the current FMM partitioning. This is accomplished by adding the field contribution of particles which were outside the neighborhood containing particle i but now are inside the neighborhood containing particle i and subtracting the field contribution of particles which were inside the neighborhood containing particle i but are now outside the neighborhood containing particle i to the potential of particles now outside the neighborhood. The process may be seen visually in Fig. 3.

This process of reconciliation is more efficient than naively comparing all particle indices outside the neighborhoods containing particle i at the previous and current time step. All that is needed are the indices of the particles in the same neighborhood as particle i given by the previous and current FMM partitioning. Denote the sets of indices of the neighborhood containing particle i at time steps j and $j+1$ by N_i and N_i^*, respectively. At time step $j+1$, the FMM computes an expansion of the field

$$\tilde{\mathbf{E}}_{N_i^{*c}}(x,y,z) = \frac{q^2 n_i}{4\pi\epsilon_0 mc^2} \sum_{k\in N_i^{*c}} \frac{n_j[(x-x_k^0)\mathbf{i} + (y-y_k^0)\mathbf{j} + \gamma^2(z-z_k^0)\mathbf{k}]}{\gamma[(x-x_k^0)^2 + (y-y_k^0)^2 + \gamma^2(z-z_k^0)^2]^{3/2}},$$

which is valid for locations within the same box as particle i. We use that N_i^c is the disjoint union of $N_i^{*c}\backslash(N_i\backslash N_i^*)$ and $N_i^*\backslash N_i$. Since $N_i^{*c}\cap(N_i\backslash N_i^*) = N_i\backslash N_i^*$ also, we may compute the fields from the previous splitting via

$$\tilde{\mathbf{E}}_{N_i^c}(x,y,z) = \tilde{\mathbf{E}}_{N_i^{*c}}(x,y,z) - \tilde{\mathbf{E}}_{(N_i\backslash N_i^*)}(x,y,z) + \tilde{\mathbf{E}}_{(N_i^*\backslash N_i)}(x,y,z), \tag{21}$$

where the fields $\tilde{\mathbf{E}}_{(N_i\backslash N_i^*)}$ and $\tilde{\mathbf{E}}_{(N_i\backslash N_i^*)}$ are computed directly. Figure 3 also shows the reconciliation process with sets labeled.

The current neighborhood N_i^* is constructed using sets output from the FMM. Specifically, the ordered set N_i and unordered set $N_i^*\backslash N_i$ are combined by first applying a quicksort [42,43] to $N_i^*\backslash N_i$ and then merging the two ordered sets using a single pass. Once the far equations are solved, N_i^* is stored as N_i. After the first time step, the ordered sets $N_i\backslash N_i^*$ and $N_i^*\backslash N_i$ are constructed by comparing N_i and N_i^* element by element.

B. Parallelization of the Simò integrator

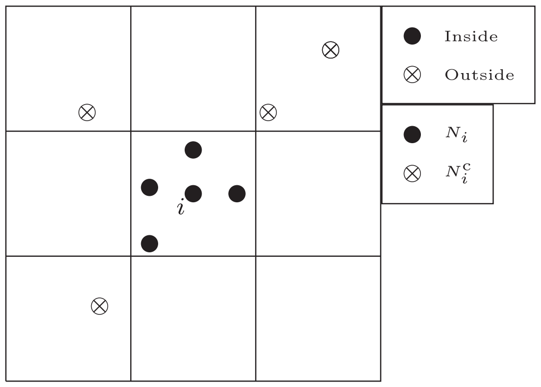
Since the computational time of the Simò integrator is quadratic with the total number of particles N, its performance decreases significantly for large N. Thus, the parallel implementation of the Simò integrator code makes more practical simulations with relatively large N. We implemented the Simò integrator in COSY INFINITY, which integrates a set of MPI commands with the COSYScript and easily constructs parallel loops. The parallel performance depends on the communication, synchronization, and the data/tasks distribution between the processors.

In the Simò integrator, the nested loops used for the computation of Coulomb forces in (11) requires the largest number of operations $N(N-1)\approx N^2$, and hence their parallelization is the main key to enhance the efficiency of the Simò integrator code. Due to the use of time bins, the first loop spans only a fraction of particles at each step (except for the first step) as explained in Appendix B. Therefore, parallelizing this loop is not so efficient because the communication overheads will quickly take over as the number of processors P increases, in addition to the second loop taking a long time to span all $N-1$ particles. On the other hand, it is more efficient to parallelize the second loop and distribute $N-1$ computations over P processors with effective communications.

To illustrate, we compared the computational times of the parallel Simò integrator simulation of 10^4 particles when parallelizing only the first loop and when parallelizing only the second loop used for the force computation. The results are shown in Fig. 4 where it is clearly more efficient to parallelize the second loop. Some other loops over N were also parallelized including the evaluation loop where the positions and momenta of the particles are updated in parallel.

C. Parallelization of PHAD

In this section, we compare three different parallelization strategies, which we refer to as the unbalanced, balanced, and parallel Simò strategies. Their main differences are in how they compute solutions to the near equations for neighborhoods $N_1, N_2, \ldots, N_k$. For the unbalanced strategy using P processors, each process j solves the near equations for neighborhoods $N_j, N_{j+P}, N_{j+2P}, \ldots$ using a serial Simò integrator. The balanced strategy is similar to the unbalanced strategy except that each process is assigned



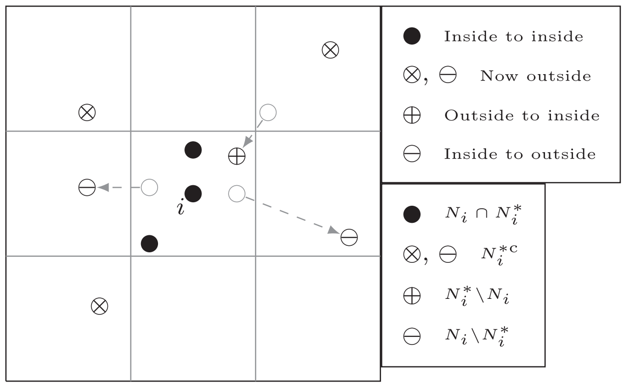


FIG. 3. Field contributions of outside particles (left) are now due to field contributions of particles now outside minus inside to outside particles plus outside to inside particles (right). Particles that were in N_i are now in N_i^{*c} minus $N_i\backslash N_i^*$, union $N_i^*\backslash N_i$.

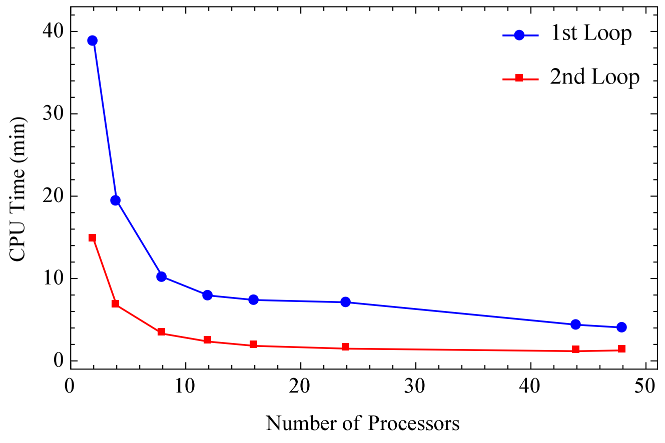


FIG. 4. Comparison of the computational times of the parallel Simò code when only one of the loops used for force computations is parallelized at a time, showing that it is more efficient to parallelize the second loop.

Unbalanced

Process 1 $n_{11} \to n_{12} \to n_{13} \to n_{31} \to n_{32} \to n_{33}$

Process 2 $n_{21} \to n_{22} \to n_{23} \to n_{24} \to n_{41} \to n_{42} \to n_{43} \to n_{44}$

Balanced

Process 1 $n_{11} \to n_{12} \to n_{13} \to n_{21} \to n_{22} \to n_{23} \to n_{24}$

Process 2 $n_{31} \to n_{32} \to n_{33} \to n_{41} \to n_{42} \to n_{43} \to n_{44}$

Parallel Simò

Process 1 $n_{11} \to n_{13}$　$n_{21} \to n_{23}$　$n_{31} \to n_{33}$　$n_{41} \to n_{43}$

Process 2 n_{12}　$n_{22} \to n_{24}$　n_{32}　$n_{42} \to n_{44}$

FIG. 5. Diagram showing which process computes the near integration for particle indexed n_{ij} for neighborhood N_i. The absence of arrows between indices indicates communication among processes.

neighborhoods which have a similar number of particles. Specifically, each process j solves the near equations for neighborhoods $N_{\sigma(j)}, N_{\sigma(j+P)}, N_{\sigma(j+2P)}, \ldots$, where σ is a permutation of indices 1 through k such that $|N_{\sigma(1)}| \leq |N_{\sigma(2)}| \leq \cdots \leq |N_{\sigma(k)}|$. In the parallel Simò strategy, each neighborhood's near equations are solved sequentially using a parallel Simò integrator which updates particle positions and momentum in parallel. Figure 5 gives which of two processes would compute solutions to the near equations for particles with indices n_{ij} in neighborhood N_i where N_1 and N_3 contain three particles and N_3 and N_4 contain four particles.

For all the distributions we have examined, there is a little difference in the performance between the unbalanced and balanced strategies as can be seen in Fig. 6. On the other hand, the parallel Simò integrator strategy seems to have the lowest performance. However, this may be due to the low maximum number of particles allowed in a neighborhood, the q value, fixed at sixty for this numerical experiment. For a larger sized neighborhood or a distribution in which there are significant differences in computational requirements between neighborhoods, the parallel Simò integrator may overcome its higher communication cost. Nonetheless, the memory required by our PHAD code increases quickly when increasing the q value, and hence we did not carry out the parallel Simò integrator strategy tests with large q because we are bounded by the available memory. Clearly, further optimization might be possible to improve parallel efficiency, subject to available hardware parameters.

