SIAM J. SCI. COMPUT.
Vol. 42, No. 1, pp. B299–B326

VLASOV–POISSON SYSTEM TACKLED BY PARTICLE SIMULATION UTILIZING BOUNDARY ELEMENT METHODS*

TORSTEN KEßLER†, SERGEJ RJASANOW†, AND STEFFEN WEIßER†

Abstract. This paper presents a grid-free simulation algorithm for the fully three-dimensional Vlasov–Poisson system for collisionless electron plasmas. We employ a standard particle method for the numerical approximation of the distribution function. Whereas the advection of the particles is grid-free by its very nature, the computation of the acceleration involves the solution of the nonlocal Poisson equation. To circumvent a volume mesh, we utilize the fast boundary element method, which reduces the three-dimensional Poisson equation to a system of linear equations on its two-dimensional boundary. This gives rise to fully populated matrices which are approximated by the $\mathcal{H}^2$-technique, reducing the computational time from quadratic to linear complexity. The approximation scheme based on interpolation has been shown to be robust and flexible, allowing a straightforward generalization to vector-valued functions. In particular, the Coulomb forces acting on the particles are computed in linear complexity. In first numerical tests, we validate our approach with the help of classical nonlinear plasma phenomena. Furthermore, we show that our method is able to simulate electron plasmas in complex three-dimensional domains with mixed boundary conditions in linear complexity.

Key words. Vlasov–Poisson system, simulation of plasmas, particle method, boundary element method, hierarchical approximation

AMS subject classifications. 35Q83, 65Z05, 65N75, 65N38, 68W25

DOI. 10.1137/18M1225823

1. Introduction.

The rapid increase in computational power in recent years due to massively parallel machines like clusters or GPUs has opened up the possibility of handling complex problems for a broad range of applications utilizing classical particle methods. Readily implemented in a computer program, they are extensible and applicable to computational problems in biology, chemistry, and physics.

Particle methods for the simulation of collisionless plasmas have been used since the 1950s, starting with the particle in cell method (PIC). We refer the reader to the classical textbooks [6, 27] for an introduction to the basic concepts and the history of the PIC method. The review articles [21] and, more recently, [44] discuss advanced aspects of plasma simulations with particle methods. An obvious strategy for simulation of the particle system is a direct summation. The force acting on a particle is determined by a summation over all interaction partners. Since particles in a plasma interact via long-range Coulomb forces, an accurate computation of the acceleration of a single particle requires a summation over all other particles in the plasma. This results in a quadratic computational complexity, which is prohibitively expensive with present computer hardware, even for medium-sized problems. Therefore, it is key to find approximations to the forces which significantly reduce the computational complexity but, at the same time, preserve their long-range character and produce consistent results.

Barnes and Hut [2] proposed an approximation scheme for gravitational problems which they called treecode. Their idea is to recursively subdivide the particle system

*Submitted to the journal's Computational Methods in Science and Engineering section November 8, 2018; accepted for publication (in revised form) November 22, 2019; published electronically February 18, 2020.

https://doi.org/10.1137/18M1225823

†Department of Mathematics, Saarland University, 66123 Saarbrücken, Germany (kessler@num.uni-sb.de, rjasanow@num.uni-sb.de, weisser@num.uni-sb.de).

into nested boxes. In three dimensions each box is split into eight boxes along the Cartesian axes. The recursive subdivision is embedded into a tree structure, from now on referred as the cluster tree. The typical depth of the cluster tree is $\mathcal{O}(\log N_p)$. The acceleration of a particle p is computed by iterating through the cluster tree, starting at its root. The forces between p and all particles in a well-separated cluster are replaced by a single force between p and a pseudoparticle at the center of mass of the cluster with mass equal to the total mass of all particles in this cluster. This generalizes easily to electrostatic problems, where the total mass has to be replaced by the total charge of the cluster. As the cluster tree has a depth of $\mathcal{O}(\log N_p)$, the numerical work for the treecode algorithm is $\mathcal{O}(N_p \log N_p)$. A very similar idea was proposed by Appel [1] with two major differences. First, he uses a binary tree, splitting boxes based on the medians of positions of the particles, and second, he avoids rebuilding the cluster tree after each time step by a merging strategy for clusters. Again, his algorithm has a complexity of $\mathcal{O}(N_p \log N_p)$. Both methods only use the monopole moment of the particle distribution for the approximation of the forces. This leads to relatively high errors, especially in the case of nonuniform particle distributions. However, both methods can be extended to include further terms of the Taylor expansion. Computations with Taylor expansions up to order m have a complexity of $\mathcal{O}((m+1)^3 N_p \log N_p)$. As the error in the far field decays exponentially, m is chosen as $m \sim |\log \varepsilon|$, where ε is a predefined error threshold.

The fast multipole method (FFM), proposed in [23] for two-dimensional problems, and extended in [24] to three-dimensional problems, is also a tree-based method. In contrast to the treecode discussed above, the FFM uses a Taylor expansion of the Newton potential in spherical coordinates up to a given order m, a technique well known in electrostatics. Whereas in the treecode expansions in only one variable were used, the FFM simultaneously expands the potential in both variables in the far field. Combined with a suitable iteration through the cluster tree, the numerical cost for the force evaluation is in $\mathcal{O}((m+1)^3 N_p)$. In its first formulation, FFM was restricted to applications with Newton potentials. Later, it was expanded to general kernels in [46] and independently, Of, Steinbach, and Wendland used FFM for the fast solution of boundary integral equations for the Laplace equation [39] and elastostatics [38].

Another important application for the fast evaluation of Coulomb potentials is molecule dynamics simulations for crystalline structures. Usually, one neglects boundaries of the crystal and uses periodic boundary conditions for the molecules and their self-consistent electric field. Those give rise to an infinite sum for the electric potential which is split into two rapidly decaying sums. This is known as Ewald summation. Evaluating the full sum gives a complexity of $\mathcal{O}(N_p^2)$. Darden, York, and Pedersen [20] combined an interpolation scheme with the fast Fourier transform to reduce the complexity to $\mathcal{O}(N_p \log N_p)$. The approximation error depends on the number of interpolation points. However, their method is restricted to structured particle distributions and, more importantly, to periodic boundary conditions.

In this paper, we present a unified hierarchical framework for the grid-free simulation of plasma in the electrostatic case in bounded domains with the help of modern $\mathcal{H}^2$-matrices. Both the particle-particle and the particle-boundary interactions have linear complexity in the number of particles. We propose the usage of interpolation for the approximation in the far field. It is very easy to implement, as it only needs the value of a rather general kernel function at the interpolation points and, furthermore, it is directly applicable to the approximation of vector-valued functions. The contribution of the boundary values to the electric field are computed via the boundary element method (BEM), which only requires a discretization of the boundary of the

domain. This reduces the three-dimensional problem posed on the whole domain to a system of integral equations on a two-dimensional manifold. Similar ideas have already been presented in [18, 16, 17, 19]. The authors used a treecode-based approximation scheme with a boundary integral formulation to simulate plasmas in one- and two-dimensional domains. Altough theory predicts a complexity of $\mathcal{O}((m+1)^3 N_p \log N_p)$ for their algorithm, they numerically observe nearly linear scaling in the number of particles. As we are using $\mathcal{H}^2$-matrices, we conclude from the theory of hierarchical matrices that our algorithm has linear complexity, in both the number of particles and the number of elements of the surface mesh. This is supported by our numerical results. Additionally, we start with the representation formula for the Poisson equation and systematically approximate the discretized boundary integral operators by $\mathcal{H}^2$-matrices. In this way, we treat the particle and the boundary part evenly in terms of the approximation schemes we use.

This article is organized as follows. Section 2 reviews the Vlasov–Poisson system. The basic concepts of boundary integral equations and the BEM are given in section 3. In section 4, we discuss hierarchical approximation techniques for Nyström and Galerkin matrices. Important aspects of the implementation of our method in a computer program are presented in section 5. Numerical examples validating our approach are given in section 6.

2. Vlasov–Poisson system. If the characteristic velocity V_0 of the particle system is small compared to the speed of light c, the dynamics of N_p charged particles with positions $(x_i)_{i=1}^{N_p}$, velocities $(v_i)_{i=1}^{N_p}$, masses $(m_i)_{i=1}^{N_p}$, and charges $(q_i)_{i=1}^{N_p}$ is given by

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= \frac{q_i}{m_i} E(x_i), \quad i = 1, \dots, N_p, \end{aligned} \tag{1}$$

coupled with the electrostatic approximation of Maxwell's equations for the electric field E

$$\operatorname{div} E = \frac{1}{\varepsilon_0} \sum_{j=1}^{N_p} q_j \delta_{x_j}, \quad \operatorname{rot} E = 0, \tag{2}$$

where ε_0 is the electric constant.

Since $\operatorname{rot} E = 0$, there exists a scalar potential ϕ with $E = -\nabla\phi$. The electrostatic Maxwell's equations can then be expressed as a scalar Poisson equation,

$$-\Delta\phi = \frac{1}{\varepsilon_0} \sum_{j=1}^{N_p} q_j \delta_{x_j}.$$

Together with the decay condition for the electric field,

$$-\nabla\phi(x) = \mathcal{O}\left(\frac{1}{|x|^2}\right), \quad |x| \to \infty,$$

we obtain the unique solution by applying the Newton potential N,

$$\phi = N \frac{1}{\varepsilon_0} \sum_{j=1}^{N_p} q_j \delta_{x_j} = \frac{1}{\varepsilon_0} \sum_{j=1}^{N_p} q_j U(\cdot, x_j),$$

where U is the fundamental solution of the Laplace operator,

$$U(x,y) = \frac{1}{4\pi|x-y|}, \quad x \neq y, \tag{3}$$

and the Newton potential for smooth functions with compact support is given by

$$N\psi(x) = \int_{\mathbb{R}^3} U(y,x)\psi(y)\,\mathrm{d}y, \quad x \in \mathbb{R}^3, \tag{4}$$

which is extended to distributions by duality. The electric field has the form

$$E = -\nabla\phi = -\frac{1}{\varepsilon_0}\sum_{j=1}^{N_p} q_j \nabla U(\cdot, x_j). \tag{5}$$

Plugging (5) into (1) and excluding self-interactions yields

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= -\frac{1}{m_i}\sum_{\substack{j=1 \\ j\neq i}}^{N_p} \frac{q_i q_j}{\varepsilon_0} \nabla_{x_i} U(x_i, x_j), \quad i = 1, \ldots, N_p. \end{aligned} \tag{6}$$

This equation is not feasible to describe the time evolution of our plasma as N_p is in the order of 10^{20} and therefore out of reach for a direct simulation. We therefore ask for an appropriate limit $N_p \to \infty$ which would give us an easier-to-handle equation.

One possibility is known as the mean field [42] or pulverization limit [35], which is based on a special scaling of charges and masses. For simplicity let us assume that our plasma consists of only one species with charge q_0 and mass m_0. The masses and charges of the particles are scaled by $1/N_p$,

$$m_i = m_0/N_p, \; q_i = q_0/N_p, \quad i = 1, \ldots, N_p.$$

This changes (6) to

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= -\frac{1}{N_p}\frac{q_0^2}{m_0\varepsilon_0}\sum_{\substack{j=1 \\ j\neq i}}^{N_p} \nabla_{x_i} U(x_i, x_j), \quad i = 1, \ldots, N_p. \end{aligned} \tag{7}$$

In their pioneering work, Neunzert and Wick [32, 33, 34] (for an English version of their ideas see [31] and also [42] for a historic review) show the convergence of (7) for $N_p \to \infty$ to the solution $f : (0,\infty) \times \mathbb{R}^3 \times \mathbb{R}^3 \to [0,\infty)$ of the Vlasov–Poisson system,

$$\begin{gathered} \partial_t f + v \cdot \nabla_x f + \frac{q_0}{m_0} E \cdot \nabla_v f = 0, \\ E = -\nabla\phi, \\ -\Delta\phi = \frac{q_0}{\varepsilon_0}\int_{\mathbb{R}^3} f\,\mathrm{d}v, \quad -\nabla\phi(x) = \mathcal{O}\left(\frac{1}{|x|^2}\right), \quad |x| \to \infty, \end{gathered} \tag{8}$$

in the weak-$*$ topology of measures. They rely on a regularization G_ε of $G = -\nabla_x U$ which is assumed to be continuous and bounded. Possible choices for G_ε include a mollified version of G or the gradient of

$$U_\varepsilon(x,y) = \frac{1}{4\pi|x-y|+\varepsilon}, \quad x, y \in \mathbb{R}^3.$$

The parameter ε is time-dependent and tends to 0 as $t \to \infty$; see [22, 45] and the references cited therein. In a recent work, Lazarovici and Pickl [29] show convergence to the Vlasov–Poisson system for a scaling that only depends on N_p, $\varepsilon = N_p^{-1/3+o(1)}$. This is nearly optimal in the sense that the mean distance between two particles scales like $N_p^{-1/3}$. Their regularization of G has to satisfy three conditions:

1. $\exists c_1 > 0\, \forall x, y \in \mathbb{R}^3, x \neq y : |G_\varepsilon(x,y)| \leq c_1/|x-y|^2,\ |\nabla_x G_\varepsilon(x,y)| \leq c_1/|x-y|^3$,
2. $\forall x, y \in \mathbb{R}^3, |x-y| \geq \varepsilon : G_\varepsilon(x,y) = G(x,y)$,
3. $\exists c_2 > 0\, \forall x, y \in \mathbb{R}^3, |x-y| < \varepsilon : |G_\varepsilon(x,y)| \leq c_2/\varepsilon^2,\ |\nabla_x G_\varepsilon(x,y)| \leq c_2/\varepsilon^3$.

