SIAM J. SCI. COMPUT.
Vol. 43, No. 3, pp. C235–C257

PIFE-PIC: PARALLEL IMMERSED FINITE ELEMENT PARTICLE-IN-CELL FOR 3-D KINETIC SIMULATIONS OF PLASMA-MATERIAL INTERACTIONS*

DAORU HAN†, XIAOMING HE‡, DAVID LUND†, AND XU ZHANG§

Abstract. This paper presents a recently developed particle simulation code package PIFE-PIC, which is a novel three-dimensional (3-D) parallel immersed finite element (IFE) particle-in-cell (PIC) simulation model for particle simulations of plasma-material interactions. This framework is based on the recently developed nonhomogeneous electrostatic IFE-PIC algorithm, which is designed to handle complex plasma-material interface conditions associated with irregular geometries using a Cartesian mesh-based PIC. Three-dimensional domain decomposition is utilized for both the electrostatic field solver with IFE and the particle operations in PIC to distribute the computation among multiple processors. A simulation of the orbital motion-limited (OML) sheath of a dielectric sphere immersed in a stationary plasma is carried out to validate parallel IFE-PIC and profile the parallel performance of the code package. Furthermore, a large-scale simulation of plasma charging at a lunar crater containing 2 million PIC cells (10 million FE/IFE cells) and about 1 billion particles, running for 20,000 PIC steps in about 154 wall-clock hours, is presented to demonstrate the high-performance computing capability of PIFE-PIC.

Key words. immersed finite element, particle-in-cell, parallel domain decomposition, plasma-material interactions

AMS subject classifications. 35R05, 65N30, 65Y05

DOI. 10.1137/20M137344X

1. Introduction.

Particle modeling of plasma dynamics has emerged as one of the most appropriate algorithms for first-principle-based modeling of many plasma-material interaction (PMI) problems. One of the fundamental phenomena in plasma-material interactions is surface charging. When an object is immersed in a plasma, its surface will collect charge from the plasma until it reaches an equilibrium surface potential determined by the current balance condition. Many plasma-material interaction problems involve multiple objects with complex geometries; therefore the interface conditions between the plasma and object need to be accurately resolved.

Being one of the most popular kinetic methods for collisionless plasma simulations, the particle-in-cell (PIC) method [11] models the charged particles as macroparticles and tracks the motions of particles in the electrostatic/electromagnetic field. The electric potential in a PIC simulation domain is governed by the second-order elliptic partial differential equations (PDEs) with discontinuous dielectric coefficients and

*Submitted to the journal's Software and High-Performance Computing section October 14, 2020; accepted for publication (in revised form) February 22, 2021; published electronically June 14, 2021. Part of this work was previously presented as a conference paper in 2018 AIAA Aerospace Sciences Meeting (paper number AIAA 2018-2196), Kissimmee, FL, 2018. The work presented here has since been signicantly revised and extended.

https://doi.org/10.1137/20M137344X

Funding: This work was partially supported by NASA Missouri Space Grant Consortium through NASA-EPSCoR-Missouri and by the NSF through grants DMS-2005272 and OAC-1919789. The third author was also supported in part by NASA-Missouri Space Grant Consortium Scholarships.

†Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology, Rolla, MO 65409 USA (handao@mst.edu, dclgzb@mst.edu).

‡Department of Mathematics and Statistics, Missouri University of Science and Technology, Rolla, MO 65409 USA (hex@mst.edu).

§Department of Mathematics, Oklahoma State University, Stillwater, OK 74078 USA (xzhang@okstate.edu).

nonhomogeneous flux jumps across the material surface interface. Numerical methods based on structured meshes, especially Cartesian meshes, are particularly desirable in these simulations because they enable efficient particle tracking and save computing time in PMIs.

The immersed finite element (IFE) method is a finite element method (FEM) for solving interface problems using interface-independent meshes such as Cartesian meshes [22, 24, 58, 60, 71, 77]. The main idea of IFE is to adjust approximating functions locally to accommodate the physical interface conditions [12, 25, 41, 42, 57, 59, 78]. An IFE method can achieve optimal convergence on an interface-independent mesh with the number and location of the degrees-of-freedom isomorphic to the standard FEM on the same mesh [28, 29, 46, 61, 87]. The first IFE method was introduced by Li in [58] for solving one-dimensional (1-D) elliptic interface problems with piecewise linear polynomials. Since then, the IFE method has been extended to higher-order approximations [2, 12, 14, 26, 31], various discretization techniques [1, 4, 21, 43, 44, 47, 64], higher-dimensional elliptic interface problems [27, 30, 40, 63, 77], and other interface PDE models [3, 7, 48, 62, 85, 86].

Over the past decade, the IFE method has been successfully used together with PIC in plasma particle simulations [5, 6, 16, 17, 18, 80]. Recently, a nonhomogeneous IFE-PIC algorithm has been developed for particle simulations of PMIs with complex geometries while maintaining the computational speed of the Cartesian mesh-based PIC [15, 39, 45, 65, 66]. To the best of our knowledge, most existing IFE-PIC algorithms are serial. The nonparallel algorithms have limitations in their capability to handle large-scale particle simulations and their efficiency in using multiple processors at the algorithm level. For a typical large-scale 3-D PIC simulation, millions to billions of particles are tracked in the computation domain that contains millions of elements. With the availability of multiprocessor computational facilities, the call for parallel IFE-PIC algorithms is urgent.

The goal of this paper is to develop and test a new parallel IFE-PIC package for particle simulations of electrostatic PMIs, namely, PIFE-PIC. We utilize a 3-D domain decomposition technique for both *field-solve* and *particle-push* procedures of the PIC model. The computations are distributed into multiple subdomains which can be handled independently by multiple processors. The key is how to efficiently exchange the information between these subdomains. In this work, neighboring subdomains have a small overlapping ("guard cells") region which will be used as a common region to interchange the PDE solutions and the particle data. Extensive numerical experiments show that our PIFE-PIC scheme significantly outperforms the serial IFE-PIC scheme. Although it maintains a similar accuracy as the serial IFE-PIC computational scheme, the high parallel performance dramatically reduces the computational time for problems of practical interests. Hence, large-scale kinetic simulations of PMIs can be carried out much more efficiently.

The rest of this paper is organized as follows. In section 2, we describe the details of 3-D domain decomposition for both IFE (field-solve) and PIC (particle-push) procedures of PIFE-PIC. In section 3, we present a code validation using a 3-D sheath problem of a dielectric sphere immersed in a stationary plasma. Section 4 presents a parallel efficiency test of the PIFE-PIC code for strong scaling. Section 5 presents an application of PIFE-PIC to simulations of lunar surface charging at a crater. Finally, a summary and conclusion are given in section 6.

2. PIFE-PIC algorithms.

2.1. Overview of PIC and IFE-PIC. PIC is a widely-used kinetic particle simulation method for plasma dynamics [11, 50]. In PIC, charged particles of

plasma species are represented by a number of simulation particles (also referred to as macroparticles or superparticles) distributed "freely" in the entire computation domain, while the field quantities such as electric potential are discretized on a mesh (thus the name "particle-in-cell"). The kernel of PIC method is the "PIC loop" which includes four essential steps: scatter, field-solve, gather, and particle-push (Figure 2.1). Within a PIC loop, quantities carried by the simulation particles are weighted onto the mesh nodes ("scatter") to form the right-hand side (RHS) term of the PDE for the solution of the electrostatic/electromagnetic field ("field-solve"), which is in turn interpolated at particle positions ("gather") to update the velocity and position of the particles ("particle-push"). Such data exchange between particles and field quantities will iterate for a desired number of steps (or till a convergence criterion is met) to obtain the self-consistent solution of both particles and fields.

For problems of PMIs, the mathematical model is an interface problem including the electrostatic/electromagnetic field problem in a self-consistent solution to the corresponding plasma dynamics problem (Figure 2.2(a)), together with the appropriate interface conditions between the plasma region and the material region (Figure 2.2(b)). For electrostatic problems presented in this work, we consider the following boundary value problem of the elliptic equation that governs the distribution of the electric potential ϕ [51]:

$$-\nabla \cdot \big(\varepsilon \nabla \phi(X)\big) = \rho(X), \quad X = (x, y, z) \in \Omega^- \cup \Omega^+, \tag{2.1}$$

$$\phi(X) = g(X), \quad X \in \Gamma_D, \tag{2.2}$$

$$\frac{\partial \phi(X)}{\partial \mathbf{n}_{\Gamma_N}} = p(X), \quad X \in \Gamma_N. \tag{2.3}$$

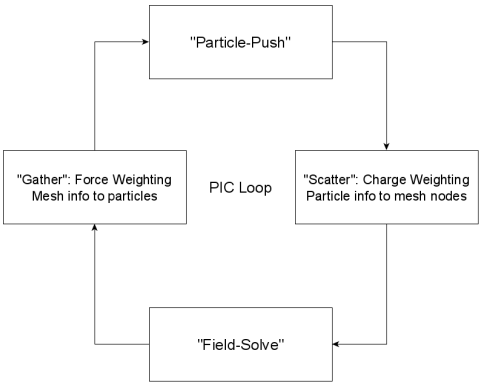
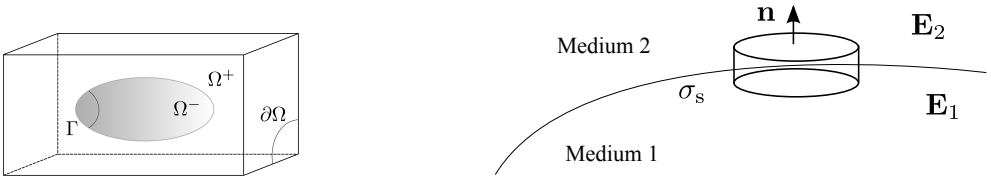