The other main components of PHAD, the FMM and the far equation integrator, are parallelized in a similar manner to the unbalanced method. In the FMM, the local expansions for $N_j, N_{j+P}, N_{j+2P}, \ldots$ are computed by process j and the far equation integrator computes the exact solution of the far equations for these particles. The one difference between the parallelization strategies for these components of PHAD is that the indices of the neighborhoods are not communicated across all processes for the last time step in the unbalanced and balanced strategies since there is no subsequent time step that requires the previous neighborhoods. In the parallel Simò strategy, each process must have

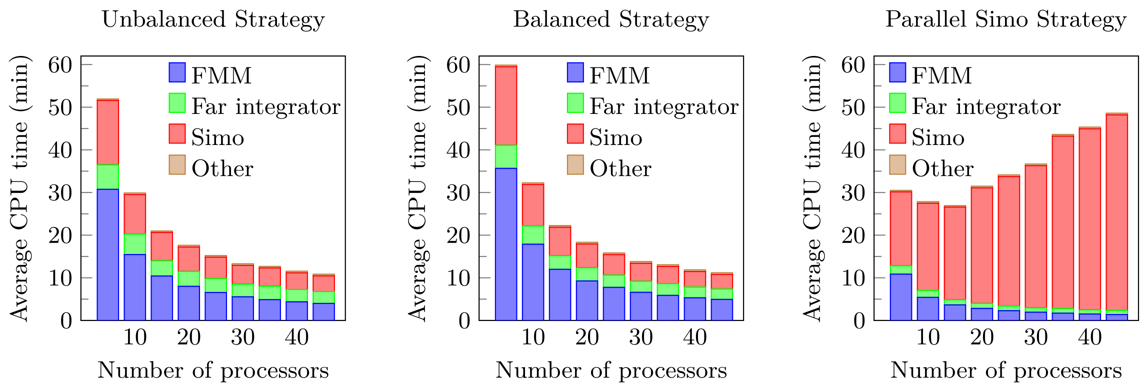


FIG. 6. Average CPU times of different subprocedures for PHAD parallelization strategies with 49.9×10^3 particles, FMM order 6, and clustering parameter $q = 60$. The unbalanced and balanced strategies show CPU times for five time steps and the parallel Simò strategy was ran for one time step.

the full collection of neighborhood index lists for the parallel Simò integrator.

Based on the computational time results of the three parallelization strategies, and to abide by the limited available memory, we chose to use the unbalanced parallelization strategy for our applications.

VI. APPLICATIONS

A. Dynamics of beamlets

Here, we study the dynamics of beamlets, i.e., several small beams that have parallel directions of motion and may be used in different applications. It has been proposed that beamlets may be employed as an alternative to the continuous transfer (CT) extraction mode between the Proton Synchotron to the Super Proton Synchotron [44]. Another application of beamlets is in a cancer treatment called intensity modulated radiation therapy (IMRT) [45]. In addition, the system of beamlets allows the study of Coulomb interactions within charge bunches, which is important to understanding the cold charged particle sources essential to producing high-intensity, high-brightness charged particle beams [46]. For instance, nanometer sharp tip emitters have been shown to produce ultracold electrons [47], and thus our studies on the effect of Coulomb interactions on electron beamlets provide important insights for the development of ultracold, high-brightness electron sources based on nanometer-scale sharp tip arrays [48,49].

We begin our applications with benchmarking PHAD accuracy and comparing its efficiency to the Simò integrator for one beamlets example. Then, we present other examples of PHAD simulations to beamlets dynamics.

1. Comparison of PHAD and the standalone Simò integrator simulations

The standalone Simò integrator is a direct N-body numerical integrator that gives an accurate simulation of the dynamics and has been previously benchmarked [24]. PHAD is proposed to give a comparable accuracy with significant efficiency improvements. The computational efficiency of both algorithms of PHAD and the standalone Simò integrator depend on different factors. The common factors between both algorithms are the number of particles and the number of processors. While the number of particles is set by the specific problem, the computational time dependence on N is quadratic for the Simò integrator and linear for PHAD (when the right PHAD parameters are selected). Increasing the number of processors used by both codes enhances their performances until the communication overheads take over and their performances start to decline.

The specific feature that influences the speed of the standalone Simò integrator is the number and type of time bins as explained in Appendix B. For PHAD, the clustering parameter q affects its performance where it plays a role

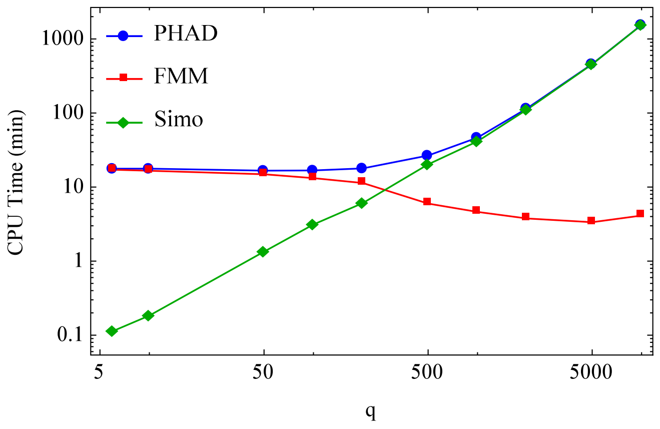


FIG. 7. Performance of PHAD as a function of the clustering parameter q for a Gaussian distribution of 10^5 electrons. The computational time of PHAD is primarily composed of the computational time of its FMM and the Simò integrator components.

both in space by the FMM subdivision, and in time stepping by the Simò integrator. Figure 7 shows how the computational times of the FMM, the Simò integrator, and PHAD depend on the clustering parameter q for a Gaussian distribution of 10^5 electrons. The computational time of the FMM decreases while that of the Simò integrator increases as the q value increases. Since the computational time of PHAD is mainly a combination of both the FMM and the Simò integrator computational times, the q value has to be chosen such that it minimizes the computational time of PHAD. For the example in Fig. 7, PHAD computational time increases and becomes almost equal to the computational time of the Simò integrator after $q = 200$. In general, selecting the right PHAD parameters allows to achieve a computational time that is linear with N while maintaining the accuracy requirements.

To benchmark PHAD accuracy against the stand alone Simò and demonstrate the efficiency difference, we compared the results and the computational times of a simulation of beamlets using both PHAD and the standalone Simò integrator. These simulations were performed using a 60-node CPU/GPU hybrid cluster called Gaea at Northern Illinois University. The 60 nodes are connected via Full 1 : 1 nonblocking InfiniBand and Ethernet switch connectors. Each node is an HP SL380s G7 that has 2× Intel X5650 2.66 GHz 6-core processors, 72 GB RAM, and 4 × 500 GB 2.5″ SATA disk drives (or 2 TB each node).

In this simulation, there are five beamlets symmetrically surrounding a centered beamlet. Each beamlet contains 10^4 electrons of a Gaussian spatial distribution, and all the beamlets are within a halo of 2×10^4 electrons of a uniform spatial distribution within a 1 mm radius. The initial temperature of the system is 0 K, and it is set to propagate along the z direction with an electric field $E_z = -20\ \mathrm{kV\,m^{-1}}$ applied for 2.5 cm. Figure 8(a) shows a cross section of the initial configuration of the beamlets system.

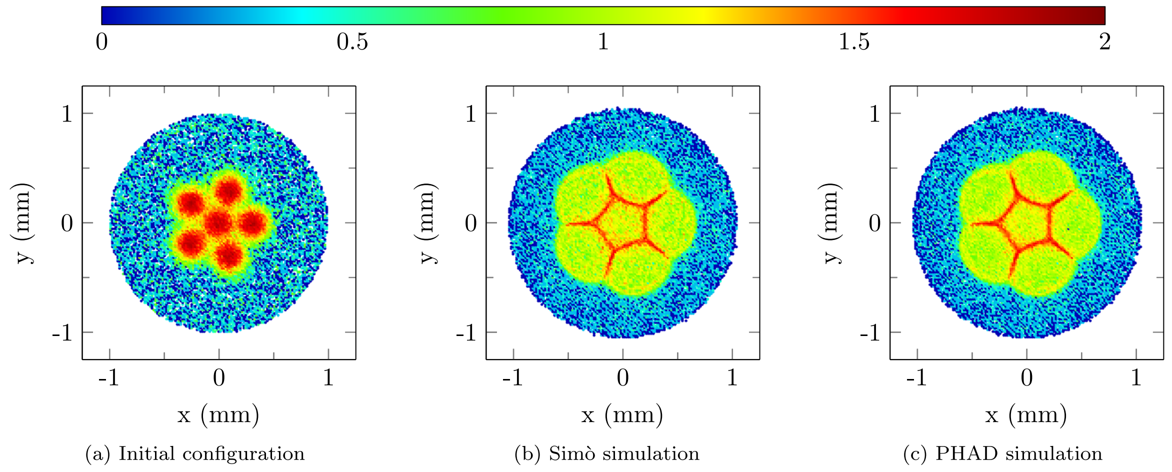


FIG. 8. Density log plots of the cross-section of six beamlets each consisting of 10^4 electrons Gaussian distributed, and surrounded by a halo of 2×10^4 uniformly distributed electrons: (a) initial condition, and the results after $t = 1.7$ ns of: (b) the standalone Simò integrator simulation and (c) PHAD simulation.

The parameters of the simulations follow: the accuracy of the Simò integrator was 10^{-12}, the maximum allowed order was 20, and the number of equal-width time bins was 40. PHAD time step was 3.3 ps which is smaller than the plasma period of ∼1 ns estimated from an average density of the beamlets. After a simulation time of 1.7 ns, the cross sections of the simulated beamlets by the standalone Simò integrator and PHAD are qualitatively nearly identical as shown in Fig. 8(b) and Fig. 8(c), respectively. For both cross sections, the number of electrons per bin was calculated from a nonparametric estimation of the probability density function (PDF) using the same number of bins and bandwidth and is depicted in Fig. 9 along with the density difference between both simulations (the noninteger number of electrons is due to the estimation from the PDF). With respect to the standalone Simò integrator, Fig. 9 (right plot) indicates that the resulted density from PHAD simulation did not change in about 88.6% of the bins, about 10.5% of the bins changed by one electron, and less than 1% of the bins changed by two or three electrons.

Moreover, we compared the coordinates of each electron from the simulations of both codes. The absolute errors for the x and y coordinates are shown in Fig. 10 where the majority of these errors are on the order of 10^{-6}. While both runs were performed using 48 processors, the computational time of PHAD was ∼21 hours while the computational time of the standalone Simò integrator was ∼50 days. This indicates that the efficiency gained from using PHAD is about an order of magnitude compared to the Simò integrator. Hence, PHAD is as accurate as the direct methods, but is significantly more efficient. Further studies of PHAD accuracy when the near equations were solved by the Picard iteration-based integrator were performed in [37] and showed good agreement between PHAD and the N-body Picard iteration-based integrator, which are still valid. Accordingly, we can employ PHAD to model more complicated applications such as the electron cooling of ion beams to which we give some examples in Section VI B.