In order to systematically derive suitable regularizations for the interaction force G, we can also regularize the charge distribution. Applying the Newton potential (4) then gives a regularization which is consistent with the charge density in the Poisson equation. Furthermore, this allows us to apply the standard theory of Sobolev spaces for elliptic problems. For our implementation we choose a radial step function,

$$\delta_y^\varepsilon = \frac{1}{|B_\varepsilon(y)|} \mathbb{1}_{B_\varepsilon(y)}, \quad y \in \mathbb{R}^3,$$

for which we have

$$U_\varepsilon(x,y) = N\delta_y^\varepsilon(x) = \frac{1}{4\pi} \begin{cases} \dfrac{3}{2\varepsilon} - \dfrac{|x-y|^2}{2\varepsilon^3}, & |x-y| < \varepsilon, \\ \dfrac{1}{|x-y|}, & |x-y| \geq \varepsilon, \end{cases} \qquad x, y \in \mathbb{R}^3. \tag{9}$$

Applying the gradient to (9) yields

$$G_\varepsilon(x,y) = \frac{1}{4\pi} \begin{cases} \dfrac{1}{\varepsilon^3}(x-y), & |x-y| < \varepsilon, \\ \dfrac{x-y}{|x-y|^3}, & |x-y| \geq \varepsilon, \end{cases} \qquad x, y \in \mathbb{R}^3. \tag{10}$$

It is easy to check that G_ε is a bounded Lipschitz continuous function that satisfies the aforementioned conditions on the regularization. Furthermore, $\delta_y^\varepsilon \in L_2(\mathbb{R}^3)$ which simplifies the analysis in section 3 when working with trace operators.

In this paper, we consider the Vlasov–Poisson system (8) in a bounded domain Ω. Instead of decay conditions on the potential ϕ we now prescribe Dirichlet or Neumann conditions on the boundary $\partial\Omega$. Additionally, we also need boundary conditions for the distribution function f. We primarily choose absorption, i.e., $f = 0$ on $\partial\Omega$. In its nondimensional form, the Vlasov–Poisson system reads

$$\begin{aligned} \partial_t f(t,x,v) + v \cdot \nabla_x f(t,x,v) + E(t,x) \cdot \nabla_v f(t,x,v) &= 0, \\ E(t,x) &= -\nabla_x \phi(t,x), \\ -\Delta_x \phi(t,x) &= \frac{1}{\beta} \int_{\mathbb{R}^3} f(t,x,v)\, \mathrm{d}v, \end{aligned} \tag{11}$$

for $(t,x,v) \in (0,\infty) \times \Omega \times \mathbb{R}^3$. Here, $\beta = (\lambda_D/L_0)^2$ is the square of the nondimensional quotient of the Debye length

$$\lambda_D = \sqrt{\frac{\varepsilon_0 k_B T_0}{n_0 q_0^2}}$$

and the characteristic length L_0 of Ω. Furthermore, k_B is the Boltzmann constant and T_0, n_0, q_0 denote the characteristic temperature, particle density, and charge of

the plasma, respectively. Initial conditions for f are usually linear combinations of Maxwellians,

$$M_{\rho_m,V,T}(x,v) = \frac{\rho_m(x)}{(2\pi T(x))^{3/2}} \exp\left(-\frac{|v-V(x)|^2}{2T(x)}\right), \quad (x,v) \in \Omega \times \mathbb{R}^3,$$

where ρ_m is the mass density, V is the macroscopic bulk velocity, and T is the temperature distribution inside the plasma.

For the numerical treatment of (11), we sample f by N_p macroparticles,

$$f(t,\cdot,\cdot) \approx \frac{|\Omega|}{N_p} \sum_{j=1}^{N_p} \delta_{x_j(t)}\, \delta_{v_j(t)}, \quad t > 0, \tag{12}$$

and regularize the charge density

$$\rho(t,x) = q_0 \int_{\mathbb{R}^3} f(t,x,v)\,\mathrm{d}v \approx q_0 \frac{|\Omega|}{N_p} \sum_{j=1}^{N_p} \frac{1}{|B_\varepsilon(x_j(t))|}\, \mathbb{1}_{B_\varepsilon(x_j(t))}, \tag{13}$$

where $|\Omega|$ is the volume of the (rescaled) domain. The Vlasov equation for the approximation (12) is equivalent to a system of ODEs,

$$\begin{aligned} \dot{x}_i &= v_i, \\ \dot{v}_i &= -\frac{q_0}{m_0} \nabla\phi(x_i), \quad i = 1,\dots,N_p, \end{aligned}$$

where ϕ is the solution to the boundary value problem

$$\begin{aligned} -\Delta\phi &= \frac{q_0}{\beta} \frac{|\Omega|}{N_p} \sum_{j=1}^{N_p} \frac{1}{|B_\varepsilon(x_j(t))|}\, \mathbb{1}_{B_\varepsilon(x_j(t))} \quad \text{in } \Omega, \\ \phi &= g_D \qquad\qquad\qquad\qquad\qquad\qquad\quad \text{on } \Gamma = \partial\Omega, \end{aligned} \tag{14}$$

and we assume a Dirichlet problem for simplicity. Keeping in mind that a special solution to the above equation is given by the Newton potential (4), a particular solution ϕ_p of the Poisson equation above for a fixed time $t > 0$ is

$$\phi_p(t,x) = \frac{q_0}{\beta} \frac{|\Omega|}{N_p} \sum_{j=1}^{N_p} U_\varepsilon(x, x_j(t)), \quad x \in \Omega. \tag{15}$$

In order to find a solution of the boundary value problem (14) with the help of ϕ_p, we have to solve the auxiliary problem

$$\begin{aligned} -\Delta\phi_0 &= 0 \qquad\quad \text{in } \Omega, \\ \phi_0 &= g_D - \phi_p \quad \text{on } \Gamma = \partial\Omega. \end{aligned} \tag{16}$$

The solution of the original problem is now

$$\phi = \phi_0 + \phi_p, \tag{17}$$

and the electric field at the time t in the position of particle i is computed as

$$E(t,x_i(t)) = -\nabla\phi_0(t,x_i(t)) + \frac{q_0}{\beta} \frac{|\Omega|}{N_p} \sum_{\substack{j=1 \\ j\neq i}}^{N_p} G_\varepsilon(x_i(t), x_j(t)). \tag{18}$$

Note that this representation is consistent with the mean field scaling (7) that leads to the Vlasov–Poisson equation. For homogeneous boundary conditions, ϕ_0 scales like $1/N_p$. This is also true for a pure Neumann or a mixed boundary value problem.

Whereas the evaluation of ϕ_p is grid-free by its nature, the numerical treatment of (16) involves, as a rule, the discretization of the domain. For simple domains of toroidal or rectangular shape the discretization of the Poisson equation on structured grids leads to linear systems whose solutions are usually found by means of the fast Fourier transform [6]. In order to evaluate the electric field at the positions of the particles and to couple the charge density with the grid, a regularization of ρ_{total} is needed. The electric field at the positions of the particles is obtained by interpolation from the grid nodes. Structured meshes also work for complex domains, where one relies on so-called cut cells near the boundary; see [28] for the electrostatic case and [36]3 for the full Maxwell system. Without a suitable postprocessing, degenerated cut cells with small side lengths put an additional constraint on the CFL condition for an explicit scheme [28, 36]. Contrarily, for the proposed BEMs we use to solve (16), no volume discretization is needed and therefore we introduce no further restriction on the CFL condition. By the use of the representation formula, we can compute the electric field at each given point inside the domain, i.e., the positions of the particles. Note that this decouples the particle discretization of the distribution function and the discretization of the Poisson equation for the electric potential. The number of particles does not affect the accuracy of the electric field, which is only controlled by the mesh size of the boundary mesh. The particles move freely through the volume Ω. As written in (18), the electric field is split into two parts: the free space interaction of the particles which ignores boundary conditions and a correction term which solves (16) and depends on boundary conditions. For this, we propose BEMs. The accuracy of the electric field only depends on the error made in the approximation of $-\nabla\phi_0$ but not on the number of particles. In contrast to PIC methods, there is no rule of thumb connecting the number of particles and the number of triangles of the surface mesh. In the following sections, we first review BEMs and the discretization for mixed problems. With the notation we define there, we are able to give a first formulation of our algorithm with quadratic complexity in subsection 3.3. Afterward, we discuss how to accelerate it and to reduce the complexity from quadratic to linear in the number of particles and the number of triangles.

3. Boundary element method. The BEM is reviewed for the general Poisson problem with mixed boundary conditions on a bounded polyhedral domain $\Omega \subset \mathbb{R}^3$ with boundary $\Gamma = \partial\Omega$. Furthermore, $\Gamma = \overline{\Gamma}_D \cup \overline{\Gamma}_N$ is split into an open Dirichlet part Γ_D and an open Neumann part Γ_N.

We first define the abstract mathematical framework for BEM based on fractional Sobolev spaces on the boundary. The section concludes with the definition of potential operators. The reader interested in the discretization of boundary integral equations may skip this section and start with the section on Galerkin discretization.

3.1. Boundary integral equations. The classical theory for boundary integral equations is based on square-integrable functions and their (weak) derivatives. Let $L_2(\Omega)$ denote the space of square-integrable functions,

$$L_2(\Omega) = \left\{ f : \Omega \to \mathbb{R} : \int_\Omega |f(x)|^2 \, \mathrm{d}x < \infty \right\},$$

which is a Hilbert space with respect to the inner product

$$(f,g)_{L_2(\Omega)} = \int_\Omega f(x)g(x)\,\mathrm{d}x, \quad f,g \in L_2(\Omega).$$

Analogously, $L_2(\Gamma)$, the space of square-integrable functions on the boundary, is defined. The Sobolev space $H^1(\Omega)$ consists of functions in $L_2(\Omega)$ which have a weak gradient in $L_2(\Omega)$, i.e., for $f \in H^1(\Omega)$ there exists a $g \in L_2(\Omega)^3$ such that for all $\varphi \in C_0^\infty(\Omega)^3$

$$\int_\Omega g(x)\cdot\varphi(x)\,\mathrm{d}x = -\int_\Omega f(x)\nabla\cdot\varphi(x)\,\mathrm{d}x,$$

where $C_0^\infty(\Omega)$ denotes the set of infinitely often differentiable functions with compact support in Ω. Equipped with the inner product

$$(f,g)_{H^1(\Omega)} = (f,g)_{L_2(\Omega)} + (\nabla f, \nabla g)_{L_2(\Omega)^3},$$

the Sobolev space $H^1(\Omega)$ is a Hilbert space. Similar spaces, called Sobolev–Slobodekii or fractional Sobolev spaces, can be defined on the boundary [43],

$$H^{1/2}(\Gamma) = \left\{ f \in L_2(\Gamma) : \int_\Gamma\int_\Gamma \frac{|f(x)-f(y)|^2}{|x-y|^3}\,\mathrm{d}s_y\,\mathrm{d}s_x < \infty \right\},$$

which forms a Hilbert spaces with inner product

$$(f,g)_{H^{1/2}(\Gamma)} = (f,g)_{L_2(\Gamma)} + \int_\Gamma\int_\Gamma \frac{(f(x)-f(y))(g(x)-g(y))}{|x-y|^3}\,\mathrm{d}s_y\,\mathrm{d}s_x,\ f,g \in H^{1/2}(\Gamma).$$

Spaces with negative indices like $H^{-1/2}(\Gamma)$ are defined as the dual spaces with respect to $\langle\cdot,\cdot\rangle_\Gamma$, the extension of the $L_2(\Gamma)$-inner product. The norm in $H^{-1/2}(\Gamma)$ is given by

$$\|u\|_{H^{-1/2}(\Gamma)} = \sup_{f\in H^{1/2}(\Gamma)\setminus\{0\}} \frac{\langle f,u\rangle_\Gamma}{\|f\|_{H^{1/2}(\Gamma)}}, \quad u \in H^{-1/2}(\Gamma).$$

For example, $H^{1/2}(\Gamma)$ contains continuous functions but not piecewise continuous functions with discontinuous jumps. But those functions are included in $H^{-1/2}(\Gamma)$. This is an important observation for choosing ansatz and test functions for the Galerkin formulation later. For a mixed formulation, we also need fractional Sobolev spaces on open subsets Γ_1 of the boundary. Then, $H^{1/2}(\Gamma_1)$ is defined by restrictions of functions from $H^{1/2}(\Gamma)$,

$$H^{1/2}(\Gamma_1) = \{ f|_{\Gamma_1} : f \in H^{1/2}(\Gamma_1)\},$$

with norm

$$\|g\|_{H^{1/2}(\Gamma_1)} = \inf_{f|_{\Gamma_1}=g} \|f\|_{H^{1/2}(\Gamma_1)}, \quad g \in H^{1/2}(\Gamma_1).$$

The space $H^{-1/2}(\Gamma_1)$ is formed by all continuous linear functionals acting on functions in $H^{1/2}(\Gamma)$ with support in Γ_1. Note that duality is understood with respect to $\langle\cdot,\cdot,\rangle_\Gamma$.