FIG. 2.1. *Four essential steps in a PIC loop.*



(a) Computation domain including the interface

(b) Electric flux jump across the interface

FIG. 2.2. *A sketch of the interface problem and interface condition.*

Here, $\Omega \in \mathbb{R}^3$ is assumed to be an open cuboidal domain, which is divided into two subdomains Ω^+ and Ω^- by an interface surface Γ such that $\overline{\Omega} = \overline{\Omega^- \cup \Omega^+ \cup \Gamma}$. The boundary $\partial\Omega$ consists of Dirichlet and Neumann portions, denoted by Γ_D and Γ_N, respectively, such that $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. The vector $\mathbf{n}_{\Gamma_N}$ is the unit outward normal of Γ_N. See the sketch in Figure 2.2(a). The functions ρ, g, and p are the source term, Dirichlet boundary function, and Neumann boundary function, respectively. The electric field $\mathbf{E} = -\nabla\phi(X)$ is discontinuous across the interface Γ with the following jump conditions imposed:

$$[\phi(X)]_\Gamma = 0, \tag{2.4}$$

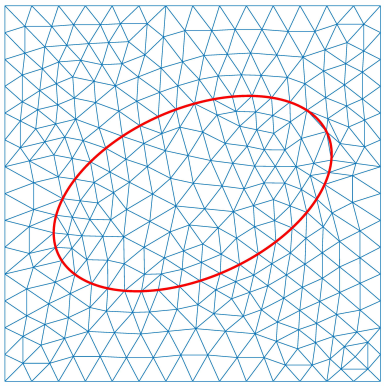
$$\left[\varepsilon \frac{\partial \phi(X)}{\partial \mathbf{n}_\Gamma}\right]_\Gamma = q(X), \tag{2.5}$$

where the jump $[\cdot]_\Gamma$ is defined by $[w(X)]_\Gamma := w^+(X)|_\Gamma - w^-(X)|_\Gamma$. The vector $\mathbf{n}_\Gamma$ is the unit normal of Γ pointing from Ω^- to Ω^+. The material-dependent coefficient $\varepsilon(X)$ is discontinuous across the interface. Without loss of generality, we assume it is a piecewise constant function defined by

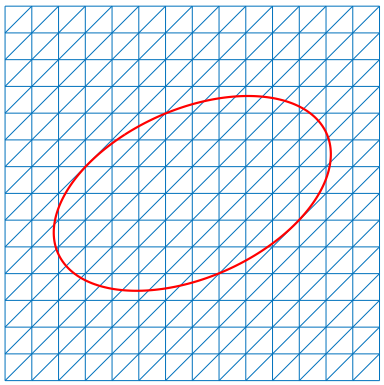
$$\varepsilon(X) = \begin{cases} \varepsilon^-, & X \in \Omega^-, \\ \varepsilon^+, & X \in \Omega^+, \end{cases}$$

where $\min(\varepsilon^+, \varepsilon^-) > 0$.

In many applications of scientific and engineering interest, the shape of the interface Γ is usually nontrivial. Traditionally, when solving field problems involving complex-shaped objects, an unstructured body-fitting mesh is employed to improve accuracy (Figure 2.3(a)). However, a structured mesh, such as Cartesian mesh (Figure 2.3(b)), is more advantageous in kinetic PIC modeling of plasma dynamics from the perspective of computing speed and efficiency, although, it has been limited to problems with relatively simple geometries due to accuracy considerations inherited from finite-difference-based schemes. To solve this dilemma while taking into account both accuracy and efficiency, the IFE-PIC method was developed to handle complex



(a) Unstructured body-fitting FE mesh



(b) Structured IFE mesh based on Cartesian mesh

FIG. 2.3. *Illustration of traditional body-fitting FE mesh and nonbody-fitting structured IFE mesh.*

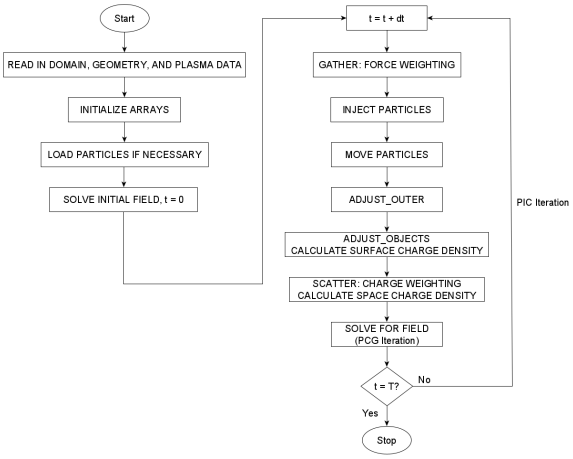


FIG. 2.4. *Flowchart of serial IFE-PIC.*

interface conditions associated with irregular geometries while maintaining the computational speed of the Cartesian mesh-based PIC. The detailed IFE formulation and IFE-PIC steps are archived in [39], and the flowchart of the serial IFE-PIC algorithm is shown in Figure 2.4. Over the past few years, the IFE-PIC method has matured to successfully model plasma dynamics problems arising from many space applications, such as ion thruster grid optics [13, 52, 54, 55], ion propulsion plume-induced contamination [56, 79, 76], charging of lunar and asteroidal surfaces [20, 35, 36, 37, 38, 83], and dust transport dynamics around small asteroids [84].

2.2. 3-D domain decomposition in PIFE-PIC. In our PIFE-PIC algorithm, the 3-D computational domain is decomposed along each dimension using the message passing interface (MPI) architecture (Figure 2.5). The domain is first decomposed into cuboid blocks with the same PIC mesh resolution. Each subdomain is handled by a processor for both field-solve and particle-push procedures of the PIC method. Two overlapping PIC cells ("guard cells") in each dimension are used in PIFE-PIC (Figure 2.6) taking advantage of the existing data structure of the serial IFE-PIC which has one layer of "guard cells" for global particle boundary conditions. Therefore, the boundaries of each subdomain are either on the global boundary or in the interior of its neighboring subdomains. Local IFE mesh is then generated for each subdomain. By virtue of the IFE formulation, PIC and IFE can use different mesh resolutions. In PIFE-PIC, PIC mesh is globally uniform to better balance the loads of particles among subdomains (processors). However, IFE mesh could be globally nonuniform but still locally uniform within each subdomain. The data interaction between IFE and PIC meshes of different resolutions is described in details in [56]. Figures 2.6 and 2.7 illustrate the 2-D and 3-D views of the domain decomposition and different resolutions.

2.3. Parallel algorithm for IFE field solver. For the parallel electrostatic field solver, Dirichlet-Dirichlet domain decomposition with overlapping cells is used to distribute the subdomains among multiple MPI processes [8]. For each subdomain, the IFE solver is the same as the sequential IFE method with Dirichlet boundary conditions [39, 53]. These Dirichlet boundary conditions are imposed at the boundaries of the subdomain, which are interior for the neighboring subdomains

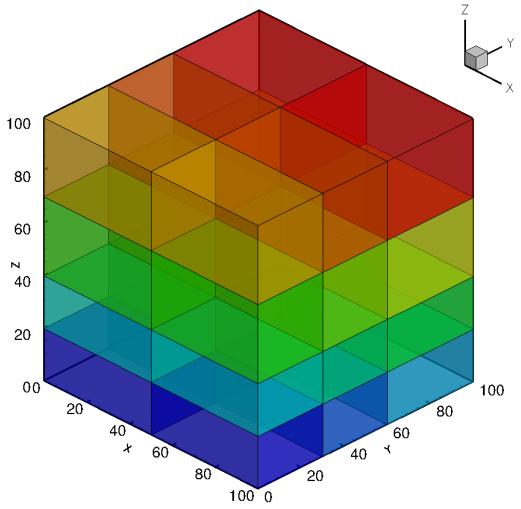
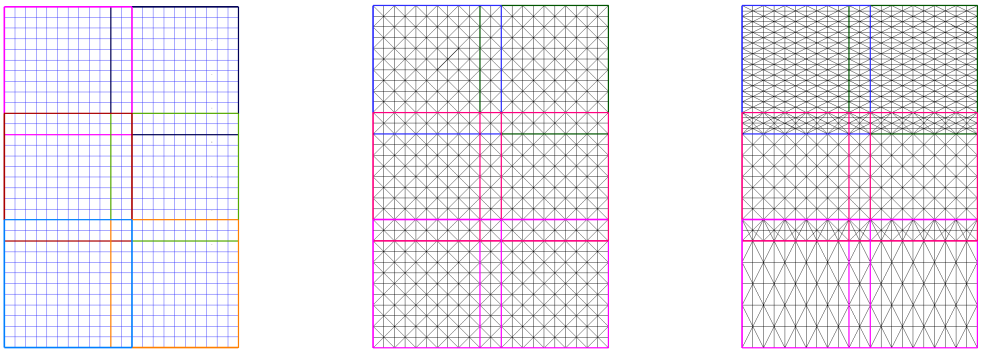


FIG. 2.5. *3-D domain decomposition for PIC blocks. Overlapping cells are not displayed. In this example, the global domain is decomposed into 2×3×4 subdomains. The blue-red color scale indicates the MPI rank of each subdomain.*



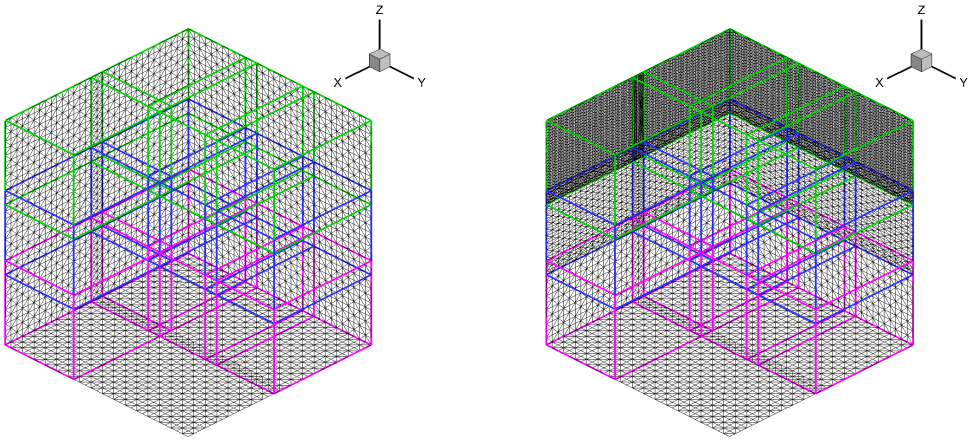
(a) Subdomains with two overlapping PIC cells in each dimension, globally uniform PIC mesh.

(b) Subdomains with two overlapping PIC cells in each dimension, globally uniform IFE mesh.

(c) Subdomains with two overlapping PIC cells in each dimension, globally nonuniform IFE mesh.