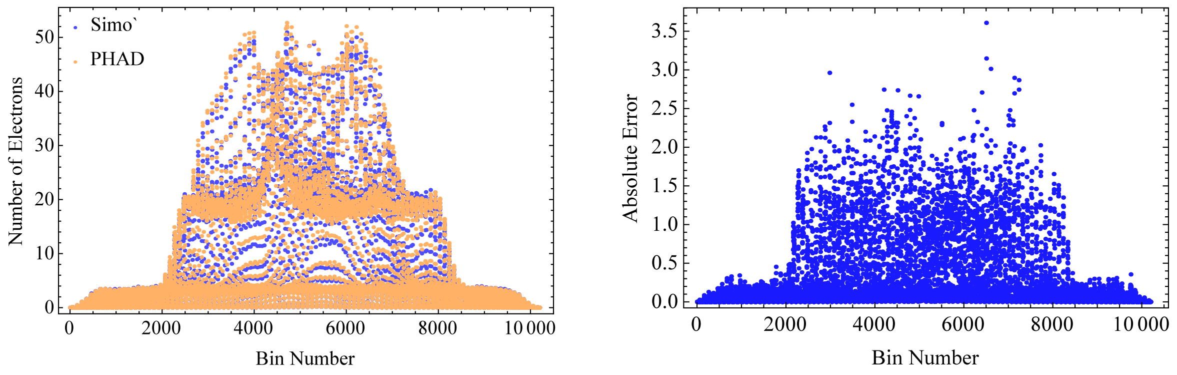


FIG. 9. The resulted transverse density of six electron beamlets within a halo of electrons simulated by the standalone Simò integrator and PHAD (left). The difference between these resulted densities (right) shows that the accuracy of PHAD is comparable to that of the standalone Simò integrator.

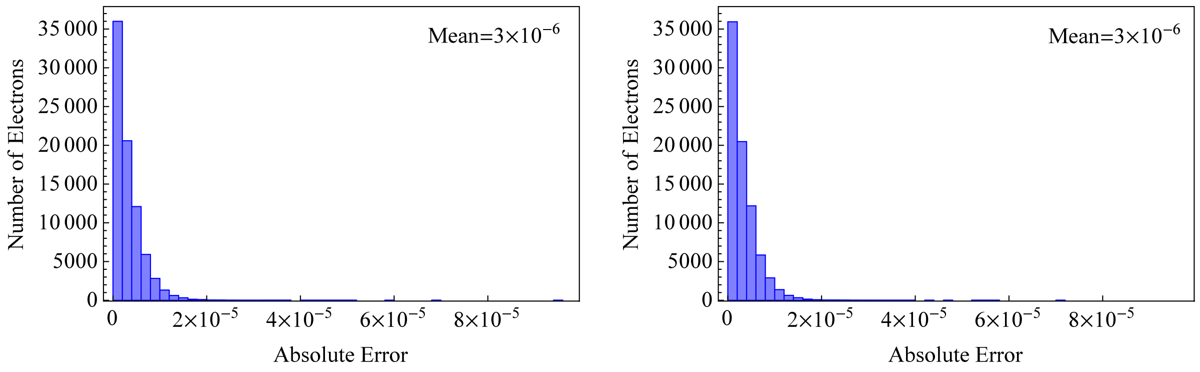


FIG. 10. The absolute errors in the *x* coordinates (left) and in the *y* coordinates (right) of the electrons in the beamlets simulations by PHAD compared to the standalone Simò integrator. The mean of the absolute errors is on the order of 10^{-6}.

2. *PHAD simulations of beamlets dynamics*

We simulated electron beamlets with a halo of electrons as depicted in Fig. 11. Initially, nine beamlets each consisting of 10^4 electrons were surrounded by 2×10^4 halo electrons uniformly distributed within a 1 mm radius for an overall charge of 17.6 fC. Electrons within the beamlets are Gaussian distributed in transverse directions with a standard deviation of about 0.08 mm. In the longitudinal direction, electrons are uniformly distributed within 5 μm of zero. The initial momenta were distributed according to the Maxwell-Boltzmann distributions at different initial temperatures. The electrons were accelerated in the longitudinal direction by a -20 kV m^{-1} electric field for 2.5 cm.

For these runs, the simulation time was about 2 ns performed by 120 time steps. PHAD time step was 16.7 ps which is smaller than the plasma period of 1.4 ns estimated from an average density of the beamlets. This time step is also smaller than the estimated collisional relaxation time for the beamlets at the different temperatures used in our simulations: ~4 ns for 20 K, ~19 ns for 100 K, and ~0.3 μs for 1000 K. Our simulations produce similar patterns in the charge densities of electrons as those detected experimentally in [46], where cooled Rubidium ion beamlets were measured. At lower initial temperatures, the particle densities evolve until there are eight well-defined spokes visible around a central hub as in Fig. 12(c). As the initial temperature is increased, these spokes become less and less defined as in Fig. 12(f). At higher temperatures, the spokes do not form as is the case for 1000 K shown in Fig. 12(i). We envision a future nondestructive ultracold electron beam emittance characterization method using these observations.

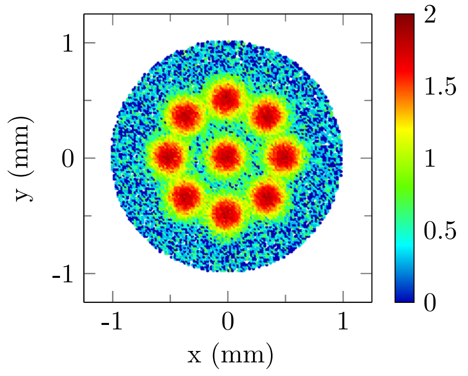


FIG. 11. Density log plot of the cross-section of the initial configuration of ten beamlets each consisting of 10^4 electrons Gaussian distributed, and surrounded by a halo of 2×10^4 uniformly distributed electrons.

B. Electron cooling of ion beams

Modern electron-ion colliders will rely on the high luminosity of heavy hadron beams, which will be accomplished via the application of various cooling methods depending on the intensity and temperature of the ion beam. For high-intensity beams, two methods of fast cooling are proposed [50]: the electron cooling (also known as the conventional electron cooling) is efficient for low temperature ion beams, and the strong electron cooling such as the coherent electron cooling (CeC) is suggested to efficiently cool high energy ion beams. Both techniques highly depend on Coulomb interactions between the ions and the cooling electrons, and it is difficult to estimate the cooling rates analytically. Hence, numerical simulations are crucial for predicting cooling rates and for the development of modern electron-ion colliders. In the following sections, we present our simulations related to both cooling methods performed by PHAD which we proposed to tackle such complicated nonlinear multiparticle large-scale beam dynamics problems. We note that these are the first electron cooling simulations based on first principles.

1. *Density modulations for the coherent electron cooling*

The coherent electron cooling (CeC) is a novel method proposed to cool high-energy intensive hadron beams and achieve high luminosity on a much faster rate than other

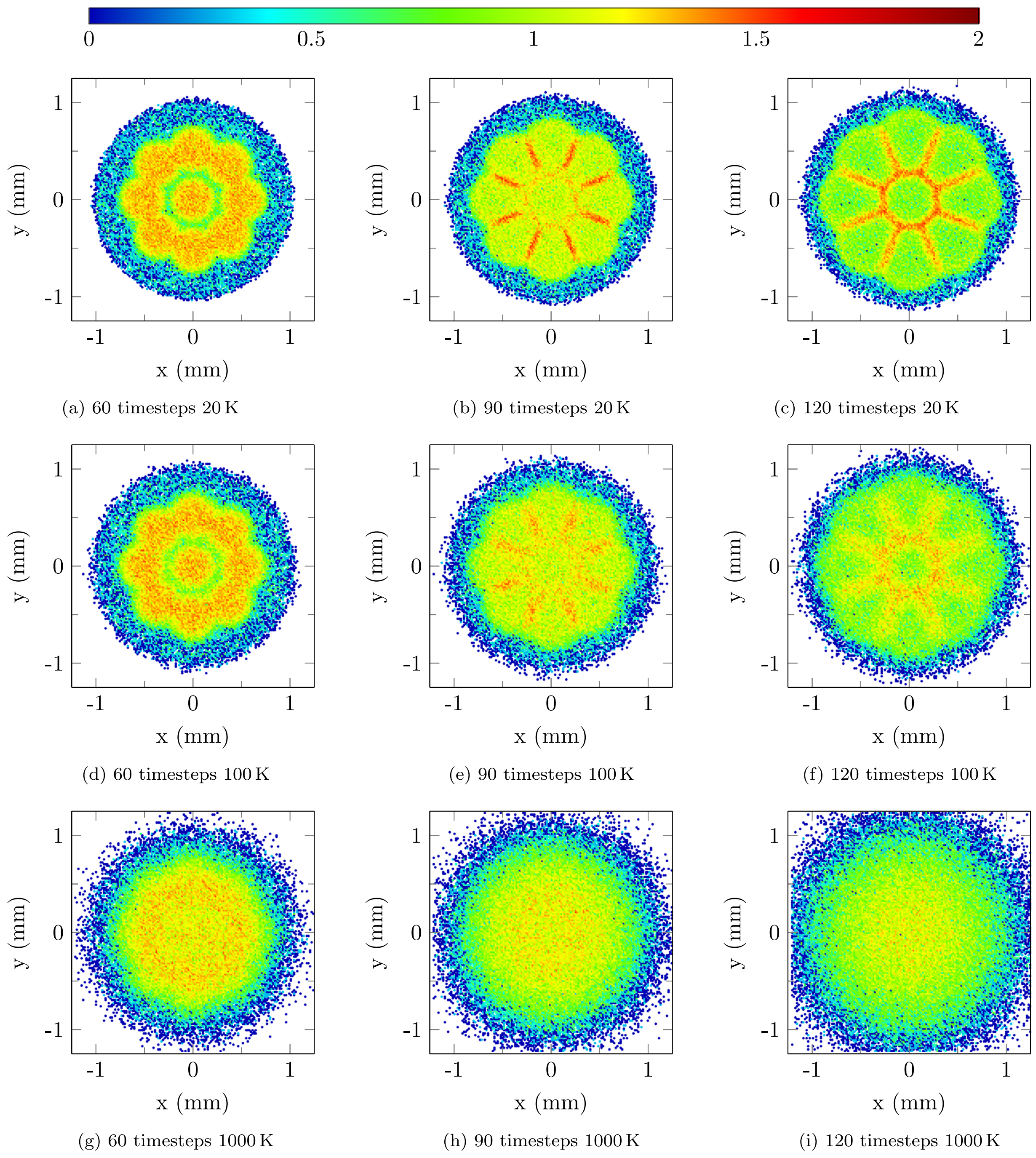


FIG. 12. Density log plots of the evolution of the electron beamlets with different initial temperatures: (a–c) 20 K, (d–f) 100 K, and (g–i) 1000 K. The simulations were performed by PHAD and different number of time steps are presented.

cooling methods [51,52]. In general, the CeC technique depends on the Debye shielding (screening) process that yields perturbations in the electron beam's density and velocity [53]. A typical CeC scheme incorporates three sections: a modulator, an amplifier, and a kicker.