Given a volume source term $g_V \in L_2(\Omega)$, a Dirichlet datum $g_D \in H^{1/2}(\Gamma_D)$, as well as a Neumann datum $g_N \in H^{-1/2}(\Gamma_N)$, the Poisson problem reads

$$\begin{aligned} -\Delta\phi &= g_V \quad \text{in } \Omega, \\ \phi &= g_D \quad \text{on } \Gamma_D, \\ n_\Omega\cdot\nabla\phi &= g_N \quad \text{on } \Gamma_N, \end{aligned} \tag{19}$$

where n_Ω denotes the outward unit normal vector on Γ. The boundary value problem is considered in the weak sense, such that the solution is sought in the Sobolev space $H^1(\Omega)$. We may follow the idea of the previous section and construct a particular solution ϕ_p in order to homogenize the right-hand side of the differential equation. An appropriate choice is the Newton potential

$$\phi_p(x) = (Ng_V)(x) = \int_\Omega U(x,y)\, g_V(y)\, \mathrm{d}y \quad \text{for } x \in \mathbb{R}^3, \tag{20}$$

where $U(x,y)$ is the fundamental solution given in (3). For $g_V = \rho_{\text{total}}/\beta$ we recover (15). The problem (19) has a unique solution that admits for $x \in \Omega$ the representation formula

$$\phi(x) = \int_\Gamma U(x,y)\gamma_1\phi(y)\, \mathrm{d}s_y - \int_\Gamma \gamma_{1,y}U(x,y)\gamma_0\phi(y)\, \mathrm{d}s_y + (Ng_V)(x), \tag{21}$$

where $\gamma_0\phi$ denotes the Dirichlet and $\gamma_1\phi$ the Neumann trace of the unknown solution ϕ. For sufficiently smooth data and $x \in \Gamma$ it holds that

$$\gamma_0\phi(x) = \phi\big|_\Gamma(x) \qquad \text{and} \qquad \gamma_1\phi(x) = \lim_{\Omega \ni \tilde{x} \to x} n_\Omega \cdot \nabla\phi(\tilde{x}).$$

These trace operators can be extended to linear bounded operators with the following mapping properties [30]:

$$\gamma_0 : H^1(\Omega) \to H^{1/2}(\Gamma) \qquad \text{and} \qquad \gamma_1 : H^1_\Delta(\Omega) \to H^{-1/2}(\Gamma),$$

where $f \in H^1_\Delta(\Omega)$ iff $f \in H^1(\Omega)$ and $\Delta f \in L_2(\Omega)$. We apply the trace operators to the representation formula (21) and obtain the system of equations

$$\begin{pmatrix} \gamma_0\phi \\ \gamma_1\phi \end{pmatrix} = \begin{pmatrix} \frac{1}{2}I - K & V \\ W & \frac{1}{2}I + K' \end{pmatrix} \begin{pmatrix} \gamma_0\phi \\ \gamma_1\phi \end{pmatrix} + \begin{pmatrix} N_0 g_V \\ N_1 g_V \end{pmatrix}. \tag{22}$$

This system contains the standard boundary integral operators which are well studied; see, e.g., [30, 41, 43]. For $x \in \Gamma$, we have the single-layer potential operator

$$(V\zeta)(x) = \gamma_0 \int_\Gamma U(x,y)\zeta(y)\, \mathrm{d}s_y \quad \text{for } \zeta \in H^{-1/2}(\Gamma),$$

the double-layer potential operator

$$(K\xi)(x) = \lim_{\varepsilon \to 0} \int_{y \in \Gamma : \|y-x\| \geq \varepsilon} \gamma_{1,y}U(x,y)\xi(y)\, \mathrm{d}s_y \quad \text{for } \xi \in H^{1/2}(\Gamma),$$

where $\gamma_{1,y}$ means that the Neumann trace γ_1 only acts on the y-variable, and the adjoint double-layer potential operator

$$(K'\zeta)(x) = \lim_{\varepsilon \to 0} \int_{y \in \Gamma : \|y-x\| \geq \varepsilon} \gamma_{1,x}U(x,y)\zeta(y)\, \mathrm{d}s_y \quad \text{for } \zeta \in H^{-1/2}(\Gamma),$$

as well as the hypersingular integral operator

$$(W\xi)(x) = -\gamma_1 \int_\Gamma \gamma_{1,y}U(x,y)\xi(y)\, \mathrm{d}s_y \quad \text{for } \xi \in H^{1/2}(\Gamma),$$

and $N_0 g_V = \gamma_0 N g_V$ as well as $N_1 g_V = \gamma_1 N g_V$.

3.2. Galkerin discretization. Obviously, if the traces $\gamma_0\phi$ and $\gamma_1\phi$ of the unknown solution ϕ are known, the representation formula (21) can be used to evaluate ϕ inside the domain Ω. However, these traces are only known on parts of the boundary according to (19). Thus, we aim to approximate them on the whole boundary Γ with the help of a Galerkin BEM, following [43]. Therefore, let Γ be meshed by a quasi-uniform, conforming surface triangulation $\mathcal{T} = \{\Gamma_k\}_{k=1}^{N_\Gamma}$ that is shape-regular in the sense of Ciarlet with N_Γ triangles and M_Γ nodes. We apply the conforming approximation spaces

$$S_h^0(\Gamma) = \operatorname{span}\left\{\varphi_k^0\right\}_{k=1}^{N_\Gamma} \subset H^{-1/2}(\Gamma), \quad \text{and} \quad S_h^1(\Gamma) = \operatorname{span}\left\{\varphi_i^1\right\}_{i=1}^{M_\Gamma} \subset H^{1/2}(\Gamma),$$

where φ_k^0 denotes the piecewise constant function that is one on the triangle of index k and zero else, and φ_i^1 denotes the usual hat function corresponding to the node with index i. For simplicity, we write $\phi = \gamma_0\phi$ and assume that the triangles and nodes are numbered in such a way that the triangles for $k = 1, \dots, N_D$ lie in Γ_D and the nodes for $i = 1, \dots, M_N$ are the ones without Dirichlet condition. We seek the approximation of the Dirichlet trace as

$$\phi_h(x) = \phi_{N,h}(x) + \phi_{D,h}(x) = \sum_{i=1}^{M_N} \phi_i \varphi_i^1(x) + \sum_{i=M_N+1}^{M_\Gamma} \phi_i \varphi_i^1(x) \tag{23}$$

and the Neumann trace as

$$t_h(x) = t_{D,h}(x) + t_{N,h}(x) = \sum_{k=1}^{N_D} t_k \varphi_k^0(x) + \sum_{k=N_D+1}^{N_\Gamma} t_k \varphi_k^0(x) \tag{24}$$

with vectors $\underline{\phi}_{N,h} = (\phi_1, \dots, \phi_{M_N})^\top \in \mathbb{R}^{M_N}$ and $\underline{t}_{D,h} = (t_1, \dots, t_{N_D})^\top \in \mathbb{R}^{N_D}$, respectively, and $\underline{\phi}_{D,h}$ and $\underline{t}_{N,h}$ accordingly. The coefficients ϕ_i, $i = M_N + 1, \dots, M_\Gamma$, and t_k, $k = N_D + 1, \dots, N_\Gamma$, are determined by interpolation of the given boundary data in (19). Inserting the ansatz (23) and (24) into (22), testing with φ_k^0, $k = 1, \dots, N_D$ and φ_i^1, $i = 1, \dots, M_N$, respectively, and integrating over Γ yields

$$\begin{aligned} &\begin{pmatrix} V_h^{DD} & -K_h^{DN} \\ {K_h^{DN}}^\top & W_h^{NN} \end{pmatrix} \begin{pmatrix} \underline{t}_{D,h} \\ \underline{\phi}_{N,h} \end{pmatrix} \\ &\qquad = \begin{pmatrix} \frac{1}{2} M_h^{DD} + K_h^{DD} & -V_h^{DN} \\ -W_h^{ND} & \frac{1}{2} {M_h^{NN}}^\top - {K_h^{NN}}^\top \end{pmatrix} \begin{pmatrix} \underline{\phi}_{D,h} \\ \underline{t}_{N,h} \end{pmatrix} - \begin{pmatrix} \underline{N}_0^D \\ \underline{N}_1^N \end{pmatrix}. \end{aligned} \tag{25}$$

The matrices are defined by

$$\begin{aligned} V_h[\ell, k] &= (V\varphi_k^0, \varphi_\ell^0)_{L_2(\Gamma)}, \quad W_h[j, i] = (D\varphi_i^1, \varphi_j^1)_{L_2(\Gamma)}, \\ K_h[\ell, i] &= (K\varphi_i^1, \varphi_\ell^0)_{L_2(\Gamma)}, \quad M_h[\ell, i] = (\varphi_i^1, \varphi_\ell^0)_{L_2(\Gamma)}, \end{aligned} \tag{26}$$

where $i, j = 1, \dots, M_\Gamma$ and $k, \ell = 1, \dots, N_\Gamma$, with the block structure

$$\begin{aligned} V_h &= \begin{pmatrix} V_h^{DD} & V_h^{DN} \\ V_h^{ND} & V_h^{NN} \end{pmatrix}, \quad W_h = \begin{pmatrix} W_h^{NN} & W_h^{ND} \\ W_h^{DN} & W_h^{DD} \end{pmatrix}, \\ K_h &= \begin{pmatrix} K_h^{DN} & K_h^{DD} \\ K_h^{NN} & K_h^{ND} \end{pmatrix}, \quad M_h = \begin{pmatrix} M_h^{DN} & M_h^{DD} \\ M_h^{NN} & M_h^{ND} \end{pmatrix}, \end{aligned} \tag{27}$$

representing the Dirichlet and Neumann boundary parts of the matrices. Fully written out, the entries for the single- and double-layer potential read

$$V_h[\ell,k] = \frac{1}{4\pi}\int_{\Gamma_\ell}\int_{\Gamma_k}\frac{1}{|x-y|}\,\mathrm{d}s_y\,\mathrm{d}s_x,$$

and

$$K_h[\ell,i] = \frac{1}{4\pi}\int_{\Gamma_\ell}\int_{\operatorname{supp}\varphi_i^1}\frac{(x-y)\cdot n_\Omega(y)}{|x-y|^3}\varphi_i^1(y)\,\mathrm{d}s_y\,\mathrm{d}s_x$$

for $k,\ell = 1,\dots,N_\Gamma$ and $i = 1,\dots,M_\Gamma$. Note that these integrals are singular if the supports of ansatz and test functions overlap. Therefore, special quadratures are used to accurately compute the matrix entries. The most general method, which is sometimes called black box quadrature, was developed by Sauter and Schwab; see [41]. It is based on a suitable regularization of the kernel function and utilizes tensorized Gauss–Legendre quadrature on $[-1,1]^4$. The quadrature error is proven to decay exponentially with the number of quadrature points.