FIG. 2.6. *2-D projection showing the domain decomposition for PIC and IFE with overlapping cells and different resolutions. The thick colored edges represent boundaries of each subdomain, including one layer of guard cells in each dimension. Therefore, there are two overlapping layers of PIC cells in each dimension.*

(Figure 2.8, left). Therefore, the field solutions at respective neighboring subdomains are used as Dirichlet boundary conditions for each subdomain. At each field-solve step of PIC, domain decomposition method (DDM) iterations among subdomains are performed such that the solutions of the overlapping cells are exchanged and updated as the new Dirichlet boundary conditions for the respective neighboring subdomains. We denote this level of iteration as the DDM iteration. The relative error e_{rel} of DDM is defined with the L^2 norm as below:



(a) A globally uniform IFE mesh. (b) A globally nonuniform IFE mesh.

FIG. 2.7. *3-D view of globally uniform and nonuniform IFE meshes. The IFE mesh for each subdomain is uniform (locally) but could be nonuniform for different subdomains (globally).*

$$e_{\text{rel}} = \frac{\|\phi_{\text{new}} - \phi_{\text{old}}\|_{L^2}}{\|\phi_{\text{old}}\|_{L^2}}, \tag{2.6}$$

where ϕ_{new} and ϕ_{old} denote solutions at the new and old steps in the DDM iteration, respectively. Within the field-solve part at each PIC step, DDM iterations are carried out till the relative error reaches a preset tolerance or reaches the preset maximum number of DDM iterations. It is noted here that since PIFE-PIC uses domain decomposition in all 3 dimensions, which means there will be surfaces (side by side), edges (2-D diagonal), and vertices (3-D diagonal) shared by two neighboring subdomains, such MPI data exchange will be carried out at guard cell nodes on "surfaces" (+/− neighbor in one direction, such as Rank 1 and Rank 2 in Figure 2.8), "edges" (+/− neighbor in two directions, such as Rank 3 and Rank 6 in Figure 2.8), and "vertices" (+/− neighbor in three directions).

2.4. Parallel scheme for PIC procedures. In PIFE-PIC, simulation particles belonging to the same subdomain are stored together on the processor that solves the electrostatic field of the same subdomain (Figure 2.8, right). In this sense, "particle quantities" and "field quantities" of each subdomain are handled by the same processor. Each processor (MPI rank) handles its own particles belonging to its domain without guard cells (see Figure 2.5). In particle-push, particles crossing the inner boundaries are sent to the corresponding rank based on their destination positions. Note that such particle motion includes similar cases as data exchange for field-solve, which are "crossing one surface" (+/− neighbor in one direction, such as Rank 1 and Rank 2 in Figure 2.8), "crossing an edge (two surfaces)" (+/− neighbor in two directions, such as Rank 3 and Rank 6 in Figure 2.8), and "crossing a vertex (three surfaces)" (+/− neighbor in three directions).

2.5. Flowchart for PIFE-PIC. Figure 2.9 shows the flowchart of PIFE-PIC. The steps in red are major steps involving MPI operations associated with domain decomposition. In total, there are three levels of iteration in PIFE-PIC. The first level is the matrix-solving iteration which uses the preconditioned conjugate gradient (PCG) algorithm (PCG level). The second one checks the relative error in the

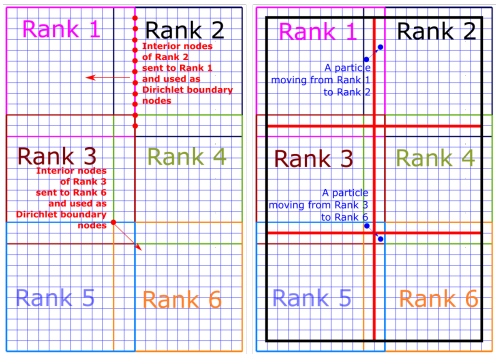


FIG. 2.8. *MPI data exchange among neighboring subdomains within DDM iteration. In both subfigures, the boundaries of the subdomains (with guard cells) of Ranks* 1 *to* 6 *are highlighted by different colors, which are the same as the colors of the text "Rank" in the corresponding subdomains. In the right subfigure, the thick red and black lines, which are the boundaries of the subdomains without guard cells, represent the inner and outer particle boundaries, respectively. Left: for field-solve operations: on the interior boundaries of the subdomains with guard cells, the nodes at a certain subdomain's boundary (e.g., Rank* 1*'s boundary nodes) are also interior nodes of its neighboring subdomain (e.g., Rank* 2*). Therefore, the field quantities stored on interior nodes of Rank* 2 *are sent to Rank* 1 *and used as Dirichlet boundary nodes. Since PIFE-PIC has* 3*-D domain decomposition, such MPI data exchange will be carried out at guard cell nodes on "surfaces" (+/− neighbor in one direction, such as Rank* 1 *and Rank* 2*), "edges" (+/− neighbor in two directions, such as Rank* 3 *and Rank* 6*), and "vertices" (+/− neighbor in three directions, not shown on this* 2*-D illustration). As for the nodes on the outer boundaries, which are the boundary of the entire problem domain, they will be handled based on the given boundary conditions of the entire problem domain, not by the MPI data exchange. Right: For particle-push operations: each processor handles its own particles belonging to its own subdomain without guard cells (see Figure* 2.5*). In particle-push, particles crossing the inner particle boundaries are sent to the corresponding rank based on their destination positions. Such particle motion also includes similar cases as data exchange for field-solve, which are "crossing one surface" (+/− neighbor in one direction, such as Rank* 1 *and Rank* 2*), "crossing an edge (two surfaces)" (+/− neighbor in two directions, such as Rank* 3 *and Rank* 6*), and "crossing a vertex (three surfaces)" (+/− neighbor in three directions, not shown on this* 2*-D illustration). For charge-weighting on the inner particle boundary, contributions from all neighboring subdomains without guard cells should be summed. As for the particles crossing the outer particle boundaries, they will be handled based on the given particle boundary conditions of the entire problem domain, not by the MPI data exchange.*

iterations of the domain decomposition method (DDM level). The third one tracks the solution of each PIC step (PIC level).

3. Code validation. We apply the PIFE-PIC code to simulate the charging of a small dielectric sphere immersed in a collisionless and stationary plasma in the OML sheath regime. Successful validations of the serial IFE-PIC against analytic OML solutions are presented in earlier work [39, 37].

3.1. Problem description and simulation setup. We consider a stationary, collisionless hydrogen plasma of equal ion and electron temperatures ($T_i = T_e$). The analytic expressions for ion and electron densities in the plasma are given by the revised OML theory [75, 19]. Therefore, the analytic potential profile near the sphere can be numerically solved from Poisson's equation in spherical coordinates.

3.1.1. Computation domain and mesh. In the simulation, we use a computation domain of a 5×5×5 Debye cube with a globally uniform PIC mesh with the

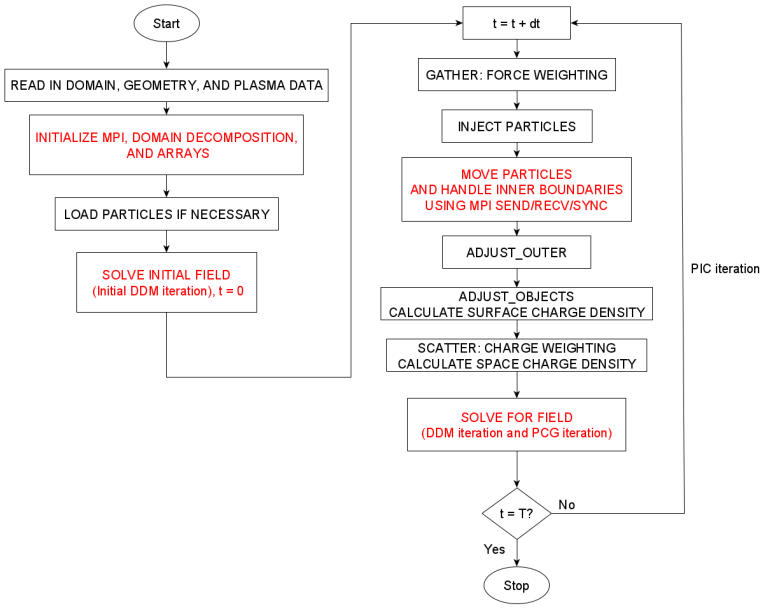
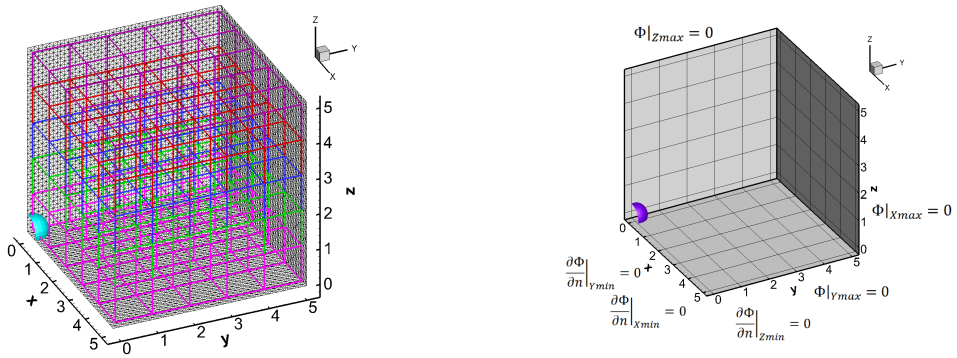


FIG. 2.9. *Flowchart of PIFE-PIC.*



(a) IFE mesh (b) Simulation setup

FIG. 3.1. *IFE mesh and setup used in the 3-D OML sheath problem for code validation. In this example, the global domain is decomposed into 5×5×5 subdomains. Different layers (along z-direction) of the subdomains are highlighted in different colors. 1/8 of the sphere is centered at the origin.*

size of $h = 0.1\lambda_D$ in all dimensions, where λ_D is the Debye length of the plasma. The entire simulation domain has $50\times50\times50 = 125{,}000$ PIC cells which is $125{,}000\times5 = 625{,}000$ tetrahedral FE/IFE cells as each cuboid PIC cell is partitioned into 5 tetrahedral FE/IFE cells in 3-D IFE-PIC [39, 37]. The IFE mesh size is also globally uniform and the same as that of the PIC mesh. The dielectric sphere is centered at $(0, 0, 0)$ with a radius of $R_s = 0.401$. Due to symmetry in all three dimensions, only 1/8 of the sphere is included in the domain. The entire domain is partitioned into $5\times5\times5$ subdomains with each subdomain computed by one MPI process. Figure 3.1 shows the 3-D IFE mesh and setup used in the simulation.