In this section, we present our PHAD simulation of the electron density modulation in the modulator section relevant to the proof-of-principle (PoP) CeC experiment in the Relativistic Heavy Ion Collider (RHIC) at Brookhaven National Laboratory (BNL). In the modulator section, the ion beam and the electron beam are merged and copropagated through a focusing field generated by a set of four quadrupoles. The Coulomb interactions between electrons and ions results in a density modulation of the electron beam as each ion attracts the electrons around itself (the process of shielding). The ion beam consists of 40 GeV/nucleon Au^{+79} ions, and the velocity of the electrons is matched to that of the ions. The parameters of the modulator lattice and the parameters of the ion and electron beams of the PoP CeC experiments at RHIC can be found in [54].

We consider a Gaussian electron beam and a single ion in a hard-edge quadrupole focusing field. Since the discreteness of the particles in particle simulations produces a shot noise that is strong compared to the modulation signal, we extract the modulation signal from the shot noise in the same manner used in [54]. The propagation of an electron beam through the modulator lattice is simulated two times: one with the presence of the ion and the other is without the ion. Subtracting the final results of these two simulations, we obtain the density modulation of the electron distribution due to the ion.

In the PoP CeC experiments at RHIC, the size of the electron beam is much larger than the lengths relevant to the shielding of the ion. Since the far away electrons do not contribute to the density modulations, we simulate a longitudinal slice of the electron beam with smaller momentum spread and a moderately smaller transverse size. The ion is positioned at the center of the Gaussian electron beam that has the RMS sizes: $\sigma_z = 1.1\ \mu$m and $\sigma_{x,y} = 0.11$ mm. The propagation through the modulator section was performed by 600 PHAD time steps of 16.7 ps, which is much smaller than the collisional relaxation time $\sim 4\ \mu$s and the electrons plasma period ~ 2 ns (both estimated from an average density).

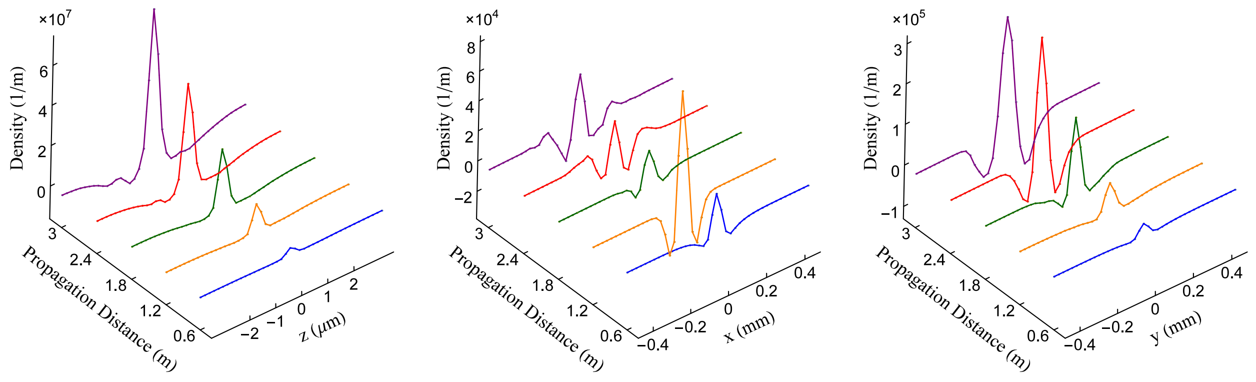
The density modulations resulting from the average of three simulations is presented in Fig. 13, where the evolution of the density modulations throughout the modulator section is shown at different propagation distances. Around the location of the ion, the longitudinal density modulation gradually increases as the propagation distance increases while the transverse modulations vary in a way that reflects the effect of the quadrupoles on the transverse beam size. Our simulation gives density modulations similar to (and supports) the results presented in [54]. Using a single ion is the initial step to quantify the modulation signal for various energies and locations of the ion. Then, the density modulations due to the ion beam can be examined and used to predict the cooling time of the CeC system.

2. Conventional electron cooling of ion beams

Electron cooling is a method proposed by G. I. Budker in 1967 to significantly reduce the ion beam emittance (phase space volume) [55]. This technique was experimentally demonstrated in 1974 at the first cooler ring NAP-M [56]. In this method, a hot ion beam is overlapped by a cold electron beam and both beams are propagated at the same average velocity through a localized small section of an accelerator or a storage ring. Through this brief interaction, the momentum is transferred from individual ions to the surrounding electrons and thus the ions experience a dynamical friction or a velocity drag which accumulates turn by turn causing a reduction of the 6D phase space volume of the ion beam (i.e., cooling the ion beam).

Most of the models used to study the electron cooling process rely on the standard analytical formulas of the dynamical friction force which are based on many assumptions, and it is challenging to include external magnetic fields. Moreover, these models generally ignore rare close encounters although their collisions play an essential role in the cooling process. As some studies such as [57,58] show, an ion's velocity and the friction force change significantly due to those few close collisions, and a full description of the dynamics can be provided by high precision collisional numerical simulations that are based on a minimal set of assumptions and that consider the whole distribution. Below, we demonstrate PHAD's capability to predict cooling indicated by the reduction of emittances.

a.Electron cooling of longitudinal emittance.—We propagate a C^{+6} ion beam of 7 MeV/nucleon kinetic energy through a 3.4-m-long straight cooling section, which is part of a 161-m-long storage ring. The initial parameters of this



(a) Longitudinal modulation (b) Horizontal modulation (c) Vertical modulation

FIG. 13. PHAD simulations of density modulations of a Gaussian electron beam due to a centered ion copropagating through the modulator section of the PoP CeC at RHIC: (a) longitudinal direction, (b) horizontal direction, and (c) vertical direction. The density modulations are shown at propagation distances: 0.6 m (blue), 1.2 m (orange), 1.8 m (green), 2.4 m (red), and 3 m (purple).

TABLE I. Initial parameters of the simulations of electron cooling of the C^{+6} ion beam.

Parameter	Value
Energy (MeV/nucleon)	7
Ion	C^{+6}
Number of ions	500
Ions RMS bunch length (mm)	5.5
Ions RMS momentum spread	2.5×10^{-4}
Ions RMS emittance [h./v.] (mm mrad)	0.15/0.1
Length of cooler (m)	3.4
Beta value in cooling section (m)	10
Magnetic field in cooling section (T)	0.1
Number of electrons	15604
Electron charge (e)	−900
Electron beam radius (mm)	7.5
Electron beam temperature [trans/long] (eV)	$0.01/10^{-5}$
Electrons bunch length (mm)	5.5
Electron beam current (mA)	15
PHAD time step (ns)	0.93

simulation are included in Table I, where the spatial distribution of the ion beam is Gaussian and the electron beam's is uniform. PHAD was used to model the cooling section, while the ions were passed through the rest of the ring using transfer maps generated by COSY INFINITY. Using an average density of the ion beam, the 0.93 ns PHAD time step is smaller than all the relevant timescales in this simulation such as the electrons plasma period of ∼40 ns; the betatron oscillation periods are ∼1 μs; the estimated ions plasma period and collisional relaxation time are of the order of 10 μs; and the synchrotron oscillation period is a few seconds.

Before we present our cooling results, we first show that PHAD preserves symplecticity which is an important feature for electron cooling simulations. This is demonstrated through the invariant 4D/6D RMS emittances of the ion beam considered in this section through the ring and the cooling section without the electron beam. Indeed, Fig. 14 shows those invariants ϵ_{4D} and ϵ_{6D} in the absence of cooling. The 2D RMS emittance was not included in this example as presence of the longitudinal magnetic field of the solenoid introduces transverse coupling.

Now, we include the electron beam through the cooling section and observe the cooling process in the longitudinal direction. Each pass through the cooling section takes about 93 ns (or 100 time steps), and the electron bunch is refreshed for the next pass. The evolution of the longitudinal emittance after 0.7 ms is shown in Fig. 15, where the cooling is indicated by the decrease of the longitudinal emittance with time. This simulation suggests that the longitudinal cooling can be achieved with an electron bunch that is shorter than the ions beam bunch length. Also, the cooling is fairly fast which could be due to the relatively large current of the electron beam with respect to its bunch length. This is the first particle-based simulation

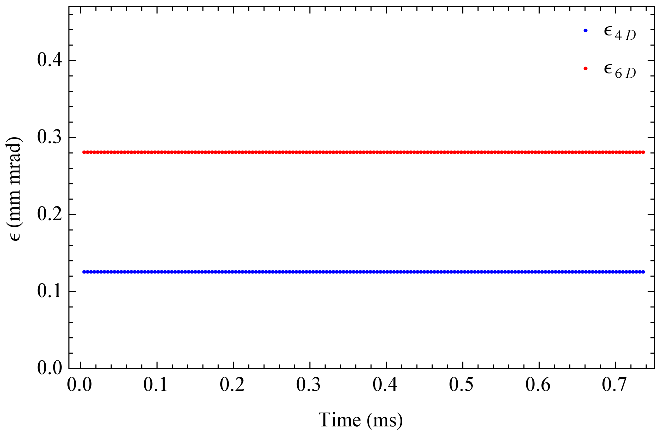


FIG. 14. A demonstration of symplecticity preservation by PHAD, showing the invariant RMS emittances ϵ_{4D}, and ϵ_{6D} of the C^{+6} ions beam in the absence of cooling. The emittances ϵ_{4D} and ϵ_{6D} are scaled as $\sqrt{\epsilon_{4D}}$ and $\sqrt[3]{\epsilon_{6D}}$, respectively.