Furthermore, we used

$$\underline{N}_0[\ell] = (N_0 g_V, \varphi_\ell^0)_{L_2(\Gamma)} \quad \text{with} \quad \underline{N}_0 = (\underline{N}_0^D, \underline{N}_0^N)^\top$$

for $\ell = 1,\dots,N_\Gamma$ and

$$\underline{N}_1[j] = (N_1 g_V, \varphi_j^1)_{L_2(\Gamma)} \quad \text{with } \underline{N}_1 = (\underline{N}_1^N, \underline{N}_1^D)^\top$$

for $j = 1,\dots,M_\Gamma$. Since in our case $N_0 g_V$ is computed easily using (20), we exploit the identity

$$N_1 g_V = \left(-\tfrac{1}{2}I + K'\right) V^{-1} N_0 g_V \tag{28}$$

in order to approximate $\underline{N}_1$ and to avoid volume integrals. We refer the interested reader to [37] for more details.

For a pure Dirichlet problem, i.e., $\Gamma_N = \varnothing$, the system reduces to

$$V_h \underline{t}_h = \left(\tfrac{1}{2}M_h + K_h\right)\underline{\phi}_h - \underline{N}_0. \tag{29}$$

We can omit the Newton potential when utilizing the proposed decomposition (17) with ϕ_0 as the solution of (16). This ansatz yields for the approximation of the Neumann trace $t_{0,h} \approx \gamma_1\phi_0$ the system of linear equations

$$V_h \underline{t}_{0,h} = \left(\tfrac{1}{2}M_h + K_h\right)\underline{\phi}_{0,h},$$

where $\phi_{0,h} \approx \gamma_0\phi_0 = g_D - \gamma_0 u_p$.

For a pure Neumann problem, i.e., $\Gamma_D = \varnothing$, the system also reduces. The hypersingular integral operator, however, is not invertible on $H^{1/2}(\Gamma)$ and thus, the stabilized system [43]

$$\widetilde{W}_h \underline{\phi}_h = \left(\tfrac{1}{2}M_h^\top - K_h^\top\right)\underline{t}_h - \underline{N}_1 \tag{30}$$

is considered, where

$$\widetilde{W}_h = W_h + \alpha\, d_h\, d_h^\top \quad \text{with} \quad d_h[i] = \left(\varphi_i^1, 1\right)_{L^2(\Gamma)},$$

and stabilization parameter $\alpha > 0$. The system (30) is uniquely solvable since the matrix $\widetilde{W}_h$ is symmetric and positive definite due to the properties of the integral operator W. Furthermore the stabilization ensures that

$$\int_\Gamma t_h(x)\,\mathrm{d}s_x = 0.$$

At the end of this section, we briefly discuss the approximation error of the Galerkin method. Since we are only interested in point values of the solution in the interior of the domain, we focus on pointwise error estimates. We cite the main results and do not give all necessary conditions for the following theorems to hold. The reader is referred to [41, 43] for more details.

LEMMA 3.1. *For the numerical approximations* ϕ_h (23) *and* t_h (24) *the error estimates*

$$\|t_h - \gamma_1\phi\|_{L_2(\Gamma)} = \mathcal{O}(h), \quad \|\phi_h - \gamma_0\phi\|_{L_2(\Gamma)} = \mathcal{O}(h^2)$$

hold, where h *is the mesh size of* $\mathcal{T}$.

From Lemma 3.1 pointwise error estimates follow.

LEMMA 3.2. *For* $x \in \Omega$ *there are* $C_1, C_2 > 0$ *such that*

$$|\phi(x) - \widetilde{\phi}_h(x)| \le C_1 h^3 \quad |\nabla\phi(x) - \nabla\widetilde{\phi}_h(x)| \le C_2 h^3$$

for the pure Dirichlet or Neumann problem. For the mixed problem we have at least quadratic convergence. Here, $\widetilde{\phi}_h$ *denotes the function obtained by plugging* ϕ_h *and* t_h *into the representation formula* (21).

3.3. Application to the Vlasov–Poisson system. We now specify the choice of the data for the general Poisson equation (19) for the self-consistent field of the particles and give a first naive version of our algorithm for the computation of the electric field in Algorithm 3.1. The volume term g_V is proportional to ρ from (13), i.e.,

$$g_V = \frac{q_0}{\beta}\frac{|\Omega|}{N_p}\sum_{j=1}^{N_p}\frac{1}{|B_\varepsilon(x_j|}\mathbb{1}_{B_\varepsilon(x_j)},$$

and therefore

$$N_0 g_V = \frac{q_0}{\beta}\frac{|\Omega|}{N_p}\sum_{i=1}^{N_p} U_\varepsilon(\cdot, x_i)$$

with U_ε given by (9). The discretized Dirichlet trace of the Newton potential now is

$$\underline{N}_0[\ell] = \frac{q_0}{\beta}\frac{|\Omega|}{N_p}\sum_{j=1}^{N_p}\left(\int_{\Gamma_\ell} U_\varepsilon(y, x_j)\,\mathrm{d}s_y\right), \quad \ell = 1, \dots, N_\Gamma.$$

For an efficient implementation into a computer program and for a hierarchical approximation of the electric field, it is important to restate all operations as matrix-vector products. We begin with $\underline{N}_0$, whose computation is expressed as

$$\underline{N}_0 = \frac{1}{\beta}\Phi_{\mathcal{T}}\, w_q, \tag{31}$$

where $w_q = (q_0|\Omega|/N_p)_{i=1}^{N_p}$ is the vector of weighted charges and the entries of $\Phi_{\mathcal{T}}$ are given by

$$\Phi_{\mathcal{T}}[\ell, j] = \int_{\Gamma_\ell} U_\varepsilon(y, x_j)\,\mathrm{d}s_y$$

for $\ell = 1, \dots, N_\Gamma$ and $i = 1, \dots, N_p$. In a similar way we reformulate the gradient of the representation formula (21). For this, we define $F_k \in \mathbb{R}^{N_p \times N_p}$, $S_k \in \mathbb{R}^{N_p \times N_\Gamma}$, $D_k \in \mathbb{R}^{N_p \times M_\Gamma}$ for $k = 1, 2, 3$:

(32)
$$S_k[i, \ell] = \int_{\Gamma_\ell} \frac{\partial}{\partial x_i^{(k)}} U(x_i, y)\,\mathrm{d}s_y, \quad D_k[i, j] = \int_{\operatorname{supp}\varphi_j} \frac{\partial}{\partial x_i^{(k)}} \gamma_{1,y} U(x_i, y)\varphi_j(y)\,\mathrm{d}s_y$$

for $i = 1, \dots, N_p$, $\ell = 1, \dots, N_\Gamma$, and $j = 1, \dots, M_\Gamma$. Furthermore,

$$F_k[i, j] = \frac{\partial}{\partial x_i^{(k)}} U_\varepsilon(x_i, x_j) \tag{33}$$

for $i, j = 1, \dots, N_p$. The matrices S_k and D_k, $k = 1, 2, 3$, are the contributions of the single- and double-layer potentials to the gradient of the solution, respectively. The matrices $(F_k)_{k=1}^3$ represent the gradient of the Newton potential, i.e., the free space field of the particles. We collect the values of the electric field evaluated at the positions of the particles in three vectors,

$$\underline{E}_k[i] = E_k(x_i), \quad k = 1, 2, 3,\ i = 1, \dots, N_p.$$

With this notation, we formulate the computation of the electric field as a series of matrix-vector multiplications. The matrices in Algorithm 3.1 are densely populated; see (26), (32), and (33). Therefore, the algorithm scales like $\mathcal{O}(N_\Gamma^2 + N_\Gamma N_p + N_p^2)$ with a preprocessing step in the order of $\mathcal{O}(N_\Gamma^3)$ for computing the Cholesky decomposition of the single-layer operator V_h.

Algorithm 3.1 Grid-free evaluation of the electric field with quadratic complexity.

Require: Mesh $\mathcal{T} = \{\Gamma_k\}_{k=1}^{N_\Gamma}$, particles $(x_i)_{i=1}^{N_p}$, and matrices V_h, K_h, M_h, W_h.
  $\underline{N}_0 \leftarrow 1/\beta\, \Phi_{\mathcal{T}}\, w_q$
  if $\Gamma_N \neq \emptyset$ **then**
    $\underline{N}_1 \leftarrow (K_h^\top - \frac{1}{2} M_h^\top) V_h^{-1} \underline{N}_0$.
  end if
  if $\Gamma_N \neq \Gamma$ **then**
    Solve (25) for $\underline{\phi}_h$ and $\underline{t}_h$.
  else
    Solve (30) for $\underline{\phi}_h$.
  end if
  for $k = 1$ **to** 3 **do**
    $\underline{E}_k \leftarrow -S_k \underline{t}_h + D_k \underline{\phi}_h - 1/\beta F_k w_q$ {Note the sign change due to $E = -\nabla\phi$}
  end for
  return $\underline{E}_1, \underline{E}_2, \underline{E}_3$.

4. Hierarchical approximation. In this section, we discuss how to find approximations to the fully populated matrices needed for computation of the electric field in Algorithm 3.1.

A direct evaluation is quadratic in both memory and computational time, which can be large, even for a relatively small number of discretization parameters. With the special structure of most of the matrices, it is possible to reduce storage requirements and computational costs to linear complexity by means of hierarchical approximations of dense matrices, called $\mathcal{H}^2$-matrices.

The matrices given in subsections 3.2 and 3.3 fit in a larger framework of integrals and point evaluations of a general kernel function. To unify the treatment within the $\mathcal{H}^2$-technique, let us introduce a more general notation. For the rest of this section we fix two index sets $\mathcal{I}$ and $\mathcal{J}$, with associated sets $X \subset \mathbb{R}^3$ and $Y \subset \mathbb{R}^3$, representing particles, nodes, or triangles of the surface mesh.

The matrices $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$ arising from (33) are point evaluations of a kernel function k, so-called Nyström matrices,

$$A[i,j] = k(y_j, x_i), \quad i \in \mathcal{I},\, j \in \mathcal{J}, \tag{34}$$

where k is the fundamental solution (3) or one component of its gradient, and $X = (x_i)_{i\in\mathcal{I}}$, $Y = (y_j)_{j\in\mathcal{J}}$. The Galerkin-type BEM matrices from (26) have the form

$$A[i,j] = \int_\Gamma \int_\Gamma k(x,y)\varphi_j(y)\psi_i(x)\,\mathrm{d}s_y\,\mathrm{d}s_x, \quad i \in \mathcal{I},\, j \in \mathcal{J}, \tag{35}$$

with trial functions $(\varphi_j)_{j\in\mathcal{J}}$, whose supports are in $Y \subset \Gamma$ and test functions $(\psi_i)_{i\in\mathcal{I}}$ with supports in $X \subset \Gamma$. Again, k denotes the fundamental solution (3) or its normal derivative.

The $\mathcal{H}^2$-matrix approximation [8, 26] is a tree-based data structure which exploits low-rank factorizations of matrix blocks,

$$A\,|_{\sigma\times\tau} \approx V\Sigma U^\top, \tag{36}$$

where $\sigma \subset \mathcal{I}$, $\tau \subset \mathcal{J}$ represent parts of X and Y, respectively, which are far apart, a term specified in Definition 4.1. Furthermore, $V \in \mathbb{R}^{\sigma\times r}$, $U \in \mathbb{R}^{\tau\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$. The dimension of Σ, $r \in \mathbb{N}$, is called the rank of the approximation. To significantly reduce the storage requirements and the computational complexity, $r \ll \max\{\#\sigma, \#\tau\}$ must hold.

The triple (V, Σ, U) may be computed by a truncated singular value decomposition. Although this leads to the best compression rates, i.e., minimal storage complexity, this method still has overall quadratic computational complexity. It is therefore key to use a method which reduces both computational and storage complexity, preferably to linear cost. Over the years, several methods have been proposed, most notable Taylor expansion [26], multipole expansion [13, 15, 23, 24, 25], adaptive cross approximation [3, 4, 5, 40], interpolation [10, 12], hybrid cross approximation [11], and the Green hybrid method [9]. For our study, we choose an approximation based on interpolation, although other methods would also work. From a theoretical point of view, the complexity reduction from quadratic to linear can be best understood when utilizing interpolation. Practically, the interpolation scheme is readily implemented and very flexible as it works with point evaluations of the kernel function only. Before we give details on the approximation scheme, we define an admissibility condition for the far field and describe the necessary tree structures for the $\mathcal{H}^2$-format.