3.1.2. Field setup. At $X_{\max}$, $Y_{\max}$, and $Z_{\max}$ boundaries, the potentials are set to 0 as the reference potential. At $X_{\min}$, $Y_{\min}$, and $Z_{\min}$ boundaries, zero-Neumann boundary conditions are applied due to symmetry (Figure 3.1(b)). The relative permittivity of the sphere is set to 4. The floating potential of the sphere is calculated from the nonhomogeneous flux jump condition at the sphere surface.

3.1.3. Particle setup. The simulation is carried out using the realistic ion-to-electron mass ratio of $m_i/m_e = 1{,}836$. Particles are preloaded into the domain before the initial field solution and injected into the domain at $X_{\max}$, $Y_{\max}$, and $Z_{\max}$ within each PIC step. Particles hitting the $X_{\min}$, $Y_{\min}$, and $Z_{\min}$ boundaries are reflected due to symmetry. Particles hitting the $X_{\max}$, $Y_{\max}$, and $Z_{\max}$ are absorbed and removed from the simulation. The normalized time step size was set to be 0.01. There were 125 particles (5×5×5) per species per cell being loaded/injected into the domain.

3.2. Simulation results. The simulation of the validation case finished in about 2 hours for a total of 50,000 PIC steps on the *Foundry* cluster provided by the Center of High-Performance Computing Research at Missouri University of Science and Technology. The computing nodes are configured with Dell C6525 nodes each having four node chassis with each node containing dual 32-core AMD EPYC Rome 7452 CPUs with 256 GB DDR4 RAM and six 480 GB SSD drives in RAID 0. All other simulations presented in this work were also carried out on the same cluster.

For this test case, the maximum number of PCG iterations was set to 50 with a tolerance of 1×10^{-6} for absolute residual, while the maximum number of initial DDM iterations (solving the initial electrostatic field before main PIC loop starts) was set to 150 and the maximum number of DDM iterations at each PIC iteration step was set to 50 with a tolerance of 1×10^{-2}. The simulation was set to run 50,000 PIC steps.

3.2.1. Initial field solution. The initial field solution (the zeroth PIC step) took about 100 DDM iterations which are more than what is needed at each step of the main PIC loop, to converge in terms of the relative error 1×10^{-2}. The idea of setting a relatively larger DDM iteration number is to obtain a better initial field for the main PIC loop. Since the initial field was solved only once, the extra DDM iterations contributed little to the overall wall-clock time of the entire simulation.

3.2.2. Solution history of main PIC loop. Figure 3.2 shows the field solution convergence history including the maximum absolute PCG residual and maximum DDM relative error as a function of PIC steps in the main PIC loop. A few phenomena are observed here:

1. For most PIC steps, PCG took about 45–50 iterations to reach the tolerance of 1×10^{-6}. The "max" in the vertical axis stands for "maximum among all subdomains" (first plot);
2. The maximum PCG absolute residual of the matrix solver has been maintained below 1×10^{-6} (second plot);
3. At early PIC steps, most DDM steps took about tens of iterations to converge below 1×10^{-2}, while later on as PIC approaches steady state, most DDM steps were able to converge in less than 15 iterations (third and forth plots).

Figure 3.3 shows the global particle number history. At the steady state, there are approximately 3.12×10^7 particles in the entire global domain. It is also shown that the numbers of particles reached steady state at normalized simulation time of about $\hat{t} = 100$.

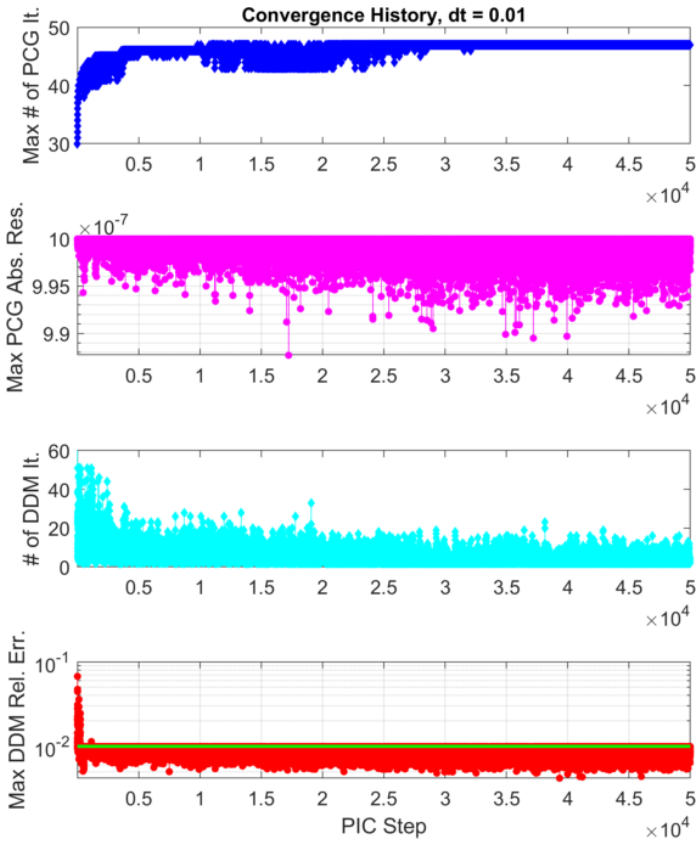


FIG. 3.2. *Field convergence history of the code validation test case, PCG absolute residual, and DDM relative error. The green line on maximum DDM relative error plot is the DDM tolerance.*

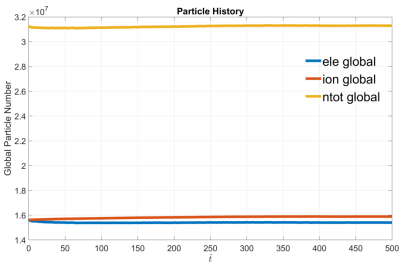
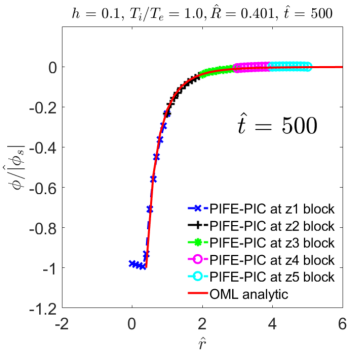


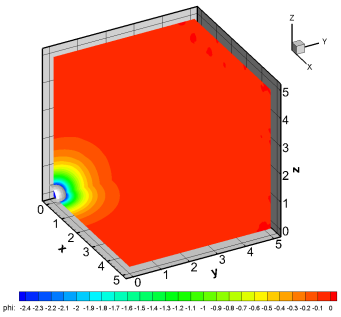
FIG. 3.3. *Global particle history of the code validation test case. "ntot" is the total number of particles (electrons plus ions).*

3.2.3. Comparison with analytic solution. Figure 3.4 shows the comparison between PIFE-PIC simulation results against analytic solution for the OML sheath problem as well as a 3-D potential contour. The potential profile agrees very well with the analytic solution, as also shown in earlier work with the serial IFE-PIC [37, 39].

3.3. Performance profiling. Table 3.1 shows the detailed timer profile of PIFE-PIC on the validation case for the entire 50,000 PIC steps, in terms of the percentage of total wall-clock time of key procedures in PIFE-PIC, namely, "gather,"



(a) Potential profile patched by 5 subdomains (blocks) along z-direction



(b) 3-D potential contours

FIG. 3.4. *Validation of OML sheath solution: PIFE-PIC versus analytic and 3-D potential contours.*

TABLE 3.1
Time percentage breakdown for all 50,000 *PIC steps.*

Computing step	**Percent of total wall-clock time**		
	In rank0001	**In rank0050**	**In rank0100**
Total wall-clock time	100.00%	100.00%	100.00%
Initialization time	0.05%	0.05%	0.05%
Main PIC Loop time	99.95%	99.95%	99.95%
Total gather time	11.21%	11.01%	10.49%
Total particle-push time	47.10%	47.10%	47.10%
Total particle-push-comm (AdjustOuter local) time*	24.89%*	24.88%*	24.89%*
Total adjust-objects time	3.45%	3.45%	3.45%
Total scatter time	3.83%	3.83%	4.10%
Total field-solve time	31.79%	31.78%	31.78%
Total field-solve-phibc (Update Phi BC) time**	4.54%**	22.82%**	22.93%**
Total other time	2.62%	2.83%	3.08%

*Included in the "particle-push time"
**Included in the "field-solve time"

"particle-push," "particle-push-comm" (particle adjustment at local boundaries and communication among subdomains), "adjust-objects" (particle collection and charge deposition), "scatter," "field-solve," "field-solve-phibc" (communication among subdomains and update of local potential boundary conditions), and "other" (including particle injection at global boundaries and calculation of electric field), for selected CPUs/subdomains (also the MPI ranks). Three subdomains (MPI ranks) are chosen, namely, rank0001, rank0050, and rank0100.

Particularly, for "field-solve," the communication time ("field-solve-phibc") is significantly higher for rank0050 and rank0100, indicating they spent more time in the "MPI-wait" part of the communication (because all subdomains did MPI-send/receive with similar load). This means that they were faster than rank0001 in the part of "field-solve." This is because, as shown in Figure 3.1(a), rank0001 is the only subdomain with interface, which adds extra work to rank0001 compared with other interface-free subdomains. Therefore, in order to improve the parallel efficiency, the working load of rank0001 should be reduced. This strategy will be utilized in the next section.

The computing time of "particle-push" essentially depends on the number of simulation particles in the domain, which also affects the accuracy and smoothness of the source term for Poisson's equation. Therefore, in practical PIC simulations,

large numbers of particles are preferred when computing resources are available. The computing time of "field-solve" (including communication) depends on at least two major factors: 1) the size of each subdomain (number of mesh cells and nodes) and balance among subdomains, and 2) the number of DDM iterations. The size of each subdomain can be determined by the domain decomposition configurations, while the number of DDM iterations is affected by the DDM relative error tolerance and the maximum number of DDM iterations. In the following section, we investigate the strong scaling performance of PIFE-PIC by varying 1) the size of each subdomain and 2) the maximum number of DDM iterations.