which considers all collisions and predicts the longitudinal cooling of the ion beam without the need to calculate the friction force. Next, we provide more PHAD simulations and comparison of cooling times of the transverse emittance for different initial configurations.

b. Electron cooling of transverse emittance.—We carried out simulations of electron cooling of high energy proton beams where we varied the initial proton-electron beam configurations aiming to identify the ones that give fast cooling times. This is performed by quantifying the reduction of the transverse emittance of the proton beam through the simulation. The following simulations were introduced in Chapter 6 of [37]. Since the cooling times at higher energies are very long compared to feasible simulation times, we resorted to a modified algorithm. Owing to the kernel-independent nature of our FMM algorithm, we introduced

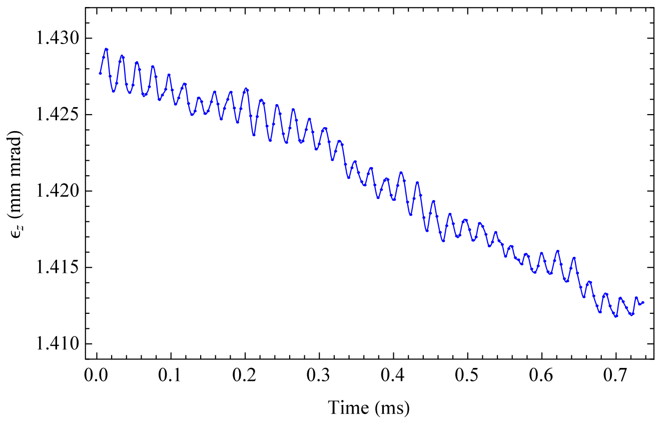


FIG. 15. The longitudinal emittance of the C^{+6} ions beam cooled with an electron beam of shorter bunch length and larger radius. The electron beam current is 15 mA and the longitudinal magnetic field in the cooling section is $B_z = 0.1$ T.

TABLE II. Initial parameters of the simulations of electron cooling of the proton beam.

Parameter	Value
Energy (MeV)	280
Ion	proton
Number of ions	100
Ions bunch length (mm)	1
Length of cooler (m)	3
Number of electrons	1000
Electron charge (e)	−32552.08
Electrons transverse momentum $p_{x,y}/p_z$	0
Electrons bunch length (mm)	1
Electron beam current (A)	1
PHAD time step (ps)	3.3

a softened Coulomb force to allow increased time steps without introducing nonphysical effects, at the expense of washing out close collisions. A softening parameter of $\lambda = 50\ \mu$m was employed (see Appendix C). Since under these circumstances there is no need for the sophisticated capabilities of the Simò integrator, the near equations were solved using a Picard-iteration based integrator of fixed order and time step as described in [40] for enhanced efficiency.

The initial parameters of these simulations are presented in Table II. In these simulations, PHAD time step was 3.3 ps which is smaller than the timescales relevant to these simulations such as the electrons plasma period that is <10 ns; the ions plasma period is ~0.3 ms; and the collisional relaxation time goes to infinity since the electron beam temperature is zero here. We varied the following

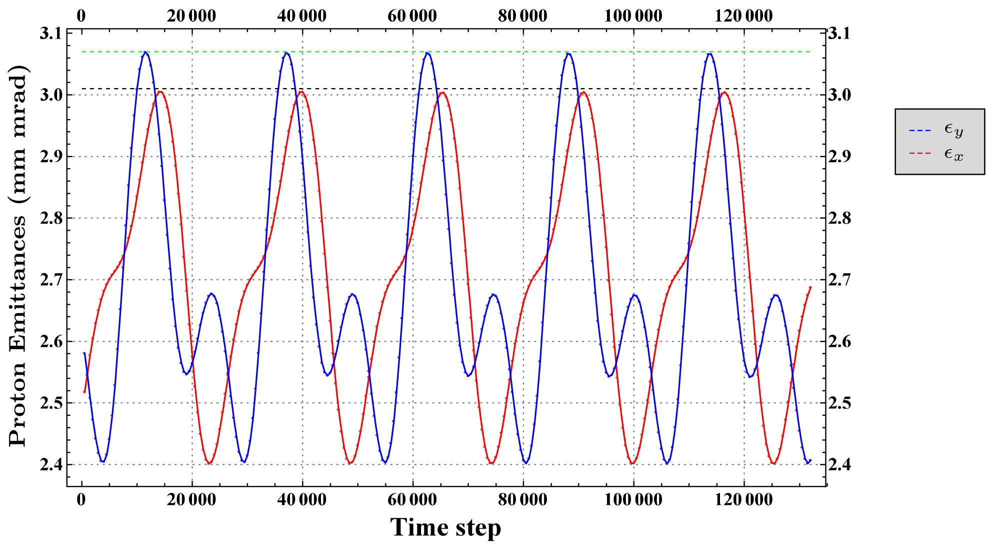


FIG. 16. The transverse emittance of the proton beam bunch of 1 cm radius cooled with an electron beam of the same radius, $B_z = 1$ T, and $p_{x,y}/p_z = 10^{-6}$. The black and green dashed lines are reference lines for ϵ_x and ϵ_y, respectively.

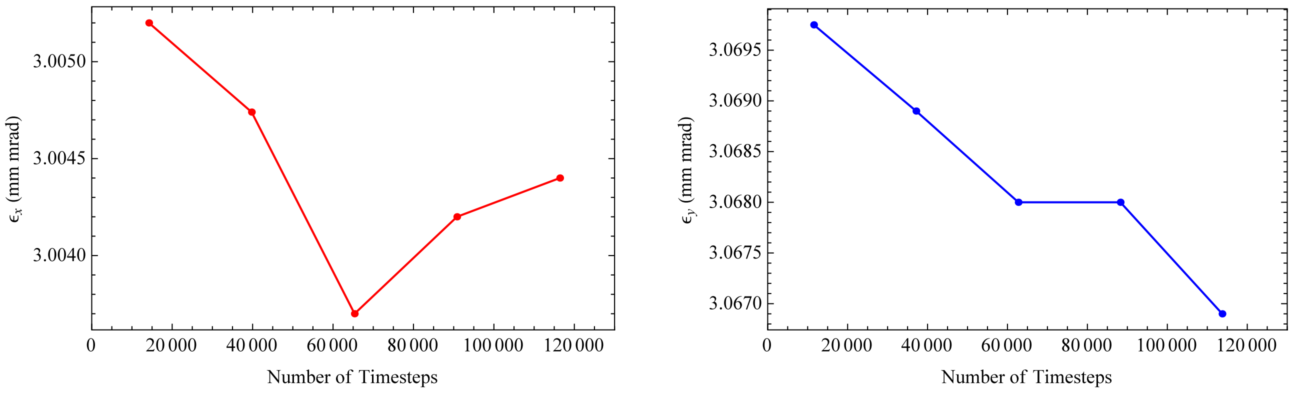


FIG. 17. Maxima of the horizontal (left) and vertical (right) emittances curves of the proton beam bunch of 1 cm radius cooled with an electron beam of the same radius, $B_z = 1$ T, and $p_{x,y}/p_z = 10^{-6}$. The decline of these maxima indicates cooling.

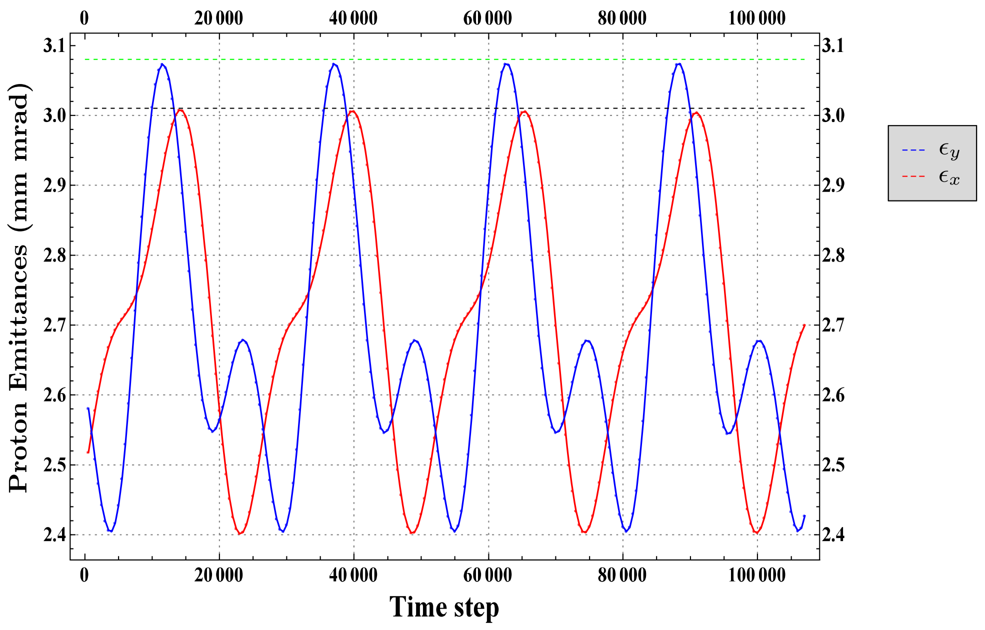


FIG. 18. The transverse emittance of the proton beam bunch of 1 cm radius cooled by a 5 mm electron beam radius, $B_z = 1$ T, and $p_{x,y}/p_z \approx 10^{-3}$. The black and green dashed lines are reference lines for ϵ_x and ϵ_y, respectively.

parameters: the radii of both the electron and proton beams, the transverse momentum of the proton beam, and the longitudinal magnetic field B_z. The electron bunches were refreshed with new cold electrons after each pass through the cooler which means after ∼16 ns (or 4700 time steps).

Beginning with a small transverse momentum of $p_{x,y}/p_z = 10^{-6}$ for the proton beam and a longitudinal magnetic field $B_z = 1$ T, both the electron and proton beams have a uniform distribution with the same bunch length and we vary their radii. In the case both the radii are set to 1 cm, the change of the transverse emittance after 0.44 μs (or 132,000 time steps) is shown in Fig. 16. The decrease of the horizontal (peaks of the red curve) and vertical (peaks of the blue curve) emittances is indicated by the horizontal dashed black and green lines, respectively. Figure 17 illustrates the decline of the maxima of each curve with time, which indicates cooling.