DEFINITION 4.1. *Suppose $\sigma \subset \mathcal{I}$ and $\tau \subset \mathcal{J}$ with corresponding subsets $X_\sigma \subset X$, $Y_\tau \subset Y$. The sets σ and τ are are η-admissible for $\eta > 0$ if*

$$\max\{\operatorname{diam}(X_\sigma), \operatorname{diam}(Y_\tau)\} \le \eta \operatorname{dist}(X_\sigma, Y_\tau).$$

Searching for the optimal partition of $\mathcal{I} \times \mathcal{J}$ in a sense that (36) holds for most blocks with minimal rank r is prohibitively expensive. Therefore, the partition of $\mathcal{I} \times \mathcal{J}$, called the block cluster tree, is constructed via partitions of $\mathcal{I}$ and $\mathcal{J}$. These are given as cluster trees; see Algorithm 4.1 for their construction. They start with the whole index set in their root. Step by step, further levels are added, where each level represents a disjoint union of the original set. The splitting of a node of the cluster tree stops if the number of indices is below a given threshold $n_{\min}$. For computational ease and also for the interpolation points used later, axis-parallel bounding boxes B are associated to all nodes of the cluster tree. This simplifies the computation of the admissibility condition a lot. The boxes in Algorithm 4.1 are usually split via principal component analysis or cardinality splitting. If one splits the boxes in each direction, the well-known octree is recovered.

Algorithm 4.1 Construction of a cluster tree.

Require: Index set $\mathcal{I}$, geometry X, axis-parallel box $B \supset X$
  if $\#\mathcal{I} < n_{\min}$ **then**
    return $T(\mathcal{I}) = (\mathcal{I}, B)$
  else
    Split B into disjoint bounding boxes $B_1, \dots, B_m$
    for $k = 1$ **to** n **do**
      Find maximal $\sigma_k \subset \mathcal{I}$ with $X_{\sigma_k} \subset B_k$
      Add son to tree by constructing a cluster tree for σ_k, X_{σ_k}, $B_{\sigma_k} = B_k$.
    end for
  end if
  return Cluster tree $T(\mathcal{I})$.

The block cluster tree contains the partition of $\mathcal{I} \times \mathcal{J}$ into nonadmissible (near field) and admissible (far field) blocks. It is defined as the cluster tree of $\mathcal{I} \times J$ with respect to Definition 4.1. Its construction is given in Algorithm 4.2.

Algorithm 4.2 Construction of a block cluster tree.

Require: cluster trees $T(\mathcal{I})$, $T(\mathcal{J})$ for geometries X, Y.
  $b \leftarrow$ isAdmissible$(\mathcal{I}, \mathcal{J})$ {Use condition from Definition 4.1.}
  if b **or** sons$(T(\mathcal{I})) = \emptyset$ **or** sons$(T(\mathcal{J})) = \emptyset$ **then**
    return $T(\mathcal{I}, \mathcal{J}) = \emptyset$
  else
    for σ **in** sons$(T(\mathcal{I}))$ **do**
      for τ **in** sons$(T(\mathcal{J}))$ **do**
        Add son by constructing a block cluster tree for $\sigma, \tau, X_\sigma, Y_\tau$.
      end for
    end for
  end if
  return $T(\mathcal{I} \times \mathcal{J})$

With the characterization of the admissible blocks, we are able to give the algorithm of low-rank approximations (36) by means of interpolation. Suppose $\sigma \times \tau \subset \mathcal{I} \times \mathcal{J}$ to be an admissible block. For $r = m^3 \in \mathbb{N}$, $(z_k)_{k=1}^r \subset [-1, 1]^3$ denotes

tensorized one-dimensional Chebyshev nodes,

$$\cos\left(\frac{2\ell-1}{2m}\pi\right), \quad \ell = 1, \dots, m.$$

These reference nodes are mapped to the boxes B_σ, B_τ, defining nodes $x^{(\sigma)}$, $y^{(\tau)}$, respectively, which then are used to define the tensorized Lagrange polynomials $(L_k^\sigma)_{k=1}^r$ and $(L_k^\tau)_{k=1}^r$. We now expand the kernel from (34) or (35) into the Lagrange basis,

$$k(y,x) \approx \sum_{k=1}^{r}\sum_{\ell=1}^{r} L_k^{(\sigma)}(x) k(y_\ell^{(\tau)}, x_k^{(\sigma)}) L_\ell^{(\tau)}(y).$$

Plugging this ansatz into (34), we obtain

$$A\,|_{\sigma\times\tau} \approx V_\sigma \Sigma_{\sigma\times\tau} U_\tau^\top,$$

where

$$V_\sigma[i,k] = L_k^{(\sigma)}(x_i), \quad U_\tau[j,\ell] = L_\ell^{(\tau)}(y_j),$$

and

$$\Sigma_{\sigma\times\tau}[k,\ell] = k(y_\ell^{(\tau)}, x_k^{(\sigma)}),$$

for $i \in \sigma, j \in \tau$, $k,\ell = 1,\dots,r$. Note that the interpolation matrices V_σ and W_τ only depend on the corresponding cluster but not on the block cluster $\sigma\times\tau$. When employing the interpolation-based approximation to Galkerin matrices (35), only the definitions of V_σ and U_τ change. In that case,

$$V_\sigma[i,k] = \int_\Gamma L_k^{(\sigma)}(x)\,\psi_i(x)\,\mathrm{d}s_x, \quad U_\tau[j,\ell] = \int_\Gamma L_\ell^{(\tau)}(y)\,\varphi_j(y)\,\mathrm{d}s_y$$

for $i \in \sigma, j \in \tau$, $k,\ell = 1,\dots,r$. The families $(V_\sigma)_{\sigma\in T(\mathcal{I})}$ and $(U_\tau)_{\tau\in T(\mathcal{J})}$ are called cluster bases for $T(\mathcal{I})$ and $T(\mathcal{J})$, respectively. An important property of the cluster bases is that they are nested in the following sense. Assume $\sigma \in T(\mathcal{I})$ to be a nonleaf node with sons $\sigma', \sigma'', \dots$.

We begin with the observation that $L^{(\sigma)}\,|_{X_{\sigma'}}$ and $L^{(\sigma')}$ both span the same polynomial space on $X_{\sigma'} \subset X_\sigma$. Written out, we have

$$L_\ell^{(\sigma)}(x) = \sum_{k=1}^{r} L_\ell^{(\sigma)}(x_k^{(\sigma')}) L_k^{(\sigma')}(x), \quad x \in X_{\sigma'},$$

for $\ell = 1,\dots,r$. Therefore,

$$V_\sigma\,|_{\sigma'\times r} = V_{\sigma'} E_{\sigma',\sigma},$$

where $E_{\sigma',\sigma}$ is called a transfer matrix. Its entries are given by

$$E_{\sigma,\sigma'}[k,\ell] = L_\ell^{(\sigma)}(x_k^{(\sigma')}), \quad k,\ell = 1,\dots,r.$$

Up to a permutation of the indices in σ, V_σ can be written blockwise as

$$V_\sigma = \begin{pmatrix} V_{\sigma'} E_{\sigma,\sigma'} \\ V_{\sigma''} E_{\sigma,\sigma''} \\ \vdots \end{pmatrix}.$$

This special format reduces the required storage as the cluster basis only depends on

small transfer matrices. The full knowledge of I and X is only needed in the leafs of the cluster tree. The nested structure of the cluster basis enables us to formulate a very efficient algorithm for matrix-vector multiplication. It is split into three parts. In the first step, called forward transform, we iterate through $T(\mathcal{J})$ and recursively collect the contributions of the transfer matrices. Multiplication with the matrix entries takes place in the interaction phase. In the last step we apply the backward transform for $T(\mathcal{I})$, which is similar to the first step. Before we give the algorithms for these three parts, let us introduce an auxiliary vector. For $u \in \mathbb{R}^{\mathcal{I}}$ we define

$$\hat{u} = \{u_\sigma \in \mathbb{R}^r : \sigma \in T(\mathcal{I})\}.$$

With a precomputed cluster basis, the first step of the $\mathcal{H}^2$-matrix-vector multiplication in Algorithm 4.5 is the forward transform from Algorithm 4.3, followed by the interaction phase which couples the input vector and the output vector via point evaluations of the kernel function. The result is then obtained by a backward transform applied to the output vector; see Algorithm 4.4. The contribution of the fully assembled nonadmissible blocks, i.e., the near field, is added in a final step. Due to the nested structure of the cluster bases, the cluster depth in order of $\log \#\mathcal{I}$ or $\log \#\mathcal{J}$ disappears from the complexity estimate. The proven storage and computational complexity estimates read as follows.

LEMMA 4.2. *Under mild assumptions on $T(\mathcal{I})$ and $T(\mathcal{J})$, the storage and therefore the computational complexity for Algorithm* 4.5 *are*

$$\mathcal{O}(r(\#I + \#J)).$$

Proof. For the proof and details on the assumptions on the cluster trees we refer the reader to [26]. ∎

Algorithm 4.3 Forward transform.

Require: cluster basis $(U_\tau)_{\tau \in T(\mathcal{J})}$, vector $w \in \mathbb{R}^{\mathcal{J}}$, index set $\mathcal{J}$.
  if $\mathcal{I}$ is a leaf node **then**
    $\hat{w}_{\mathcal{J}} \leftarrow U_{\mathcal{J}}^\top w\,|_{\mathcal{J}}$
  else
    for τ **in** sons($\mathcal{J}$) **do**
      forward transform $(U_\tau)_{\tau \in T(\mathcal{J})}$, w, τ
      $\hat{w}_{\mathcal{J}} \leftarrow \hat{w}_{\mathcal{J}} + E_{\mathcal{J},\tau}^\top \hat{w}_\tau$
    end for
  end if
  return $\hat{w}$

5. Notes on the implementation. In this section we discuss our scheme with regard to its implementation utilizing hierarchical matrices. We also give an overview of the employed software packages. All approximations with $\mathcal{H}^2$-matrices in this section use polynomial interpolation as discussed in section 4.

At each time step, the system of boundary integral equations (22) is solved to obtain the Dirichlet and Neumann traces for the representation formula (21). The matrices from the discrete formulation (26) only depend on the discretization of the boundary Γ but not on the Dirichlet or Neumann boundary data or the positions of the particles. Therefore they are computed in a preprocessing step and stored.

Algorithm 4.4 Backward transform.

Require: cluster basis $(V_\sigma)_{\sigma\in T(\mathcal{I})}$, set of vectors $\hat{u}$, index set $\mathcal{I}$.
  if $\mathcal{I}$ is a leaf node **then**
    $u\,|_{\mathcal{I}} \leftarrow V_{\mathcal{I}}\hat{u}_{\mathcal{I}}$
  else
    for σ **in** sons($\mathcal{I}$) **do**
      $u\,|_{\mathcal{I}} \leftarrow u\,|_{\mathcal{I}} + E_{\mathcal{I},\sigma}u\,|_{\tau}$
      backward transform $(V_\sigma)_{\sigma\in T(\mathcal{I})}$, $\hat{u}$, σ
    end for
  end if
  return u

Algorithm 4.5 $\mathcal{H}^2$-matrix-vector multiplication $u \leftarrow u + \alpha Aw$.

Require: $\alpha \in \mathbb{R}$, $A \in \mathbb{R}^{\mathcal{I}\times\mathcal{J}}$, $w \in \mathbb{R}^{\mathcal{J}}$, $u \in \mathbb{R}^{\mathcal{I}}$, $(V_\sigma)_{\sigma\in T(\mathcal{I})}, (U_\tau)_{\tau\in T(\mathcal{J})}$.
  Compute $\hat{w}$ by a forward transform from $(U_\tau)_{\tau\in T(\mathcal{J})}$, w, $\mathcal{J}$.
  for σ **in** $T(\mathcal{I})$ **do**
    $\hat{u}_\sigma \leftarrow 0$
  end for
  for $\sigma\times\tau$ **in** $T(\mathcal{I}\times\mathcal{J})$ admissible **do**
    $\hat{u}_\sigma \leftarrow \hat{u}_\sigma + \alpha\,\Sigma_{\sigma\times\tau}\,\hat{w}_\tau$
  end for
  $u \leftarrow 0$
  Compute u by a backward transform from $(V_\sigma)_{\sigma\in T(\mathcal{I})}$, $\hat{u}$, $\mathcal{I}$
  for $\sigma\times\tau$ **in** $T(\mathcal{I}\times\mathcal{J})$ *not* admissible **do**
    $u\,|_\sigma \leftarrow u\,|_\sigma + \alpha A\,|_{\sigma\times\tau}\; w\,|_\tau$
  end for

Afterward, they are used for simulations with the same geometry but possibly different boundary data or particle distributions. All BEM matrices are approximated by $\mathcal{H}^2$-matrices. For moderately sized problems, we also compute the inverse of the single-layer potential and approximate it by an $\mathcal{H}^2$-matrix. Although this leads to cubic complexity in the number of triangles, solving the linear system directly is faster than using an iterative method. For larger problems, this is not feasible anymore. We then apply a preconditioned conjugate gradient method; see [43] for preconditioning techniques in case of BEM matrices.