4. Parallel efficiency: Strong scaling. For most large-scale problems of practical interests, the problem size is usually determined by the physical phenomena to be resolved. Therefore, to test the parallel efficiency of PIFE-PIC, we use the *strong* scaling approach such that the problem size is *fixed* while the number of processors increases. For this set of tests, the problem size was fixed as a $10\times10\times10$ Debye cube with a globally uniform PIC mesh size of $h = 0.1\lambda_D$ in all dimensions. The entire simulation domain has $100\times100\times100 = 1$ million PIC cells (5 million tetrahedral FE/IFE cells) and about 54 million particles. For these runs, the maximum number of PCG iteration was set to 1,000 with a tolerance of 1×10^{-6} for absolute residual. For the initial field solution, the maximum number of DDM iteration was set to 100, while for each step within the main PIC loop, the maximum number of DDM iteration was set to be 10 or 6 for two different groups with same tolerance of 1×10^{-2}. The normalized time step size was set to be 0.01 and all simulations ran for 20,000 PIC steps. The speedup is defined as $S = T_s/T_p$, where T_s is the serial runtime and T_p is the parallel runtime on p MPI processes. The strong scaling parallel efficiency is then defined as $E = S/p = T_s/(p \cdot T_p)$. We chose two groups of configurations to test the parallel efficiency:

- Group I: Using at most 10 DDM iterations per main-loop PIC step;
- Group II: Using at most 6 DDM iterations per main-loop PIC step.

Table 4.1 lists the domain decomposition configurations for each test case, along with the sizes of smallest and biggest subdomains in terms of number of cells in each direction. The different sizes of subdomains indicate load imbalances among CPUs, which may affect the MPI communication cost as shown in Table 3.1, and thus affect the parallel efficiency. Based on the above discussion for Table 3.1, Rank 1, which handles the interface part, has extra computational cost compared with other CPUs, hence should handle smaller subdomain in order to better balance the working loads. Therefore, we utilize the following strategy to decompose the computation domain in each direction: make the first subdomain (block) smaller (but still enclosing the object) and the rest *of the same size*. For instance, to decompose the 100 cells in one direction into 6 blocks, we use the configuration of $(1 \times 10 + 5 \times 18 = 100)$, i.e., the first block takes 10 cells, and the rest 5 blocks take 18 cells each, totaling 100 cells in one direction. For other numbers of blocks to distribute the 100 cells in each direction, we use the following configurations:

- 4 blocks: $1 \times 19 + 3 \times 27 = 100$;
- 5 blocks: $1 \times 12 + 4 \times 22 = 100$;
- 6 blocks: $1 \times 10 + 5 \times 18 = 100$;
- 7 blocks: $1 \times 10 + 6 \times 15 = 100$.

Table 4.2 lists the total wall-clock time, speedup, and parallel efficiency of each case for both Group I and Group II. The timer data was taken over all 20,000 PIC steps. A few trends are observed here:

TABLE 4.1
Domain Decomposition Configurations for Strong Scaling Test Cases.

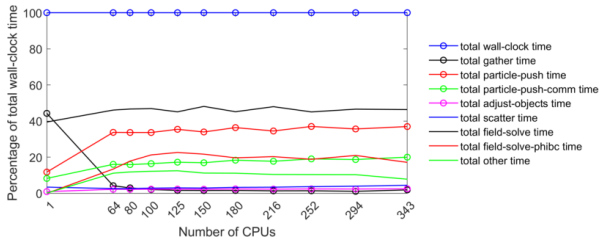
# of subdomains	DD configurations	Size of smallest subdomain (cells)	Size of biggest subdomain (cells)
1 (serial)	1×1×1	100×100×100	100×100×100
64	4×4×4	19×19×19	27×27×27
80	4×4×5	19×19×12	27×27×22
100	4×5×5	19×12×12	27×22×22
125	5×5×5	12×12×12	22×22×22
150	5×5×6	12×12×10	22×22×18
180	5×6×6	12×10×10	22×18×18
216	6×6×6	10×10×10	18×18×18
252	6×6×7	10×10×10	18×18×15
294	6×7×7	10×10×10	18×15×15
343	7×7×7	10×10×10	15×15×15

TABLE 4.2
Strong Scaling Test Results.

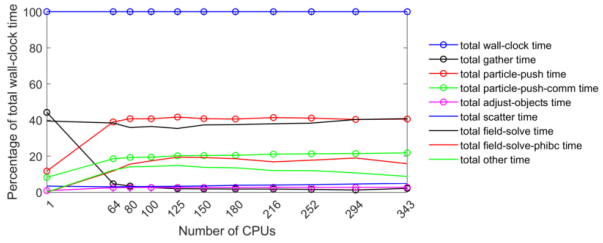
# of subdomains	Total time T_{I} (min)	Speedup S_{I}	Efficiency E_{I}	Total time T_{II} (min)	Speedup S_{II}	Efficiency E_{II}
1 (serial)	12,084	1	100.00%	12,084	1	100.00%
64	200	60.47	94.49%	173	69.77	109.02%
80	163	73.92	92.40%	135	89.59	111.99%
100	143	84.25	84.25%	113	106.57	106.57%
125	111	108.75	87.00%	96	125.98	100.79%
150	99	122.10	81.40%	85	142.49	94.99%
180	80	151.41	84.11%	74	164.01	91.12%
216	74	163.90	75.88%	62	193.57	89.62%
252	63	191.55	76.01%	58	209.40	83.10%
294	65	186.26	63.35%	60	200.95	68.35%
343	57	212.76	62.03%	53	227.99	66.47%

1. In general, the parallel efficiency of cases in Group II (at most 6 DDM iterations per PIC step) is higher than the corresponding case in Group I (at most 10 DDM iterations per PIC step). This is obviously because fewer DDM iterations per PIC step would save more time in the field-solve part, thus resulting in a shorter total wall-clock time.
2. For both Group I and Group II, the parallel efficiency starts quite high (close to or even above 100%) and gradually decays when the CPU number and communications increase.
3. As mentioned above, we also notice that for Group II, some parallel cases achieved >100% *superlinear speedup* in this strong scaling. One possible reason is that, with domain decomposition, the number of mesh nodes of each subdomain is significantly less than that of the entire domain (which is the serial case). Therefore, the matrix size of each subdomain is much smaller than that of the serial case. Since the PCG solver scales at $N\log(N)$ where N is the size of the matrix, the PCG solution of parallel cases are much faster than that of the serial case. Therefore, when the communication overhead is not significant, superlinear speedup may occur. As the number of CPUs increases, the superlinear speedup is suppressed by the communication overhead.

Figure 4.1 plots the percentage of total wall-clock time to show the performance of PIFE-PIC for each domain decomposition configuration in Group I and Group II.



(a) Group I



(b) Group II

FIG. 4.1. *Percentage of total wall-clock time of key procedures, for each domain decomposition configuration.*

Some trends are also observed: 1) The percentage breakdown of the key PIC procedures is fairly consistent across all parallel configurations. 2) The "field-solve" step took about 40% of the total wall-clock time across all parallel cases, whereas about 20% (of the total wall-clock time) was spent on the "field-solve-phibc" step. Based on more detailed data, which is omitted due to the page limitation, we also observe that the percentage of the time for "field-solve-phibc" varies within about 10% for all subdomains/ranks. This is much better balanced than that shown in Table 3.1. 3) The "particle-push" step takes about 35–40% of the total wall-clock time consistently across all parallel cases, whereas about 20% (of the total wall-clock time) is spent on the "particle-push-comm" step.

5. Application to lunar crater charging. In this section, we apply PIFE-PIC to simulate the plasma charging at a lunar crater under average solar wind (SW) conditions to demonstrate the large-scale simulation capability of PIFE-PIC. In the following, we will first briefly describe the lunar surface charging problem, then introduce the setup of the simulation, and finally present the results and discussion.

5.1. Problem description. The problem considered is SW plasma charging near the lunar surface, specifically, near the lunar craters at the terminator region for lunar exploration missions. The Moon is directly exposed to the solar radiation and various space plasma environments which directly interact with the lunar surface. A direct consequence of such interactions is surface charging. Observations have found that the potential of the sunlit surface is typically a few tens of volts positive with respect to ambient due to photoelectron emission, while that of the surface in shadow can be hundreds to thousands of volts negative because of the hot electron flux from ambient plasma that can dominate the charging process [10, 23, 32, 33, 34, 72, 74, 82]. Both solar illumination and plasma flow can have a substantial influence on lunar surface charging. At the lunar terminator, the rugged surface terrain, such as that near a crater, generates localized plasma wakes and shadow regions which can lead to

TABLE 5.1
Average SW and photoelectron (at 90° Sun elevation angle) parameters.

	Number density n, cm^{-3}	Drifting velocity v_d, km/s	Thermal velocity v_t, km/s	Temperature T, eV	Debye length λ_D, m
SW electron	8.7	468	1453	12	8.73
SW ion	8.7	468	31	10	N/A*
Photoelectron	64	N/A*	622	2.2	1.38

*N/A denotes "not applicable"

strong differential charging at the surface [9, 70, 81]. Both the localized plasma flow field and the charged lunar surface are expected to have substantial influence on the charging of spacecraft/landers/rovers/habitats for future surface missions.

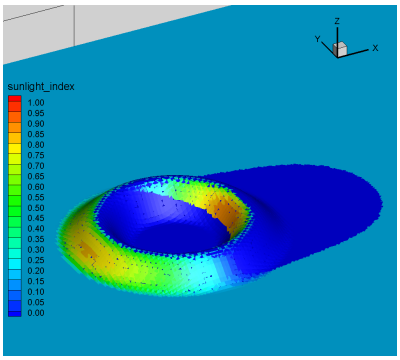
The lunar surface is covered by the lunar regolith layer which separates the solid bedrock from the plasma environment. The regolith layer in most areas is about 4 to 20 meters thick [68, 73]. A complete model of plasma charging on the lunar surface needs to explicitly take into account the properties of the regolith layer, such as permittivity, layer thickness, and the lunar electrical ground.

The serial version of IFE-PIC method has been successfully applied to simulations of lunar plasma charging [38]. In order to illustrate the high performance computing capability of the PIFE-PIC package in this paper, we apply PIFE-PIC to a much larger scale parallel simulation with a larger simulation domain including a lunar crater and much more simulation particles. The plasma environment is chosen to be the average SW and photoelectron parameters at the lunar surface [81], as shown in Table 5.1. It is noted here that the Debye length of photoelectrons at 90° Sun elevation angle (1.38 m) is used as the reference length to normalize spatial dimensions in PIFE-PIC.