Then, we reduced the radius of the proton beam to 5 mm. The resulting transverse emittance in this simulation did not show a decline, indicating a longer cooling time compared to the equal radii of both beams. For the opposite case when the proton beam has a 1 cm radius that is larger than the 5 mm radius of the electron beam, we observed a reduction of the transverse emittance similar to that when both beams are of equal radii. We proceeded with this spatial configuration (proton beam of 1 cm radius and electron beam of 5 mm radius) in the following simulations.

We increased the transverse momentum of the proton beam to $p_{x,y}/p_z = 10^{-3}$ and performed the simulation for more than 100,000 time steps (about 0.36 μs). Figure 18 shows the change of the proton beam's horizontal (peaks of the red curve) and vertical (peaks of the blue curve) emittances with respect to the horizontal dashed black and green lines, respectively. This figure shows a slight decrease of the transverse emittance indicating a slow cooling compared to the previous case of the smaller transverse momentum.

Lastly, we fixed the transverse momentum to $p_{x,y}/p_z = 10^{-3}$, and we increased the longitudinal magnetic field to 1.5 T. During 3.4 μs (1,005,000 time steps), Fig. 19 shows the proton beam's horizontal (peaks of the red curve) and vertical (peaks of the blue curve) emittances with respect to the horizontal black and orange lines, respectively. From this figure, a clear reduction of the transverse emittance is

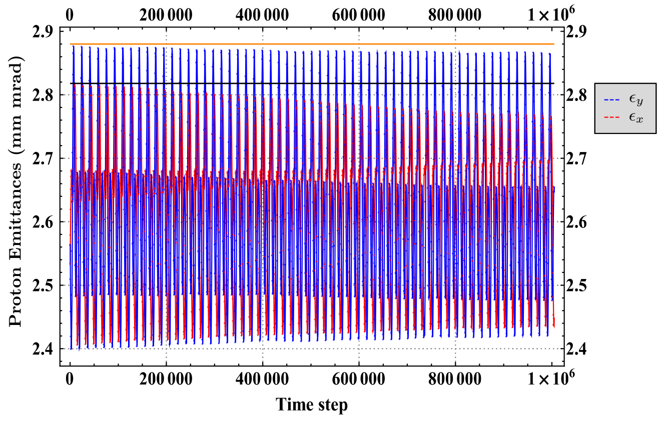


FIG. 19. The transverse emittance of the proton beam cooled bunch of 1 cm radius with a 5 mm electron beam radius in the presence of a strong longitudinal magnetic field of $B_z = 1.5$ T, and $p_{x,y}/p_z \approx 10^{-3}$. The black and orange lines are reference lines for ϵ_x and ϵ_y, respectively.

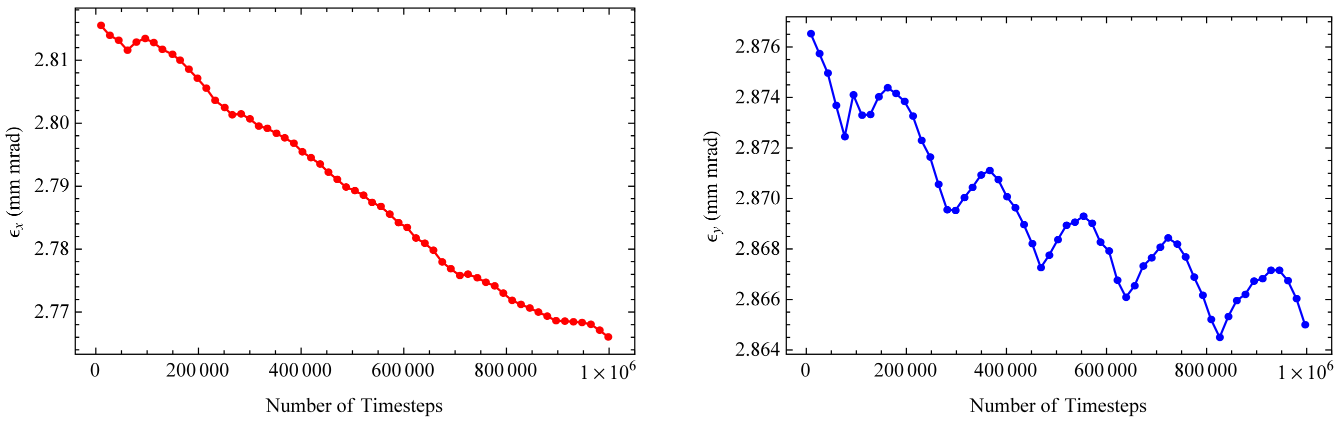


FIG. 20. Maxima of the horizontal (left) and vertical (right) emittances curves of the proton beam bunch of 1 cm radius cooled with a 5 mm electron beam radius, $B_z = 1.5$ T and $p_{x,y}/p_z \approx 10^{-3}$.

observed. The decline of the maxima of both emittances curves is also depicted in Fig. 20.

Based on the simulation results presented within this section, electron cooling of the transverse emittance is faster when the transverse size of the electron beam is smaller than or equal to that of the proton beam, and when the transverse momentum of the proton beam is small. These results also suggest that cooling time is faster for stronger longitudinal magnetic field as it is known for the magnetized cooling [59].

VII. SUMMARY AND CONCLUSIONS

Beam dynamics, especially in the nonlinear multiparticle regime, is a subject best treated numerically. Historically, this recognition led to the development of many algorithms and codes based on them; some of these codes even rise to the community code level. However, the vast majority of these codes are based upon variants of what generically may be called mean-field, collisionless codes. While this is sufficient for many applications, several recent beam physics applications emphasized the need for new, collisional algorithms and codes based on them. We briefly summarized two such applications here: ultracold electron beam generation based on arrays of sharp nanotips and electron cooling of ion beams.

We developed a new, efficient algorithm for high-fidelity collisional simulations. It is based on using the fast multipole method to compute forces among the beam particles pairwise, our new Simò integrator for time propagation, and Strang splitting to speed up the simulations while preserving symplecticity. Efficiency is guaranteed by linear scaling with particle number of the force computation; by provable computational effort minimization in the Simò time stepping for an *a priori* user-set error requirement; by the ability to separate near and far regions from the perspective of each particle and treat the regions independently from the computational point of view; by the ability to use large time steps due to the splitting technique; and by thorough parallelization of the code. High-fidelity is achieved by the tunable accuracy of the FMM with guaranteed error bounds, the adaptivity both in order and time of the Simò integrator, and maintenance of numerical (machine precision level) symplecticity.

The resulting algorithm, termed PHAD, was implemented and benchmarked. With our code, based on COSY INFINITY, we estimated roughly an order of magnitude decrease in time to solution compared to the previous best solution method, the Simò integrator. Therefore, PHAD now makes possible much faster simulations of larger scale collisional beam dynamics problems than before, while maintaining adequate accuracy. The algorithm's performance was illustrated with two applications where collisions play a significant role: generation of ultrabright beams and electron cooling of ions.

ACKNOWLEDGMENTS

This work was supported in part by the U.S. Department of Energy, Office of Nuclear Physics, under Contract No. DE-SC0008588. This work was also supported in part by the U.S. Department of Energy, Office of High Energy Physics, under Contract No. DE-SC0020241. This work used resources of the Center for Research Computing and Data at Northern Illinois University.

APPENDIX A: DIFFERENTIAL ALGEBRA

The performance of our numerical method relies on the concept of differential algebra (DA) and its implementation in COSY INFINITY. Thus, we give here a brief description of the DA method following its presentation in [23]. Classical numerical methods for differentiation that compute Taylor expansions of functions were considered impractical and inaccurate since their algorithms depend on the exact value of the functions at particular points. The DA methods are based on the fact that there are more information that can be extracted from a function other than its exact values. Defining an operator T that can extract Taylor coefficients of a function up to an arbitrary order, the function can be translated to an equivalence class consisting of all functions of the same Taylor expansion to the same order. Supplying a set of equivalence classes with well-defined basic arithmetic operations can turns it into a commutative algebra. When the operations of differentiation and integration are introduced, the resulting structure is called a differential algebra (DA).

Starting with the first nontrivial DA in $\mathbb{R}^2$, the real numbers a_0 and a_1 establish the set of all ordered pairs (a_0, a_1). The commutative algebra ${}_1D_1$ is formed by the ordered pairs along with the arithmetic operations of addition, scalar multiplication, and vector multiplication defined as

$$\begin{gathered}(a_0, a_1) + (b_0, b_1) = (a_0 + b_0, a_1 + b_1)\\ t \cdot (a_0, a_1) = (t \cdot a_0, t \cdot a_1)\\ (a_0, a_1) \cdot (b_0, b_1) = (a_0 \cdot b_0, a_0 \cdot b_1 + a_1 \cdot b_0),\end{gathered}$$

where t is a scalar. Defining the infinitesimal or the differential $d = (0, 1)$, any $(a_0, a_1) \in {}_1D_1$ can be written as

$$(a_0, a_1) = (a_0, 0) + (0, a_1) = a_0 + d \cdot a_1,$$

where the first and the second components are called the real and the differential parts, respectively. Also, any $(a_0, a_1) \in {}_1D_1$ satisfies the relation $(a_0, a_1)^n = (a_0^n, n \cdot a_0^{n-1} a_1)$ for all $n > 1$. A multiplicative inverse can be defined as $(a_0, a_1)^{-1} = (a_0^{-1}, -a_1/a_0^2)$ if and only if $a_0 \neq 0$. The derivation operator is defined as the map $\partial\colon {}_1D_1 \mapsto {}_1D_1$ by $\partial(a_0, a_1) = (0, a_1)$, and $({}_1D_1, \partial)$ is a differential algebra.