The computation of the discrete Dirichlet trace of the Newton potential $\underline{N}_0$ in (31) requires the L^2-projection onto the space of piecewise constant trial functions. This can be formulated as a matrix-vector multiplication. The computationally expensive matrix is efficiently approximated by an $\mathcal{H}^2$-matrix. Let us fix quadrature rules for all triangles of the surface mesh. Ideally all nodes lie on the edges of the triangles, therefore reducing the number of function evaluations as triangles sharing a common edge also share quadrature nodes and differ only in the weights. Typical choices are the midpoints of the edges of the triangles or the vertices of the triangles. We collect all quadrature nodes in a global set $(y_j)_{j\in\mathcal{Q}}$ and denote the positions of the particles by $(x_i)_{i=1}^{N_p}$. We can write

$$\underline{N}_0 \approx \frac{1}{\beta} M_{\mathcal{Q}}\Phi_{\mathcal{Q}}w_q, \tag{37}$$

where $w_q = (q_0|\Omega|/N_p)_{i=1}^{N_p}$ is the vector of weighted charges and

$$\Phi_{\mathcal{Q}}[j,i] = U_\varepsilon(y_j, x_i) = \frac{1}{4\pi}\frac{1}{|y_j - x_i|}, \quad j \in \mathcal{Q},\ i = 1, \dots, N_p,$$

are the evaluations of the fundamental solutions of the particles at the quadrature nodes and $M_{\mathcal{Q}}$ is a sparse matrix mapping the global nodes $(y_j)_{j\in\mathcal{Q}}$ to the corresponding triangles, multiplied with the quadrature weights.

The matrix $\Phi_{\mathcal{Q}}$ is a Nyström matrix whose approximation by $\mathcal{H}^2$-matrices is discussed in section 4. The evaluation of $\underline{N}_0$ is reduced to linear complexity in both the number of particles and the number of triangles. The Neumann trace $\underline{N}_1$ is now readily computed by the relation (28) in linear complexity given the matrices are approximated by $\mathcal{H}^2$-matrices.

For the computation of the electric field the gradient of the representation formula is evaluated at the positions of the particles. In order to efficiently apply the gradient, each evaluation of a component is reformulated as a matrix-vector product; see Algorithm 3.1. Due to the special structure of the fundamental solution (3), these products are computed simultaneously. Since

$$-\nabla_x U(x,y) = \frac{x-y}{4\pi|x-y|^3}, \quad x \neq y \in \mathbb{R}^3,$$

the entries of the matrices $(S_k)_{k=1}^3$, $(D_k)_{k=1}^3$, and $(F_k)_{k=1}^3$ from (32) and (33) have a common denominator. Additionally, its computation is the most expensive part of the algorithm. Therefore we compute the third power of the distance only once and use this result for the computation of all matrices. Again, these matrices fit into the general framework from section 4 and are efficiently approximated by $\mathcal{H}^2$-matrices reducing the complexity of the matrix-vector multiplication from quadratic to linear with respect to the number of particles. Summarizing, the quadratic Algorithm 3.1 is transformed to an algorithm with linear complexity by replacing all full matrices by their $\mathcal{H}^2$-approximations and by performing the matrix-vector multiplication as described in Algorithm 4.5. Transferred to a computer program, the subroutines for a dense matrix-vector multiplication simply have to be changed to their $\mathcal{H}^2$-matrix equivalents. In this sense, using $\mathcal{H}^2$-matrices accelerates the algorithm without changing important properties like the exact evaluation of the Coulomb force in the near field or the highly accurate evaluation of the gradient of the representation formula; see Lemma 3.2.

In our scheme, the aforementioned $\mathcal{H}^2$-matrices involving the positions of the particles are never fully built. Instead we exploit their hierarchical structure and compute the matrix-vector products on the fly. Iterating through the block cluster tree and accumulating the contribution of the admissible leafs, the computation of the full matrices $\Phi_{\mathcal{Q}}, (S_k)_{k=1}^3, (D_k)_{k=1}^3$, and $(F_k)_{k=1}^3$ are reduced to the computation of the small leaf matrices. Only storage for these small matrices is allocated which are freed after a matrix-vector multiplication with parts of the vector w_q. The positions of the particles change after each time step. It is therefore necessary to rebuild the cluster tree, block cluster trees, and the cluster basis. Although with a formal complexity of $\mathcal{O}(N_p \log N_p)$, the computational time is negligible compared to the computation of the BEM gradient; see the timings in section 6.

The computation of the electric field relies heavily on an efficient implementation of the hierarchical matrix format and tree-based data structures. We developed our

code based on the H2Lib.[1] Written in the programming language C, all basic data structures and higher level routines like matrix-vector and matrix-matrix multiplication or factorization algorithms are available, as well as a BEM module for the Laplace equation in three dimensions, which is used in the subsequent computations.

6. Numerical examples. In this section we present several numerical examples. We begin with benchmarking the evaluation of the electric field and conclude with physically motivated examples that demonstrate classical plasma phenomena.

6.1. Verification of linear complexity. We numerically validate the linear scaling of the computational time for the evaluation of the electric field at the positions of the particles. The computation is split into four parts:

1. building the cluster basis in $\mathcal{O}(N_p \log N_p)$,
2. computation of $\underline{N}_0$ according to (37) in linear complexity),
3. computation of the particle-particle force (see (33) in linear complexity), and
4. evaluating the gradient of the representation formula (21) in linear complexity.

For our tests, we triangulate the surface of the unit ball in $\mathbb{R}^3$ and uniformly distribute negatively charged particles inside the domain. Appropriate nondimensionalization is irrelevant for this test, so we set all masses, charges, and weights to unity. Homogeneous Dirichlet boundary conditions are chosen for the electric potential. We use $m = 5$ interpolation nodes at each spatial direction for the $\mathcal{H}^2$-matrix approximation. The minimal cluster leaf size $n_{\min}$ is $2m^3$ and the admissibility constant η is 2.

Figure 1 shows the relative computational times for a fixed mesh with varying number of particles. The relative magnitudes of the different steps during the computation of the electric field are given in Figure 2. Although formally being of complexity $\mathcal{O}(N_p \log N_p)$, we observe a linear scaling of the computation of the cluster basis. Furthermore, the absolute timings are in the order of 100 ms, making this part of the algorithm negligible compared to rest of the algorithm which takes in the order of seconds. The evaluation of $\underline{N}_0$ almost perfectly scales linearly with the number of particles. The evaluation of the gradient of the Newton potential and of the representation formula follows a linear trend. The constant hidden in the $\mathcal{O}$ notation of Lemma 4.2 depends on the form of the block cluster tree. As the particles are distributed randomly in the unit ball, we cannot expect to obtain the same shape constant for the block cluster tree for a large range of numbers of particles. Figure 3 shows that the computation of the gradient of the representation formula and of $\underline{N}_0$ scale linearly with the number of triangles.

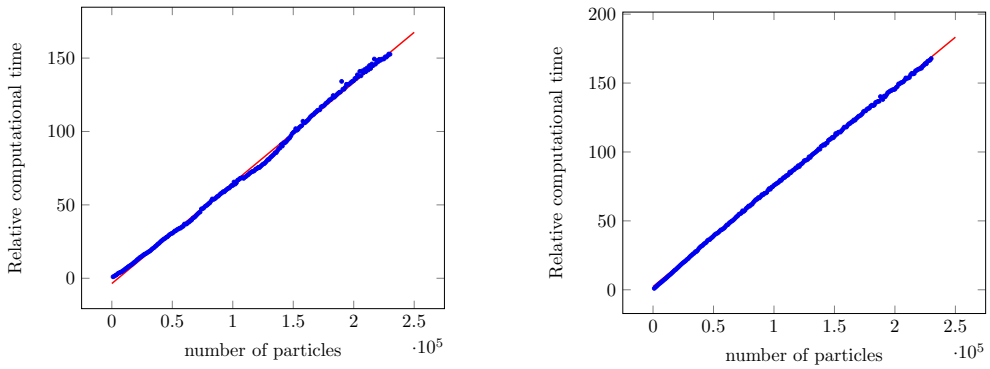
6.2. Physically motivated examples. For most applications the plasma contains positively and negatively charged particles. Usually, the positive charge exists of ionized atoms and electrons form the negatively charged part. Since the atoms are much heavier than the electrons they are modeled as immobile. This gives rise to a homogeneous positive background charge, such that the system is electrically neutral from the outside. The Poisson equation in (11) changes to

$$-\Delta_x \phi = \frac{1}{\beta}\left[1 - w \sum_{i=1}^{N_p} \delta^{\varepsilon}_{x_i}\right]$$

with boundary conditions

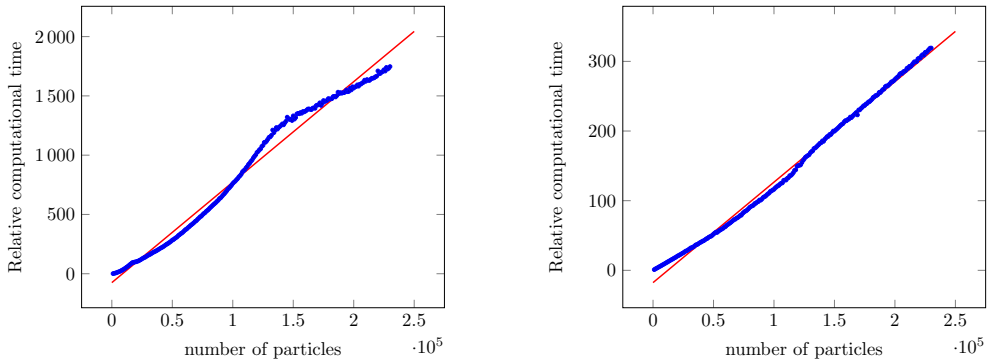
$$\begin{aligned} \phi &= g_D \quad \text{on } \Gamma_D, \\ n_\Omega \cdot \nabla \phi &= g_N \quad \text{on } \Gamma_N. \end{aligned}$$

[1] The source code and further information can be found at http://h2lib.org/.



(a) Rebuilding cluster basis

(b) Newton potential



(c) field of the particles

(d) representation formula

FIG. 1. *Single-core timings relative to* 1,000 *particles for building the cluster basis in* (a) *(*$2.7 \cdot 10^{-3}$ *s), the evaluation of* $\underline{N}_0$ *in* (b) *(*$2.3 \cdot 10^{-3}$ *s), the particle field in* (c) *(*$5.3 \cdot 10^{-3}$ *s), and the gradient of the representation formula in* (d) *(*$1.4 \cdot 10^{-1}$ *s). The number of triangles is* 1,280. *Computations were performed on an Intel Xeon Gold* 6154@3 *GHz with* `icc` 19. *Relevant compiler flags are* `-Ofast -xHost`.