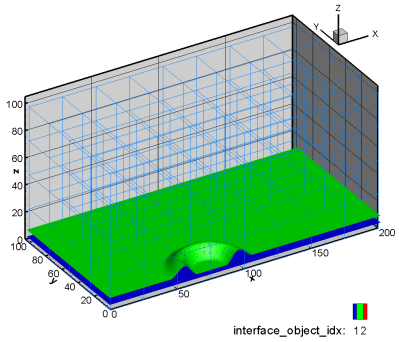
5.2. Simulation setup.

5.2.1. Lunar crater geometry and simulation domain. In PIFE-PIC, the geometry of the lunar crater is realized through an algebraic equation describing the surface terrain in the form of $z = z(x, y)$ where z denotes the surface height. For the lunar crater considered here, the shape is realized by a few characteristic parameters such as inner-rim radius, outer-rim radius, depth, rim height, etc. (Figure 5.1(a)) according to the *Lunar Sourcebook* [49]. The specific diameter of a real lunar crater can be measured through NASA Jet Propulsion Laboratory's website, Moon Trek [69]. The crater considered in this study has these characteristic dimensions: inner-rim radius $10.5\times1.38 = 14.49$ m, top-rim radius $20.2\times1.38 = 27.88$ m, outer-rim radius $30.9\times1.38 = 42.64$ m, and top height $6.7\times1.38 = 9.25$ m. Details of the approach to set up the lunar crater geometry is given in [67].

The simulation domain has $200\times100\times100 = 2$ million PIC cells (10 million tetrahedral FE/IFE cells) including half of the lunar crater due to symmetry with respect to the X-Z plane at $y = 0$ (Figure 5.1(a)). Each PIC cell is a $1.38\times1.38\times1.38$ cube. In physical units, the domain size is approximately 276 m by 138 m by 138 m. At the $Z_{\min}$ boundary, the simulation domain includes a layer of the lunar bedrock with a thickness of $L_{\text{bedrock}} = 4.5\times1.38 = 6.21$ m. On top of the bedrock is a layer of dielectric regolith with a thickness of $L_{\text{regolith}} = (9.5 - 4.5)\times1.38 = 6.9$ m. The relative permittivities of the lunar regolith layer and the bedrock are taken to be $\epsilon_{\text{regolith}} = 4$ and $\epsilon_{\text{bedrock}} = 10$, respectively [49]. Three-dimensional domain decomposition of $8\times4\times4$ (total 128 MPI processes) is used to run the simulation (Figure 5.1(b)).



(a) The geometry of the lunar crater realized in PIFE-PIC. Color contours show the "sunlight index" indicating the inner product of Sun vector (10° above the ground in the $X-Z$ plane) and local surface normal vector.



(b) The simulation domain including the lunar bedrock (below the blue layer) and the lunar regolith (between the blue and green layers). The light-blue edges show the domain decomposition (8×4×4 = 128 MPI processes).

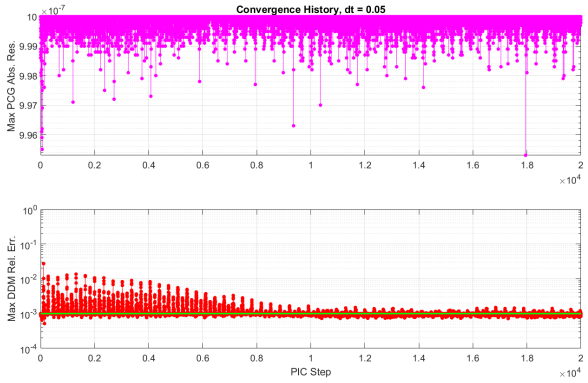
FIG. 5.1. *The lunar crater geometry and simulation domain.*

5.2.2. Particle and field boundary conditions. Particles representing SW ions and electrons are preloaded and injected into the domain with an angle of 10° towards the surface in the X-Z plane (Figure 5.1(a)). Particles representing photoelectrons are generated at the sunlit regions according to the local sunlight index. At the global $X_{\min}$, $X_{\max}$, $Y_{\max}$, and $Z_{\max}$ domain boundaries, ambient SW particles are injected. Particles hitting the global $Y_{\min}$ boundary are reflected due to symmetry. Particles hitting the lunar surface are collected, and their charges are accumulated to calculate surface charging.

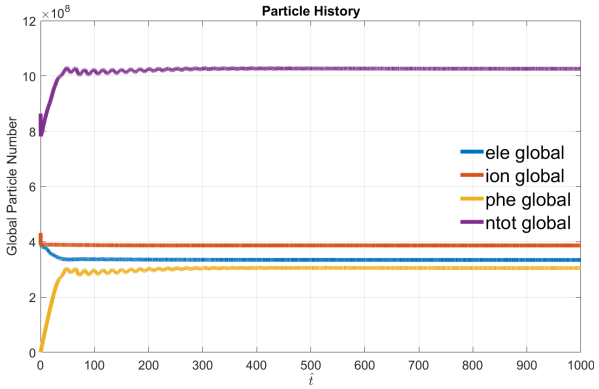
Dirichlet boundary condition of $\Phi = 0$ is applied at the $Z_{\max}$ boundary (the unperturbed SW), whereas Neumann boundary condition of zero electric field is applied on all other five domain boundaries. The PCG maximum iterations was set to 150 with a tolerance of 1×10^{-6} for absolute residual. The maximum number of DDM iteration for initial field solution was set to 800, and the maximum number of DDM iteration for each step within the main PIC loop was set to 200 with a tolerance of 1×10^{-3}. The simulation ran for 20,000 PIC steps.

5.3. Convergence history. The run took about 154 hours to finish 20,000 PIC steps with the time step size of 0.05 (total simulation time till $\hat{t} = 1,000$). Figure 5.2 shows the convergence history of the lunar crater charging simulation including the maximum absolute PCG residual and maximum DDM relative error and particle number histories. It is shown that the field solution residuals and relative errors started to level off near PIC step of 10,000 ($\hat{t} = 500$), and at steady state, the entire domain had about 1.03 billion particles. After $\hat{t} = 500$, the autosaved simulation results are all similar. The results presented below are those at $\hat{t} = 1,000$.

5.4. Surface charging results. Figure 5.3 illustrate the density contours of SW ions, solar wind electrons, photoelectrons, and total space charge near the crater. The SW ion and electron density contours clearly exhibit a localized plasma wake formed by the crater rim. The photoelectron density contours clearly exhibit the lack of photoemission in the shadow region. The total space charge density contours show the nonneutral regions associated with the wake caused by the crater rim.



(a) Field solution PCG absolute residual and DDM relative error history.

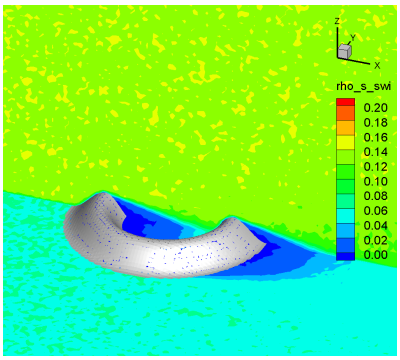


(b) Global particle number history.

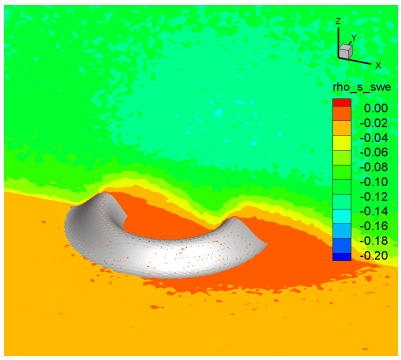
FIG. 5.2. *The lunar crater simulation convergence history.*

Figure 5.4 illustrates the potential contours of the domain and near the crater. It is shown, for the average SW conditions considered here, the surface potential in the sunlit region of the crater is charged to about $16\times 2.2 \simeq 35$ V while the surface in the shadow region is charged to about $-24 \times 2.2 \simeq -53$ V. It is noted as this length scale is on the order of tens of meters, such differential surface charging will affect the lunar surface activities for exploration missions, such as the risk of discharging/arcing and horizontal/vertical transport of lofted charged lunar dusts.

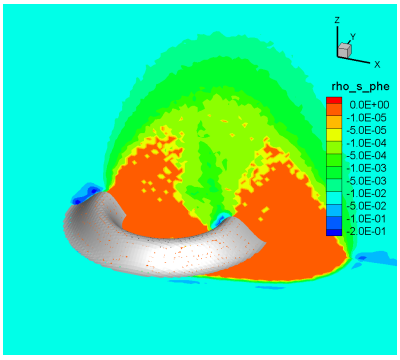
6. Summary and conclusion. In this paper, we presented a most recently developed 3-D PIFE-PIC method, for kinetic particle simulations of PMIs especially electrostatic surface charging. PIFE-PIC is based on the serial nonhomogeneous electrostatic IFE-PIC algorithm, which was designed to handle complex interface conditions associated with irregular geometries while maintaining the computational speed of Cartesian mesh-based PIC. Three-dimensional domain decomposition is used in both field-solve and particle-push procedures of PIC to distribute the computation among multiple processors. A validation case of 3-D OML sheath of a dielectric sphere immersed in a stationary plasma was carried out and results agreed well with



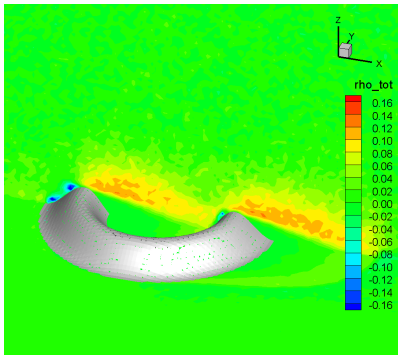
(a) SW ion density contours.



(b) SW electron density contours.

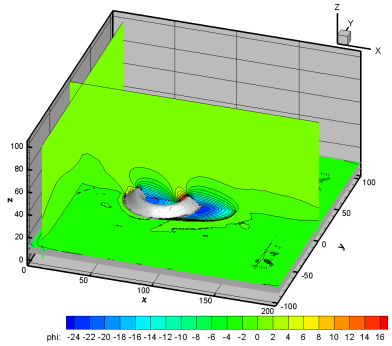


(c) Photoelectron density contours.