For the purpose of the automatic computation of derivatives, an operator [] is introduced as a map from the space of differentiable functions to ${}_1D_1$ such that $[f(x)] = (f(x), f'(x))$ for a function $f(x)$ with its value and derivative are given at a point x. The arithmetic operations for functions f and g can be written as follow

$$\begin{gathered}[f] + [g] = [f + g]\\ t.[f] = [t.f]\\ [f].[g] = [f.g].\end{gathered}$$

For a real x, it can be shown that $[f(x)] = f([x])$ where $[x] = (x, 1)$. Using the DA variable $(x, 1)$, we can compute the value and the derivative of $f(x)$ just through arithmetic operations. As an example, consider the following function and its derivative

$$f(x) = x^2 + \frac{1}{1+x}, \qquad f'(x) = 2x - \frac{1}{(1+x)^2}.$$

At $x = 2$, direct evaluations give $f(2) = 4.3$ and $f'(2) = 3.89$. However, we can evaluate both the function and its derivative using the DA representation of $x = 2 = (2, 1)$ and we get

$$\begin{aligned}f((2,1)) &= (2,1)^2 + \frac{1}{1 + (2,1)} = (4,4) + \frac{1}{(3,1)}\\ &= (4.3, 3.89).\end{aligned}$$

This treatment can be extended to any common intrinsic function g_i as $g_i([f]) = [g_i(f)]$ or $g_i[(a_0, a_1)] = [g(a_0), a_1 g_i'(a_0)]$. Therefore, any function can be represented by a finite number of intrinsic functions and operations in ${}_1D_1$. Following the description of $({}_1D_1, \partial)$, we can arrive to the differential algebra $({}_nD_v, \partial_1, \ldots, \partial_v)$ which allows the computation of derivatives of functions in v variables up to order n.

In the implementation of the DA techniques in COSY INFINITY software, evaluating a function in DA creates a DA vector that contains the truncated Taylor polynomial around a particular expansion point. These DA vectors are treated according to the various supported operations of DA such as arrays of DA vectors, arithmetic operations, derivation and anti-derivation, evaluation, inversion, and composition. Thus, implementing Taylor methods using DA in COSY INFINITY is very efficient and is very accurate due to the ability of computing high orders.

APPENDIX B: OVERVIEW OF THE SIMÒ INTEGRATOR

The main aspects of the Simò integrator were explained in [24] and some of them are summarized here. The optimal time step size h_{s_i} of particle i is obtained from (20) and it involves finding the radius of convergence ρ_i. For a function expanded into a Taylor polynomial, the distance from its expansion center to the nearest singularity in the complex plane is its radius of convergence. The components of the right-hand side of the near equation are expanded into Taylor polynomials in the Simò integrator, and converge to the solution in the interval $[0, \rho_i]$. If we define $\alpha_i(t) = f_i^2 + \hat{p}_{x_i}^2 + \hat{p}_{y_i}^2 + \hat{p}_{z_i}^2$, and $\beta_{i,j}(t) = (x_i - x_j)^2 + (y_i - y_j)^2 + \gamma^2 (z_i - z_j)^2$, then the components of the right-hand sides of (9) and (11) are singular when $\mathcal{T}^n_{(t,0)}[\alpha_i(t)] = 0$ or $\mathcal{T}^n_{(t,0)}[\beta_{i,j}(t)] = 0$. Hence, the closest root of $\mathcal{T}^n_{(t,0)}[\alpha_i(t)]$ or $\mathcal{T}^n_{(t,0)}[\beta_{i,j}(t)]$ to the center of the complex plane represents ρ_i for a particle i.

In order to reduce the number of floating point operations, the Simò integrator estimates the radius of convergence from the coefficients of $\mathcal{T}^n_{(t,0)}[\alpha_i(t)]$ and $\mathcal{T}^n_{(t,0)}[\beta_{i,j}(t)]$. We chose to implement Lagrange's lower bound to estimate ρ_i as suggested by the results of our numerical experiments of different theorems of lower bounds [24]. While these numerical experiments showed that estimating ρ_i at order 2 is accurate enough, some of our simulations of real applications required higher orders. Therefore, we have modified the order at which ρ_i (and h_{s_i})

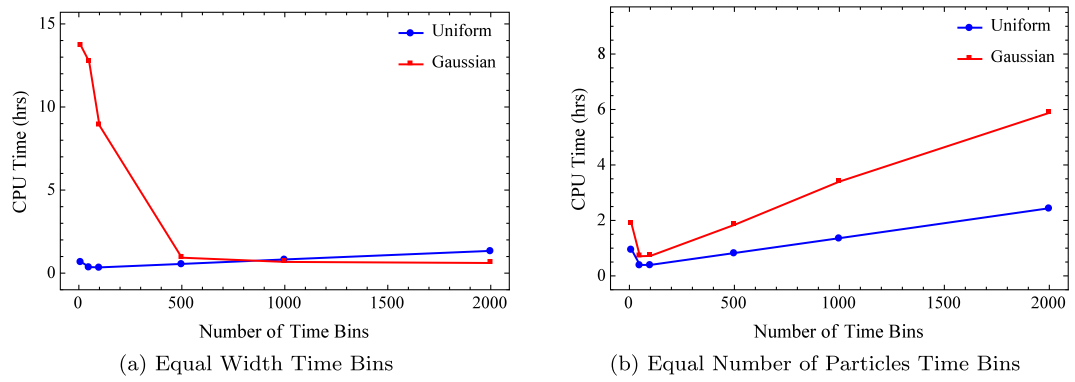


FIG. 21. The dependence of the computational time on the number of time bins for the two types: (a) equal width time bins, (b) equal number of particles time bins.

is determined. To maintain the efficiency of the Simò integrator, ρ_i is first estimated at order 2 as before. If the order reaches half the maximum allowed order (set by the user) and the required accuracy has not been achieved yet, the estimation of ρ_i is updated at this order only for particle i. If this updated estimation still does not achieve the required accuracy as the order increases, another estimation is performed at the maximum allowed order.

An additional efficiency in the Simò integrator is provided through the use of time bins. After all the Taylor polynomials of the solutions are generated and the optimal time steps and orders are calculated, particles are distributed over a number of time bins based on their optimal time steps. The first bin contains the particles that have the smallest propagation time steps. At the end of each time step, the first bin particles are propagated, and will need new expansions to be computed in the next step while the other particles in other bins will keep their computed expansions. All the particles will be propagated to the final simulation time in the final step.

Moreover, the Simò integrator implements two types of time bins: equal width time bins and equal number of particles time bins. The efficiency of the Simò integrator can be enhanced by the appropriate choice of the number and type of time bins. In general, more particles in the first bin means more computational time. However, less particles in the first bin means less computational time per step but requires more time steps to integrate the total simulation time, which could increase the computational time. This suggests that the computational time starts large for small number of time bins and decreases until it reaches a minimum, and then increases again for larger number of time bins. Figure 21 shows this behavior for two beams of 25×10^3 particles with the same initial conditions except for their spatial distributions. In this case, the optimal time steps will mainly depend on the distances between the particles where uniformly distributed particles have comparable optimal time steps. In a Gaussian distribution, the particles time steps vary with small ones for the core particles and larger ones for the particles outside the core. As a result, the variation of the computational time with the number of bins is more noticeable for the Gaussian distribution than the uniform distribution in both types of bins as shown in Figs. 21(a) and 21(b).

APPENDIX C: SOFTENED COULOMB POTENTIAL OF DA-FMM

In its usual incarnation, the FMM algorithm allows the inclusion of all collisional effects. However, if only the mean fields are of interest, accounting for the strong effects due to close encounters would cause unphysical behavior. To avoid any unphysical effects, it us customary to smooth the Coulomb potential. Unfortunately, the standard FMM cannot tackle this problem. Fortunately, our DA based FMM allows easy employment of a softened (smoothed) Coulomb potential and without altering the algorithm [4,18]. The softening of the Coulomb potential is performed by introducing a Plummer-like softening (smoothing) parameter, denoted by λ here. The softened 2D Coulomb potential at a target (x, y) can be written as

$$V(x,y,\lambda) = -\frac{1}{2}\sum_{i=1}^{N} \log[(x_i - x)^2 + (y_i - y)^2 + \lambda^2], \quad \text{(C1)}$$

where $r = \sqrt{x^2 + y^2 + \lambda^2}$. With the DA variables $d_x = x/r^2$, $d_y = y/r^2$, $d_r^2 = 1/r^2$, and $d_\lambda = \lambda/r^2$, the resulting potential is

$$V(x,y,\lambda) = -\frac{1}{2}\sum_{i=1}^{N}\{\log(x^2+y^2+\lambda^2) + \log[1+(x_i^2+y_i^2)d_r^2 - 2(x_i d_x + y_i d_y)]\}. \quad \text{(C2)}$$

The rest of the algorithm proceeds in similar fashion to the usual DA-FMM algorithm as described in [4,18].

[1] A. W. Chao and M. Tigner, *Handbook of Accelerator Physics and Engineering* (World Scientific, Singapore, 1999).

[2] C. E. Mitchell, R. D. Ryne, and K. Hwang, Spectral galerkin solver for intense beam vlasov equilibria in nonlinear constant focusing channels, Phys. Rev. E **100**, 053308 (2019).

[3] J. Qiang, Symplectic particle-in-cell model for space-charge beam dynamics simulation, Phys. Rev. Accel. Beams **21**, 054201 (2018).

[4] A. Gee and B. Erdelyi, Smooth fast multipole method for space charge tracking: An alternate to particle-in-cell, in *Proceedings of the 6th International Particle Accelerator Conference (IPAC'15), Richmond, Virginia, USA, May 3-8, 2015* (JACoW, Geneva, 2015), pp. 663–665.

[5] B. Erdelyi, E. Nissen, and S. Manikonda, A differential algebraic method for the solution of the poisson equation for charged particle beams, Commun. Comput. Phys. **17**, 47 (2015).

[6] J. D. Jackson, *Classical Electrodynamics*, 3rd ed. (Wiley, New York, NY, 1999).

[7] J.-L. Vay, C. G. R. Geddes, E. Cormier-Michel, and D. P. Grote, Effects of hyperbolic rotation in Minkowski space on the modeling of plasma accelerators in a Lorentz boosted frame, Phys. Plasmas **18**, 030701 (2011).

[8] W. Qiu-Dong, The global solution of the N-body problem, Celest. Mech. Dyn. Astron. **50**, 73 (1991).

[9] J. Struckmeier, Stochastic effects in real and simulated charged particle beams, Phys. Rev. ST Accel. Beams **3**, 034202 (2000).

[10] M. Bai, D. Jeon, S. Y. Lee, K. Y. Ng, A. Riabko, and X. Zhao, Stochastic beam dynamics in quasi-isochronous storage rings, Phys. Rev. E **55**, 3493 (1997).