Note that the integral of the right-hand side over Ω is zero, as $w = |\Omega|/N_p$. A particular solution for the homogeneous background charge is

$$\phi_b(x) = -\frac{1}{6\beta}|x|^2, \quad x \in \Omega.$$

By subtracting traces of the particular solution ϕ_b, we transform the boundary value problem to

$$\begin{aligned} -\Delta_x \phi_e &= \frac{1}{\beta} w \sum_{i=1}^{N_p} \delta_{x_i}, \\ \phi_e &= g_D - g_b && \text{on } \Gamma_D, \\ n_\Omega \cdot \nabla \phi_e &= g_N - n_\Omega \cdot \nabla \phi_b && \text{on } \Gamma_N. \end{aligned}$$

The electric field is now obtained by

$$E = -\nabla \phi_e - \nabla \phi_b.$$

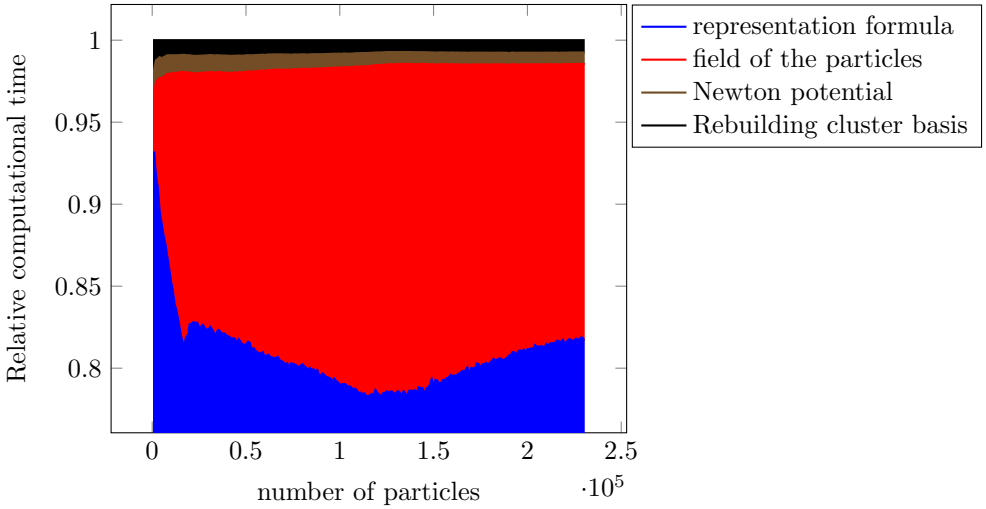


FIG. 2. *Cumulative computational times for* 1,280 *triangles and varying number of particles.*

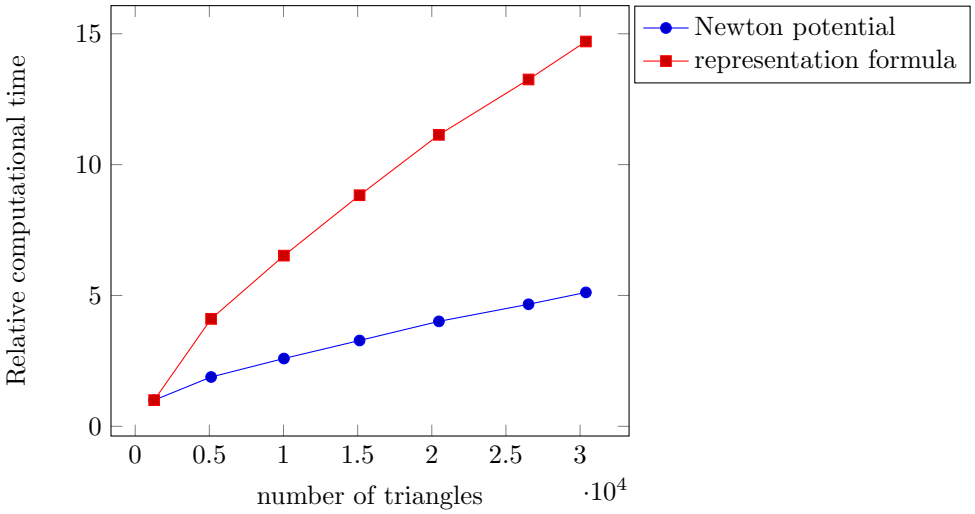


FIG. 3. *Single-core timings for different number of triangles relative to* 1,280 *triangles for* 10,000 *particles. The base values are* $1.9 \cdot 10^{-2}\,s$ *and* $1.4\,s$ *for the Newton potential and representation formula, respectively. For information on the CPU and the compiler, see the caption of Figure* 1.

As ϕ_b is independent of the geometry and the distribution of the particles, its evaluation and the evaluation of its gradient are grid-free, as well as the computation of ϕ_e. Computations with background charge can be found in subsections 6.2.2 and 6.2.3.

6.2.1. Accelerator. As a first example for nontrivial boundary conditions, we consider an accelerator geometry, meshed with 8,904 triangles. The physically relevant parameters are $L_0 = 0.1\,\mathrm{m}$, $n_0 = 10^{12}\,\mathrm{m}^{-3}$, and $k_B T_0 = 1\,\mathrm{eV}$. The profile of the rotationally symmetric accelerator and the boundary conditions for the electric potential are depicted in Figure 4. Initially, 10,000 particles are placed in the left cylinder with a bulk velocity of 10 in the positive x-direction and are absorbed at

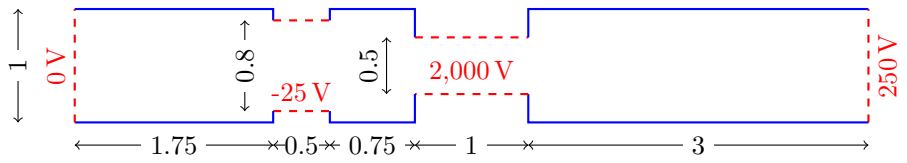


FIG. 4. *Profile of the accelerator along the x-axis. On the solid blue parts, homogeneous Neumann conditions are imposed. The voltages along the dashed red lines indicate the value of the Dirichlet boundary condition on these segments.*

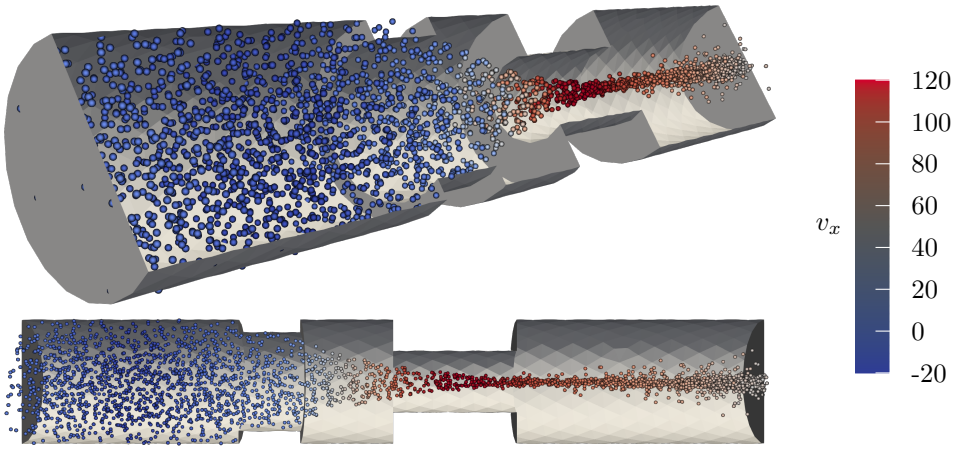


FIG. 5. *Final distribution of* 3,000 *particles inside the accelerator. The color indicates the velocity of the particles in the x-direction.*

the boundary. Once they pass the first narrow, called the screen, they are focused such that they pass the second narrow, the accelerator, without being absorbed by the boundaries. The distribution of 3,000 particles after 100 time steps with a time step size of 10^{-3} is shown in Figure 5.

6.2.2. Plasma oscillations. As a first example with a homogeneous background charge, we examine plasma oscillations. The geometry is a cylinder along the z-axis with radius 1 and height 5, centered in 0. It is discretized with 2,110 triangles. The characteristic quantities are $L_0 = 0.1\,\mathrm{m}$, $n_0 = 10^{12}\,\mathrm{m}^{-3}$, and $k_B T_0 = 1\,\mathrm{eV}$. 5,000 particles are distributed uniformly in a smaller cylinder of height 4 around the center of the geometry. Their initial velocities are set to 0. The boundary is absorbing; at the bases we set homogeneous Dirichlet conditions and homogeneous Neumann conditions on the rest. Physically, the latter boundary condition means that we impose a vanishing surface charge density; in particular there is no net charge on this part of the boundary. Mathematically, since the normal vectors point in a radial direction, the condition

$$0 = \gamma_1 \phi = n_\Omega \cdot \gamma_0 \nabla \phi = -n_\Omega \cdot \gamma_0 E$$

ensures that the field lines close to the boundary are parallel to the cylinder axis. To prevent the particles from being absorbed at the lateral surface of the cylinder, we add a constant magnetic field in the order of 10 mT along the z-axis. The acceleration due to the magnetic field is computed with the Boris scheme [6, 7] using a time step

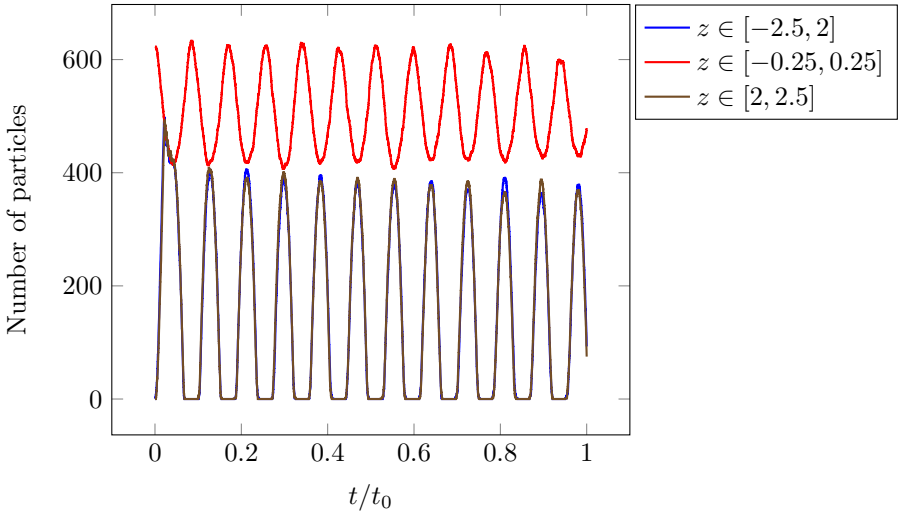


FIG. 6. *Number of particles in three parts of the cylinder over time for* $n = 10n_0$.

size of 10^{-4}. In an infinite system, the plasma oscillates with the plasma frequency

$$\omega_p = \sqrt{\frac{n_0 e^2}{\varepsilon_0 m_e}},$$

which depends only on the electron density. As we simulate the plasma in a bounded domain, we cannot expect the plasma to oscillate with the frequency ω_p. Instead, we validate that the frequency for the bounded domain is still a function of the square root of n_0. In order to do so, we vary the electron density n from n_0 to $100n_0$. Counting the number of particles in three parts of the cylinder, $z \in [-2.5, 2]$, $z \in [-0.25, 0.25]$, and $z \in [2, 2.5]$, at each time step, we extract the dominating nonzero frequency after with the help of the discrete Fourier transform. The numbers of particles in the left, the middle, and the right part of the cylinder for $n = 10n_0$ are shown in Figure 6. The distribution of the particles oscillates with dominating frequency of 12 in units of $1/t_0$, which corresponds to an angular frequency of

$$\omega_c = 3.2 \cdot 10^8 \, \frac{1}{\mathrm{s}}$$

in physical units. This is in the order of the plasma frequency

$$\omega_p = \sqrt{\frac{10 n_0 e^2}{\varepsilon_0 m_e}} \approx 1.8 \cdot 10^8 \, \frac{1}{\mathrm{s}}.$$

The spectra of the lines in Figure 6 differ only in magnitude, not in the positions of peaks. Therefore, we only show the spectrum of the second line of Figure 6 in Figure 7. Repeating this for several densities between n_0 and $100n_0$ yields Figure 8, from which the dependency of the frequency on the square root of the density is clearly deduced.

6.2.3. Plasma sheath. A classical nonlinear phenomenon in plasma physics is the formation of sheaths; see the classical textbook [14]. For this example, we set $L_0 = 0.1\,\mathrm{m}$, $n_0 = 10^{13}\,\mathrm{m}^{-3}$, and $k_B T_0 = 1\,\mathrm{eV}$. We uniformly distribute 10,000

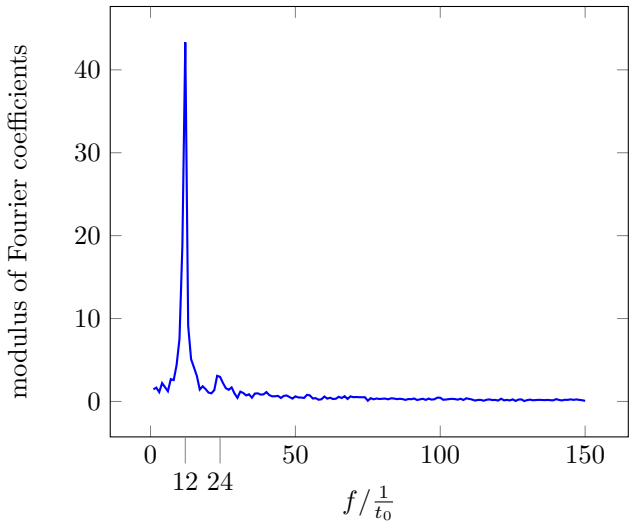


FIG. 7. *Fourier spectrum of the number of particles in the middle of the cylinder, the red line in Figure* 6. *The constant mode is excluded from the spectrum.*