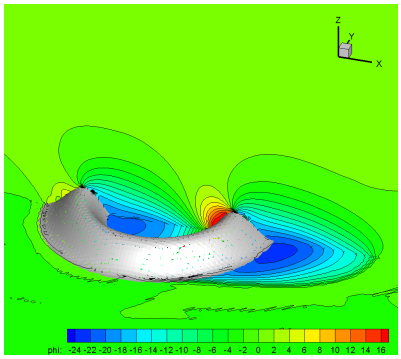


(d) Total charge density contours.

FIG. 5.3. *Normalized density contours. For electrons, numerical values include a negative sign indicating the negative charges. The densities are normalized by* 64 cm^{-3}*, and the spatial dimensions are normalized by* 1.38 *m.*



(a) Potential contours showing the differential charging near the lunar crater.



(b) Zoom-in view of the potential contours near the lunar crater.

FIG. 5.4. *Potential contours of lunar crater charging. The potential values are normalized by* 2.2 *V, and the spatial dimensions are normalized by* 1.38 *m.*

the analytic solution. A series of strong scaling tests were performed to profile the parallel efficiency for a problem of fixed size which has 1 million PIC cells (5 million tetrahedral FE/IFE cells), about 54 million particles, and running 20,000 PIC steps on the Foundry cluster at Missouri University of Science and Technology. Parallel efficiency up to approximately 110% superlinear speedup was achieved.

An application of PIFE-PIC to a larger problem, SW plasma charging at a lunar crater, is presented to show the capability of PIFE-PIC for practical problems of science and engineering interest. The lunar crater charging simulation has 2 million PIC cells (10 million tetrahedral FE/IFE cells), about 1 billion particles, and running for 20,000 PIC steps. The simulation finished in about 154 wall-clock hours with domain decomposition of $8\times4\times4 = 128$ MPI processes. This demonstrates that PIFE-PIC can be utilized to carry out realistic large-scale particle simulations of PMIs routinely on supercomputers with distributed memory.

REFERENCES

[1] S. Adjerid, N. Chaabane, T. Lin, and P. Yue, *An immersed discontinuous finite element method for the Stokes problem with a moving interface*, J. Comput. Appl. Math., 362 (2019), pp. 540–559.

[2] S. Adjerid and T. Lin, *p-th degree immersed finite element for boundary value problems with discontinuous coefficients*, Appl. Numer. Math, 59 (2009), pp. 1303–1321.

[3] S. Adjerid, T. Lin, and Q. Zhuang, *Error estimates for an immersed finite element method for second order hyperbolic equations in inhomogeneous media*, J. Sci. Comput., 84 (2020), p. 35.

[4] S. Adjerid and K. Moon, *An immersed discontinuous Galerkin method for acoustic wave propagation in inhomogeneous media*, SIAM J. Sci. Comput., 41 (2019), pp. A139–A162.

[5] J. Bai, Y. Cao, Y. Chu, and X. Zhang, *An improved immersed finite element particle-in-cell method for plasma simulation*, Comput. Math. Appl., 75 (2018), pp. 1887–1899.

[6] J. Bai, Y. Cao, X.-M. He, and P. E, *An implicit particle-in-cell model based on anisotropic immersed-finite-element method*, Comput. Phys. Commun., 261 (2021), 107655.

[7] J. Bai, Y. Cao, X.-M. He, H. Liu, and X. Yang, *Modeling and an immersed finite element method for an interface wave equation*, Comput. Math. Appl., 76 (2018), p. 1625–1638.

[8] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, 1996.

[9] O. E. Berg, *A lunar terminator configuration*, Earth Planet. Sci. Lett., 39 (1978), pp. 377–381.

[10] O. E. Berg, F. F. Richardson, and H. Burton, *Apollo 17 preliminary science report*, Technical Report NASA SP-330, National Aeronautics and Space Administration, 1973.

[11] C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation (Series in Plasma Physics)*, Institute of Physics Publishing, Bristol, UK, 1991.

[12] B. Camp, T. Lin, Y. Lin, and W. Sun, *Quadratic immersed finite element spaces and their approximation capabilities*, Adv. Comput. Math., 24 (2006), pp. 81–112.

[13] H. Cao, Y. Cao, Y. Chu, X.-M. He, and T. Lin, *A Huygens immersed-finite-element particle-in-cell method for modeling plasma-surface interactions with moving interface*, Commun. Nonlinear Sci. Numer. Simul., 59 (2018), pp. 132–148.

[14] W. Cao, X. Zhang, and Z. Zhang, *Superconvergence of immersed finite element methods for interface problems*, Adv. Comput. Math., 43 (2017), pp. 795–821.

[15] Y. Cao, Y. Chu, X.-M. He, and T. Lin, *An iterative immersed finite element method for an electric potential interface problem based on given surface electric quantity*, J. Comput. Phys., 281 (2015), pp. 82–95.

[16] Y. Cao, Y. Chu, X. Zhang, and X. Zhang, *Immersed finite element methods for unbounded interface problems with periodic structures*, J. Comput. Appl. Math., 307 (2016), pp. 72–81.

[17] Y. Chu, Y. Cao, X.-M. He, and M. Luo, *Asymptotic boundary conditions with immersed finite elements for interface magnetostatic/electrostatic field problems with open boundary*, Comput. Phys. Commun., 182 (2011), pp. 2331–2338.

[18] Y. Chu, D. Han, Y. Cao, X.-M. He, and J. Wang, *An immersed-finite-element particle-in-cell simulation tool for plasma surface interaction*, Int. J. Numer. Anal. Model., 14 (2017), pp. 175–200.

[19] G. L. Delzanno and X.-Z. Tang, *Comparison of dust charging between orbital-motion-limited theory and particle-in-cell simulations*, Phys. Plasmas, 22 (2015), 113703.
[20] D. Depew, D. Han, J. Wang, X.-M. He, and T. Lin, *Immersed-Finite-Element Particle-In-Cell simulations of lunar surface charging, #*199, Proceedings of the 13th Spacecraft Charging Technology Conference, Pasadena, California, June 23-27, 2014.
[21] R. E. Ewing, Z. Li, T. Lin, and Y. Lin, *The immersed finite volume element methods for the elliptic interface problems. Modelling '*98 *(Prague)*, Math. Comput. Simulation, 50 (1999), pp. 63–76.
[22] W. Feng, X.-M. He, Y. Lin, and X. Zhang, *Immersed finite element method for interface problems with algebraic multigrid solver*, Commun. Comput. Phys., 15 (2014), pp. 1045–1067.
[23] J. W. Freeman and M. Ibrahim, *Lunar electric fields, surface potential and associated plasma sheaths*, The Moon, 14 (1975), pp. 103–114.
[24] Y. Gong, B. Li, and Z. Li, *Immersed-interface finite-element methods for elliptic interface problems with non-homogeneous jump conditions*, SIAM J. Numer. Anal., 46 (2008), pp. 472–495.
[25] R. Guo and T. Lin, *A group of immersed finite-element spaces for elliptic interface problems*, IMA J. Numer. Anal., 39 (2019), pp. 482–511.
[26] R. Guo and T. Lin, *A higher degree immersed finite element method based on a Cauchy extension for elliptic interface problems*, SIAM J. Numer. Anal., 57 (2019), pp. 1545–1573.
[27] R. Guo and T. Lin, *An immersed finite element method for elliptic interface problems in three dimensions*, J. Comput. Phys., 414 (2020), 109478.
[28] R. Guo, T. Lin, and Y. Lin, *Error estimates for a partially penalized immersed finite element method for elasticity interface problems*, ESAIM Math. Model. Numer. Anal., 54 (2020), pp. 1–24.
[29] R. Guo, T. Lin, and Q. Zhuang, *Improved error estimation for the partially penalized immersed finite element methods for elliptic interface problems*, Int. J. Numer. Anal. Mod., 16 (2019), pp. 575–589.
[30] R. Guo and X. Zhang, *Solving three-dimensional interface problems with immersed finite elements: A-priori error analysis*, J. Comput. Phys., 441 (2021), 110445.
[31] J. Guzmán, M. A. Sánchez, and M. Sarkis, *Higher-order finite element methods for elliptic problems with interfaces*, ESAIM Math. Model. Numer. Anal., 50 (2016), pp. 1561–1583.
[32] J. S. Halekas, G. T. Delory, D. A. Brain, R. P. Lin, M. O. Fillingim, C. O. Lee, R. A. Mewaldt, T. J. Stubbs, W. M. Farrell, and M. K. Hudson, *Extreme lunar surface charging during solar energetic particle events*, Geophys. Res. Lett., 34 (2007), L02111.
[33] J. S. Halekas, G. T. Delory, R. P. Lin, T. J. Stubbs, and W. M. Farrell, *Lunar prospector observations of the electrostatic potential of the lunar surface and its response to incident currents*, J. Geophys. Res., 113 (2008), A09102.
[34] J. S. Halekas, Y. Saito, G. T. Delory, and W. M. Farrell, *New views of the lunar plasma environment*, Planet. Space Sci., 59 (2011), pp. 1681–1694.
[35] D. Han, *Particle-in-Cell Simulations of Plasma Interactions with Asteroidal and Lunar Surfaces*, PhD thesis, University of Southern California, 2015.
[36] D. Han and J. Wang, 3*-D fully-kinetic particle-in-cell simulations of small asteroid charging in the solar wind*, IEEE Trans. Plasma Sci., 47 (2019), pp. 3682–3688.
[37] D. Han, J. Wang, and X.-M. He, *A non-homogeneous immersed-finite-element particle-in-cell method for modeling dielectric surface charging in plasmas*, IEEE Trans. Plasma Sci., 44 (2016), pp. 1326–1332.
[38] D. Han, J. Wang, and X.-M. He, *Immersed-finite-element particle-in-cell simulations of plasma charging at lunar terminator*, J. Spacecraft Rockets, 55 (2018), pp. 1490–1497.
[39] D. Han, P. Wang, X.-M. He, T. Lin, and J. Wang, *A* 3*D immersed finite element method with non-homogeneous interface flux jump for applications in particle-in-cell simulations of plasma-lunar surface interactions*, J. Comput. Phys., 321 (2016), pp. 965–980.
[40] C. He and X. Zhang, *Residual-based a posteriori error estimation for immersed finite element methods*, J. Sci. Comput., 81 (2019), pp. 2051–2079.
[41] X.-M. He, *Bilinear immersed finite elements for interface problems*, Ph.D. Dissertation, Virginia Polytechnic Institute and State University, (2009).
[42] X.-M. He, T. Lin, and Y. Lin, *Approximation capability of a bilinear immersed finite element space*, Numer. Methods Partial Differential Equations, 24 (2008), pp. 1265–1300.
[43] X.-M. He, T. Lin, and Y. Lin, *A bilinear immersed finite volume element method for the diffusion equation with discontinuous coefficients, dedicated to Richard E. Ewing on the occasion of his* 60*th birthday*, Commun. Comput. Phys., 6 (2009), pp. 185–202.