[11] J. Qiang, R. D. Ryne, and S. Habib, Self-consistent langevin simulation of coulomb collisions in charged-particle beams, in *SC '00: Proceedings of the 2000 ACM/IEEE Conference on Supercomputing, Dallas, Texas, USA, November 4-10, 2000* (IEEE Computer Society, New York, 2000).

[12] A. I. Kozynchenko and S. A. Kozynchenko, About improving efficiency of the p3m algorithms when computing the inter-particle forces in beam dynamics, Comput. Phys. Commun. **212**, 47 (2017).

[13] G. I. Bell, D. L. Bruhwiler, A. Fedotov, A. Sobol, R. S. Busby, P. Stoltz, D. T. Abell, P. Messmer, I. Ben-Zvi, and V. Litvinenko, Simulating the dynamical friction force on ions due to a briefly co-propagating electron beam, J. Comput. Phys. **227**, 8714 (2008).

[14] J. Barnes and P. Hut, A hierarchical o(n log n) force-calculation algorithm, Nature (London) **324**, 446 (1986).

[15] L. Hernquist, Hierarchical n-body methods, Comput. Phys. Commun. **48**, 107 (1988).

[16] L. Greengard and V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. **73**, 325 (1987).

[17] A. Arnold, F. Fahrenberger, C. Holm, O. Lenz, M. Bolten, H. Dachsel, R. Halver, I. Kabadshow, F. Gähler, F. Heber, J. Iseringhausen, M. Hofmann, M. Pippig, D. Potts, and G. Sutmann, Comparison of scalable fast methods for long-range interactions, Phys. Rev. E **88**, 063308 (2013).

[18] S. Abeyratne, A. Gee, and B. Erdelyi, An adaptive fast multipole method in cartesian basis, enabled by algorithmic differentiation, Commun. Nonlinear Sci. Numer. Simul. **72**, 294 (2019).

[19] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer Ser. Comput. Math., Vol. 31 (Springer, Berlin, Heidelberg, 2006).

[20] S. J. Aarseth, *Gravitational N-Body Simulations: Tools and Algorithms*, Cambridge Monographs on Mathematical Physics (Cambridge University Press, Cambridge, England, 2003).

[21] W. Dehnen and J. I. Read, N-body simulations of gravitational dynamics, Eur. Phys. J. Plus **126**, 55 (2011).

[22] C. Simó, in *Global Analysis of Dynamical Systems*, edited by B. K. H. W. Broer and G. Vegter (IOP, United Kingdom, 2001), pp. 373–389.

[23] M. Berz, *Modern Map Methods in Particle Beam Physics*, Advances in Imaging and Electron Physics (Academic Press, San Diego, CA, 1999).

[24] A. Al Marzouk and B. Erdelyi, Collisional n-body numerical integrator with applications to charged particle dynamics, SIAM J. Sci. Comput. **40**, B1517 (2018).

[25] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. **5**, 506 (1968).

[26] B. Erdélyi and M. Berz, Optimal Symplectic Approximation of Hamiltonian Flows, Phys. Rev. Lett. **87**, 114302 (2001).

[27] R. D. Ruth, A canonical integration technique, IEEE Trans. Nucl. Sci. **30**, 2669 (1983).

[28] E. Forest and R. D. Ruth, Fourth-order symplectic integration, Physica D: Nonlinear Phenomena **43**, 105 (1990).

[29] H. Yoshida, Construction of higher order symplectic integrators, Phys. Lett. A **150**, 262 (1990).

[30] M. Suzuki, Fractal decomposition of exponential operators with applications to many-body theories and monte carlo simulations, Phys. Lett. A **146**, 319 (1990).

[31] M. Suzuki, General theory of fractal path integrals with applications to many-body theories and statistical physics, J. Math. Phys. (N.Y.) **32**, 400 (1991).

[32] J. Candy and W. Rozmus, A symplectic integration algorithm for separable Hamiltonian functions, J. Comput. Phys. **92**, 230 (1991).

[33] K. Modin, On explicit adaptive symplectic integration of separable hamiltonian systems, Proceedings of the Institution of Mechanical Engineers, *Part K: Journal of Multi-body Dynamics* 222, 289 (2008).

[34] S. Abeyratne, B. Erdelyi, and H. D. Schaumburg, A novel code with high-order adaptive dynamics to solve the N-body problem, in *Proceedings of the 54th ICFA Advanced Beam Dynamics Workshop on High-Intensity and High-Brightness Hadron Beams (HB'14), East Lansing, MI, USA, November 10–14, 2014* (JACoW, Geneva, 2014), pp. 70–73.

[35] S. Abeyratne and B. Erdelyi, N-body code to demonstrate electron cooling, in *Proceedings of the International Workshop on Beam Cooling and Related Topics (COOL'15), Newport News, VA, USA, September 28–October 2, 2015* (JACoW, Geneva, 2016), pp. 59–62.

[36] S. Abeyratne and B. Erdelyi, The first particle-based proof of principle numerical simulation of electron cooling, in *Proceedings of the North American Particle Accelerator*

Conference (NAPAC'16), Chicago, IL, USA, October 9–14, 2016 (JACoW, Geneva, 2017), pp. 1241–1244.

[37] S. Abeyratne, New Computational Approaches to the N-body Problem with Applications to Electron Cooling of Heavy Ion Beams, Ph.D. thesis, Northern Illinois University, DeKalb, IL (2016).

[38] B. Cipra, The best of the 20th century: Editors name top 10 algorithms, SIAM News **33**, 1 (2000).

[39] K. Makino and M. Berz, COSY INFINITY version 9, Nucl. Instrum. Methods Phys. Res., Sect. A **558**, 346 (2006).

[40] H. D. Schaumburg, A. Al Marzouk, and B. Erdelyi, Picard iteration-based variable-order integrator with dense output employing algorithmic differentiation, Numerical Algorithms **80**, 377 (2019).

[41] À. Jorba and M. Zou, A software package for the numerical integration of ODEs by means of high-order Taylor methods, Exp. Math. **14**, 99 (2005).

[42] C. A. R. Hoare, Algorithm 64: Quicksort, Commun. ACM **4**, 321 (1961).

[43] R. Sedgewick, Implementing quicksort programs, Commun. ACM **21**, 847 (1978).

[44] S. Machida, C. Prior, S. Gilardoni, M. Giovannozzi, A. Huschauer, and S. Hirlander, Numerical investigation of space charge effects on the positions of beamlets for transversely split beams, Phys. Rev. Accel. Beams **20**, 121001 (2017).

[45] N. Lee, D. R. Puri, A. I. Blanco, and K. S. C. Chao, Intensity-modulated radiation therapy in head and neck cancers: An update, Head Neck: J. Sci. Spec. Head and Neck **29**, 387 (2007).

[46] D. Murphy, R. W. Speirs, D. V. Sheludko, C. T. Putkunz, A. J. McCulloch, B. M. Sparkes, and R. E. Scholten, Detailed observation of space– charge dynamics using ultracold ion bunches, Nat. Commun. **5** (2014).

[47] A. Feist, N. Bach, N. Rubiano da Silva, T. Danz, M. Möller, K. E. Priebe, T. Domröse, J. G. Gatzmann, S. Rost, J. Schauss, S. Strauch, R. Bormann, M. Sivis, S. Schäfer, and C. Ropers, Ultrafast transmission electron microscopy using a laser-driven field emitter: Femtosecond resolution with a high coherence electron beam, Ultramicroscopy **176**, 63 (2017).

[48] A. Lueangaramwong, G. Andonian, and P. Piot, Application of transverse-to-longitudinal phase-space-exchanged beam produced from a nano-structure photocathode to a soft x-ray free-electron laser, in *Proceedings of the 9th International Particle Accelerator Conference (IPAC'18), Vancouver, BC, Canada, April 29–May 4, 2018* (JACoW, Geneva, 2018), pp. 3379–3381.

[49] A. Tencate and B. Erdelyi, A high-precision emission computational model for ultracold electron sources, in *Proceedings of the 2019 North American Particle Accelerator Conference, (NAPAC'19), Lansing, MI, USA, September 1–6, 2019* (JACoW, Geneva, 2019), pp. 622–625.

[50] Y. S. Derbenev, On possibilities of fast cooling of heavy particle beams, AIP Conf. Proc. **253**, 103 (1992).

[51] V. Litvinenko and Y. Derbenev, Free electron lasers and high-energy electron cooling, in *Proceedings of FEL 2007, Novosibirsk, Russia, August 26–31, 2007* (JACoW, 2007).

[52] V. N. Litvinenko and Y. S. Derbenev, Coherent Electron Cooling, Phys. Rev. Lett. **102**, 114801 (2009).

[53] G. Wang, Y. Hao, V. Litvinenko, S. Reiche, D. L. Bruhwiler, I. V. Pogorelov, and B. T. Schwartz, Simulations of a single-pass through a coherent electron cooler for 40 Gev/n Au+79, in *Proceedings of the 2011 Particle Accelerator Conference (PAC'11), New York, NY, USA, March 28–April 1, 2011* (IEEE, New York, 2011).

[54] J. Ma, X. Wang, G. Wang, K. Yu, R. Samulyak, and V. Litvinenko, Simulation studies of modulator for coherent electron cooling, Phys. Rev. Accel. Beams **21**, 111001 (2018).

[55] G. I. Budker, An effective method of damping particle oscillations in proton and antiproton storage rings, Sov. At. Energy **22**, 438 (1967).

[56] G. I. Budker, N. S. Dikanskij, D. V. Pestrikov, I. N. Meshkov, V. I. Kudelainen, B. N. Sukhina, V. V. Parkhomchuk, and A. N. Skrinsky, Experimental studies of electron cooling, Part. Accel. **7**, 197 (1976).

[57] G. I. Bell, D. L. Bruhwiler, A. Fedotov, A. Sobol, R. S. Busby, P. Stoltz, D. T. Abell, P. Messmer, I. Ben-Zvi, and V. Litvinenko, Simulating the dynamical friction force on ions due to a briefly co-propagating electron beam, J. Comput. Phys. **227**, 8714 (2008).

[58] A. Sobol, D. Bruhwiler, G. Bell, A. Fedotov, and V. Litvinenko, Numerical calculation of dynamical friction in electron cooling systems, including magnetic field perturbations and finite time effects, New J. Phys. **12**, 093038 (2010).

[59] Y. S. Derbenev and A. Skrinsky, The effect of an accompanying magnetic field on electron cooling, Part. Accel. **8**, 235 (1978).