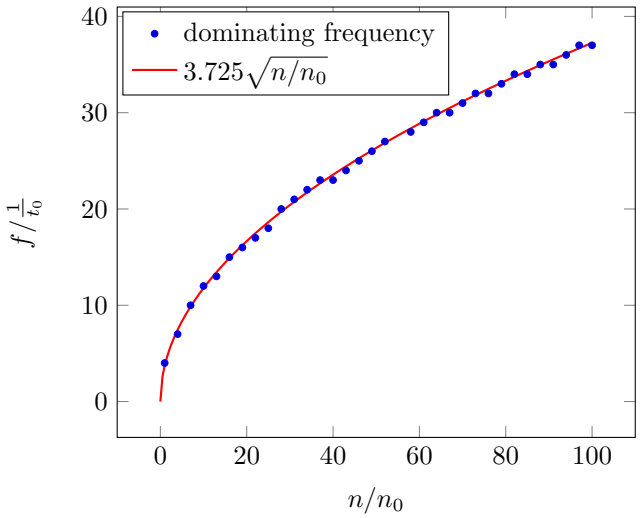


FIG. 8. *Frequency of the oscillation of the number of particles in the middle of the cylinder as a function of the electron density.*

particles with velocity following a Maxwellian distribution with temperature 1 and bulk velocity 0 within the unit sphere, which is discretized with 1,280 triangles. The particles are absorbed at the boundary; for the electric potential, we impose homogeneous Dirichlet boundary conditions. The system is evolved with a time step size of 10^{-3}. Figure 9 shows the number of particles within the unit sphere as a function of time. At the beginning, the fastest particles leave the sphere, giving rise to a positive charge at the boundary. With the growing potential barrier, the particles are excluded from a thin area near the boundary, the so-called sheath, and are confined inside the sphere. Figure 10 includes the final radial distribution function of the particles inside

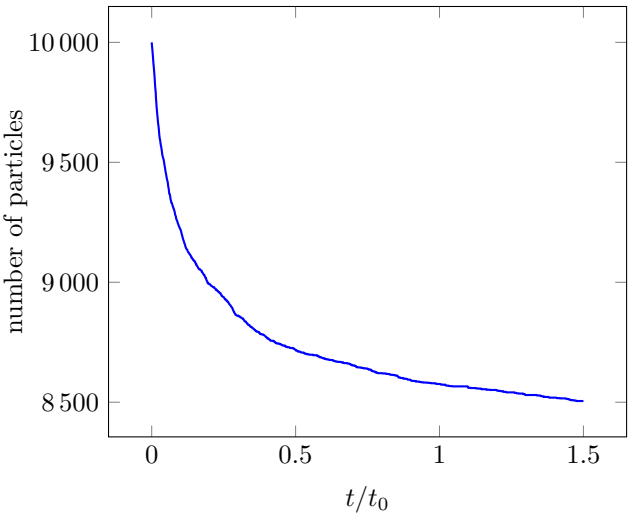


FIG. 9. *Number of particles inside the sphere over time.*

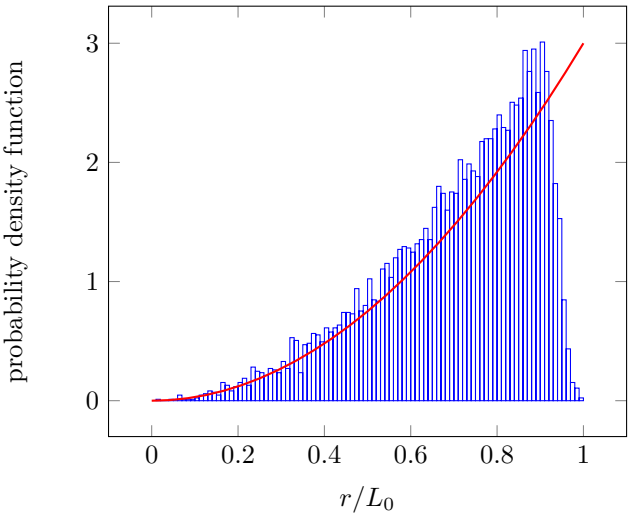


FIG. 10. *Radial histogram of the final particle distribution inside the sphere. The solid red line shows the probability density function of the uniform distribution.*

the sphere and the analytical radial distribution function of a uniformly distributed random variate inside the unit sphere. While the final positions are still uniformly distributed up to a radius of approximately 0.6, the distribution strongly deviates from the uniform distribution especially close to radii of 1, where it suddenly drops to 0.

6.3. Summary. To summarize, the numerical examples show that we are able to simulate important nonlinear plasma phenomena like plasma oscillations or the formation of sheaths. The results also match available theoretical predictions. Furthermore, the numerical study demonstrates the linear complexity of our method and its applicability to three-dimensional domains with mixed boundary values. The effi-

ciency and flexibility of our approach open the possibilities for future simulations of complex problems in different plasma regimes.

REFERENCES

[1] A. W. Appel, *An efficient program for many-body simulation*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 85–103.

[2] J. Barnes and P. Hut, *A hierarchical $o(n \log n)$ force-calculation algorithm*, Nature, 324 (1986), pp. 446–449.

[3] M. Bebendorf, *Approximation of boundary element matrices*, Numer. Math., 86 (2000), pp. 565–589.

[4] M. Bebendorf, *Hierarchical Matrices*, Lecture Notes in Comput. Sci. Eng. 63, Springer, Berlin, 2008.

[5] M. Bebendorf and S. Rjasanow, *Adaptive low-rank approximation of collocation matrices*, Computing, 70 (2003), pp. 1–24.

[6] C. Birdsall and A. B Langdon, *Plasma Physics via Computer Simulation*, Ser. Plasma Phys., Taylor & Francis, New York, 2005.

[7] J. Boris, *The Acceleration Calculation from a Scalar Potential*, Technical report MATT-152, Plasma Physics Laboratory, Princeton University, Princeton, NJ, 1970.

[8] S. Börm, *Efficient Numerical Methods for Non-local Operators*, EMS Tracts Math. 14, European Mathematical Society, Zurich, 2010.

[9] S. Börm and S. Christophersen, *Approximation of integral operators by Green quadrature and nested cross approximation*, Numer. Math., 133 (2016), pp. 409–442.

[10] S. Börm and L. Grasedyck, *Low-rank approximation of integral operators by interpolation*, Computing, 72 (2004), pp. 325–332.

[11] S. Börm and L. Grasedyck, *Hybrid cross approximation of integral operators*, Numer. Math., 101 (2005).

[12] S. Börm, M. Löhndorf, and J. M. Melenk, *Approximation of integral operators by variable-order interpolation*, Numer. Math., 99 (2005), pp. 605–643.

[13] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Stat. Comput., 9 (1988), pp. 669–686.

[14] F. F. Chen, *Introduction to Plasma Physics and Controlled Fusion*, Springer, Berlin, 2016.

[15] H. Cheng, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm in three dimensions*, J. Comput. Phys., 155 (1999), pp. 468–498.

[16] A. Christlieb, R. Krasny, J. Verboncoeur, J. Emhoff, and I. Boyd, *Grid-free plasma simulation techniques*, IEEE Trans. Plasma Sci., 34 (2006), pp. 149–165.

[17] A. J. Christlieb and K. Cartwright, *Boundary integral corrected particle-in-cell*, in Proceedings of the IEEE 35th International Conference on Plasma Science, 2008.

[18] A. J. Christlieb, R. Krasny, and J. Verboncoeur, *Efficient particle simulation of a virtual cathode using a grid-free treecode Poisson solver*, IEEE Trans. Plasma Science, 32 (2004), pp. 384–389.

[19] A. J. Christlieb, R. Krasny, and J. P. Verboncoeur, *A treecode algorithm for simulating electron dynamics in a Penning-Malmberg trap*, Comput. Phys. Commun., 164 (2004), pp. 306–310.

[20] T. Darden, D. York, and L. Pedersen, *Particle mesh Ewald: An N log(N) method for Ewald sums in large systems*, J. Chem. Phys., 98 (1993), pp. 10089–10092.

[21] J. M. Dawson, *Particle simulation of plasmas*, Rev. Modern Phys., 55 (1983), pp. 403–447.

[22] K. Ganguly and H. D. Victory, Jr., *On the convergence of particle methods for multidimensional Vlasov-Poisson systems*, SIAM J. Numer. Anal., 26 (1989), pp. 249–288.

[23] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[24] L. Greengard and V. Rokhlin, *The rapid evaluation of potential fields in three dimensions*, in Vortex Methods, C. Anderson and C. Greengard, eds., Lecture Notes in Math. 1360, Springer, Berlin, 1988, pp. 121–141.

[25] L. Greengard and V. Rokhlin, *A new version of the fast multipole method for the Laplace equation in three dimensions*, Acta Numer., 6 (1997), pp. 229–269.

[26] W. Hackbusch, *Hierarchical Matrices: Algorithms and Analysis*, Springer Ser. Comput. Math. 49, Springer, Berlin, 2015.

[27] R. Hockney and J. Eastwood, *Computer Simulation Using Particles*, CRC Press, Boca Raton, FL, 1988.

[28] V. Kolobov and R. Arslanbekov, *Electrostatic PIC with adaptive Cartesian mesh*, J. Phys. Conf. Ser., 719 (2016), 012020.
[29] D. Lazarovici and P. Pickl, *A mean field limit for the Vlasov-Poisson system*, Arch. Ration. Mech. Anal., 225 (2017), pp. 1201–1231.
[30] W. C. H. McLean, *Strongly Elliptic Systems and Boundary Integral Equations*, Cambridge University Press, Cambridge, UK, 2000.
[31] H. Neunzert, *An introduction to the nonlinear Boltzmann-Vlasov equation*, in Kinetic Theories and the Boltzmann Equation (Montecatini, 1981), Lecture Notes in Math. 1048, Springer, Berlin, 1984, pp. 60–110.
[32] H. Neunzert and J. Wick, *Theoretische und numerische Ergebnisse zur nichtlinearen Vlasov-Gleichung*, in Numerische Lösung nichtlinearer partieller Differential- und Integrodifferentialgleichungen (Tagung Math. Forschungsinst., Oberwolfach, 1971), Lecture Notes in Math. 267, Springer, Berlin, 1972, pp. 159–185.
[33] H. Neunzert and J. Wick, *Die Theorie der asymptotischen Verteilung und die numerische Lösung von Integrodifferentialgleichungen*, Numer. Math., 21 (1973/74), pp. 234–243.
[34] H. Neunzert and J. Wick, *Die Approximation der Lösung von Integro-Differentialgleichungen durch endliche Punktmengen*, in Numerische Behandlung nichtlinearer Integrodifferential- und Differentialgleichungen (Tagung, Math. Forschungsinst., Oberwolfach, 1973), Lecture Notes in Math. 395, Springer, Berlin, 1974, pp. 275–290.
[35] D. R. Nicholson, *Introduction to Plasma Theory*, Wiley, New York, 1983.
[36] C. Nieter, J. R. Cary, G. R. Werner, D. N. Smithe, and P. H. Stoltz, *Application of Dey–Mittra conformal boundary algorithm to 3D electromagnetic modeling*, J. Comput. Phys., 228 (2009), pp. 7902–7916.
[37] G. Of, O. Steinbach, and P. Urthaler, *Fast evaluation of volume potentials in boundary element methods*, SIAM J. Sci. Comput., 32 (2010), pp. 585–602.
[38] G. Of, O. Steinbach, and W. L. Wendland, *Applications of a fast multipole Galerkin in boundary element method in linear elastostatics*, Comput. Vis. Sci., 8 (2005), pp. 201–209.
[39] G. Of, O. Steinbach, and W. L. Wendland, *The fast multipole method for the symmetric boundary integral formulation*, IMA J. Numer. Anal., 26 (2006), pp. 272–296.
[40] S. Rjasanow and O. Steinbach, *The Fast Solution of Boundary Integral Equations*, Math. Anal. Tech. Appl. Eng., Springer, Berlin, 2007.
[41] S. Sauter and C. Schwab, *Boundary Element Methods*, Springer Ser. Comput. Math. 39, Springer, Berlin, 2011.
[42] H. Spohn, *Large Scale Dynamics of Interacting Particles*, Springer, Berlin, 1991.
[43] O. Steinbach, *Numerical Approximation Methods for Elliptic Boundary Value Problems: Finite and Boundary Elements*, Springer, Berlin, 2007.
[44] J. P. Verboncoeur, *Particle simulation of plasmas: Review and advances*, Plasma Phys. Controlled Fusion, 47 (2005), pp. A231–A260.
[45] S. Wollman, *On the approximation of the Vlasov-Poisson system by particle methods*, SIAM J. Numer. Anal., 37 (2000), pp. 1369–1398.
[46] L. Ying, G. Biros, and D. Zorin, *A kernel-independent adaptive fast multipole algorithm in two and three dimensions*, J. Comput. Phys., 196 (2004), pp. 591–626.