[44] X.-M. He, T. Lin, and Y. Lin, *Interior penalty bilinear IFE discontinuous Galerkin methods for elliptic equations with discontinuous coefficient, dedicated to David Russell's 70th birthday*, J. Syst. Sci. Complex., 23 (2010), pp. 467–483.
[45] X.-M. He, T. Lin, and Y. Lin, *Immersed finite element methods for elliptic interface problems with non-homogeneous jump conditions*, Int. J. Numer. Anal. Model., 8 (2011), pp. 284–301.
[46] X.-M. He, T. Lin, and Y. Lin, *The convergence of the bilinear and linear immersed finite element solutions to interface problems*, Numer. Methods Partial Differential Equations, 28 (2012), pp. 312–330.
[47] X.-M. He, T. Lin, and Y. Lin, *A selective immersed discontinuous Galerkin method for elliptic interface problems*, Math. Methods Appl. Sci., 37 (2014), pp. 983–1002.
[48] X.-M. He, T. Lin, Y. Lin, and X. Zhang, *Immersed finite element methods for parabolic equations with moving interface*, Numer. Methods Partial Differential Equations, 29 (2013), pp. 619–646.
[49] G. H. Heiken, D. T. Vaniman, and B. M. French, *Lunar Sourcebook: A User's Guide to the Moon*, Cambridge University Press, Cambridge, 1991.
[50] R. W. Hockney and J. W. Eastwood, *Computer Simulation Using Particles*, Adam Hilger, London, 1988.
[51] J. D. Jackson, *Classical electrodynamics*, 3rd ed., Wiley, New York, 1999.
[52] H. Jian, Y. Chu, H. Cao, Y. Cao, X.-M. He, and G. Xia, *Three-dimensional IFE-PIC numerical simulation of background pressure's effect on accelerator grid impingement current for ion optics*, Vacuum, 116 (2015), pp. 130–138.
[53] R. Kafafy, T. Lin, Y. Lin, and J. Wang, *Three-dimensional immersed finite element methods for electric field simulation in composite materials*, Int. J. Numer. Meth. Engrg., 64 (2005), pp. 940–972.
[54] R. Kafafy and J. Wang, *Whole subscale ion optics simulation: Direct ion impingement and electron backstreaming*, in 41st AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, AIAA 2005-3691, Tucson, Arizona, July 2005.
[55] R. Kafafy and J. Wang, *Whole ion optics gridlet simulations using a hybrid-grid immersed-finite-element particle-in-cell code*, J. Propul. Power, 23 (2007), pp. 59–68.
[56] R. I. Kafafy and J. Wang, *A hybrid grid immersed finite element particle-in-cell algorithm for modeling spacecraft-plasma interactions*, IEEE Trans. Plasma Sci., 34 (2006), pp. 2114–2124.
[57] D. Kwak, S. Jin, and D. Kyeong, *A stabilized $P1$-nonconforming immersed finite element method for the interface elasticity problems*, ESAIM Math. Model. Numer. Anal., 51 (2017), pp. 187–207.
[58] Z. Li, *The immersed interface method using a finite element formulation*, Appl. Numer. Math., 27 (1998), pp. 253–267.
[59] Z. Li, T. Lin, Y. Lin, and R. C. Rogers, *An immersed finite element space and its approximation capability*, Numer. Methods Partial Differential Equations, 20 (2004), pp. 338–367.
[60] Z. Li, T. Lin, and X. Wu, *New Cartesian grid methods for interface problems using the finite element formulation*, Numer. Math., 96 (2003), pp. 61–98.
[61] T. Lin, Y. Lin, and X. Zhang, *Partially penalized immersed finite element methods for elliptic interface problems*, SIAM J. Numer. Anal., 53 (2015), pp. 1121–1144.
[62] T. Lin, D. Sheen, and X. Zhang, *A locking-free immersed finite element method for planar elasticity interface problems*, J. Comput. Phys., 247 (2013), pp. 228–247.
[63] T. Lin, D. Sheen, and X. Zhang, *A nonconforming immersed finite element method for elliptic interface problems*, J. Sci. Comput., 79 (2019), pp. 442–463.
[64] T. Lin, Q. Yang, and X. Zhang, *A priori error estimates for some discontinuous Galerkin immersed finite element methods*, J. Sci. Comput., 65 (2015), pp. 875–894.
[65] C. Lu, J. Wan, Y. Cao, and X.-M. He, *A fully decoupled iterative method with three-dimensional anisotropic immersed finite elements for Kaufman-type discharge problems*, Comput. Meth. Appl. Mech. Eng., 372 (2020), 113345.
[66] C. Lu, Z. Yang, J. Bai, Y. Cao, and X.-M. He, *Three-dimensional immersed finite element method for anisotropic magnetostatic/electrostatic interface problems with non-homogeneous flux jump*, Int. J. Numer. Meth. Eng., 121 (2020), pp. 2107–2127.
[67] D. Lund, J. Zhao, A. Lamb, and D. Han, Fully Kinetic PIFE-PIC Simulations of Plasma Charging at Lunar Craters, in AIAA SciTech 2020, AIAA 2020-1549, Orlando, Florida, January 6-10, 2020.
[68] D. S. McKay, G. Heiken, A. Basu, G. Blanford, S. Simon, R. Reedy, B. M. French, and J. Papike, *Chapter 7: The lunar regolith*, in Lunar Sourcebook: A User's Guide to the Moon, Cambridge University Press, Cambridge, 1991, pp. 285–356.

[69] NASA-JPL, *Moon Trek*, trek, nasa.gov/moon (15 July 2020).
[70] A. R. Poppe, M. Piquette, A. Likhanskii, and M. Horányi, *The effect of surface topography on the lunar photoelectron sheath and electrostatic dust transport*, Icarus, 221 (2012), pp. 135–146.
[71] T. Preusser, M. Rumpf, S. Sauter, and L. O. Schwen, *3D composite finite elements for elliptic boundary value problems with discontinuous coefficients*, SIAM J. Sci. Comput., 33 (2011), pp. 2115–2143.
[72] D. L. Reasoner and W. J. Burke, *Characteristics of the lunar photoelectron layer in the geomagnetic tail*, J. Geophys. Res., 77 (1972), pp. 6671–6687.
[73] Y. G. Shkuratov and N. V. Bondarenko, *Regolith layer thickness mapping of the moon by radar and optical data*, Icarus, 149 (2001), pp. 329–338.
[74] T. J. Stubbs, J. S. Halekas, W. M. Farrell, and R. R. Vondrak, *Lunar surface charging: A global perspective using lunar prospector data*, in Workshop on Dust in Planetary Systems (ESA SP-643), H. Krueger and A. Graps, eds., Kauai, Hawaii, USA, September 26-30, 2005, pp. 181–184.
[75] X.-Z. Tang and G. L. Delzanno, *Orbital-motion-limited theory of dust charging and plasma response*, Phys. Plasmas, 21 (2014), p. 123708.
[76] D. Han and J. Wang, Simulations of Ion Thruster Plume Contamination with A Whole Grid Sputtered Mo Source Model, in 49th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, AIAA 2013-3888, San Jose, California, July 14 - 17, 2013.
[77] S. Vallaghè and T. Papadopoulo, *A trilinear immersed finite element method for solving the electroencephalography forward problem*, SIAM J. Sci. Comput., 32 (2010), pp. 2379–2394.
[78] C. Wang, P. Sun, and Z. Li, *An iterative approach for constructing immersed finite element spaces and applications to interface problems*, Int. J. Numer. Anal. Model., 16 (2019), pp. 167–191.
[79] J. Wang, Y. Cao, R. Kafafy, J. Pierru, and V. K. Decyk, *Simulations of ion thruster plume-spacecraft interactions on parallel supercomputer*, IEEE Trans. Plasma Sci., 34 (2006), pp. 2148–2158.
[80] J. Wang, X.-M. He, and Y. Cao, *Modeling spacecraft charging and charged dust particle interactions on lunar surface*, in Proceedings of the 10th Spacecraft Charging Technology Conference, Biarritz, France, 2007.
[81] J. Wang, X.-M. He, and Y. Cao, *Modeling electrostatic levitation of dusts on lunar surface*, IEEE Trans. Plasma Sci., 36 (2008), pp. 2459–2466.
[82] R. Willis, M. Anderegg, B. Feuerbacher, and B. Fitton, *Photoemission and secondary electron emission from lunar surface material*, in Photon and Particle Interactions with Surfaces in Space, R. Grard, ed., Astrophysics and Space Science Library 37, Springer, Amsterdam, 1973.
[83] W. Yu, D. Han, and J. Wang, *Numerical Simulations of Dust Dynamics Around Small Asteroids*, IEEE Trans. Plasma Sci., 47 (2019), pp. 3724–3730.
[84] W. Yu, J. J. Wang, and D. Han, *Numerical Modeling of Dust Dynamics Around Small Asteroids*, in AIAA SPACE Forum 2016, AIAA 2016-5447, Long Beach, California, September 13-16, 2016.
[85] H. Zhang, T. Lin, and Y. Lin, *Linear and quadratic immersed finite element methods for the multi-layer porous wall model for coronary drug-eluting stents*, Int. J. Numer. Anal. Mod., 15 (2018), pp. 48–73.
[86] Q. Zhang, K. Ito, Z. Li, and Z. Zhang, *Immersed finite elements for optimal control problems of elliptic PDEs with interfaces*, J. Comput. Phys., 298 (2015), pp. 305–319.
[87] X. Zhang, *Nonconforming immersed finite element methods for interface problems*, Ph.D. Dissertation, Virginia Polytechnic Institute and State University, 2013.