J. Plasma Phys. (2020), *vol.* 86, 905860308 © The Author(s), 2020.
Published by Cambridge University Press
doi:10.1017/S0022377820000410

A parallel particle-in-cell code for spacecraft charging problems

Kai Zhang[1], Shiying Cai[1], Chunpei Cai[1,†] and David L. Cooke[2]

[1]Department of Mechanical Engineering-Engineering Mechanics, Michigan Technological University, Houghton, MI 49931, USA

[2]Space Vehicle Directorate, Air Force Research Laboratory, Kirtland Air Force Base, Albuquerque, NM 87117, USA

(Received 25 September 2019; revised 17 April 2020; accepted 20 April 2020)

This paper reports the recent development of a full-scale particle-in-cell (PIC) simulation package highlighting an efficient electro-statics (ES)-PIC implementation for spacecraft charging problems. Numerical simulations are crucial in studying plasma flows because analytical solutions are rare, and experiments are expensive. There are many types of plasma flow that need various numerical methods for efficient and accurate simulations; as such, how to implement and organize those different methods into one comprehensive simulation package is challenging. This work adopted several modern software design patterns and developed a versatile package that includes various PIC schemes. This package has an open architecture, clean interfaces with both serial and parallel simulation capabilities. Two benchmark test cases are included to demonstrate the capabilities of this package. Then two plasma flows around a positively charged probe are simulated, and the results are discussed. The simulation results are consistent with past simulation results, and new insights are obtained. This work can lead to the development and organization of more sophisticated plasma simulation solvers in the future.

Key words: plasma flows, plasma simulation, space plasma physics

1. Introduction

Plasma interactions may include multiple physics, multiple species and there are numerous practical applications. The length scale may range from the size of a galaxy to nano-scale etching processes, and the characteristic time scale can vary significantly as well, for example, ranging from pico-seconds to seconds. Plasma applications include flames, lighting, auroras, fusion reactors and nebulae. Typical practical application of plasma flows in space engineering involves spacecraft–space plasma environment interactions, such as materials aging, sputtering and downgrading. Dilute plasma flows over a positive probe are fundamental problems with complex physics without a good understanding as yet, and there are many applications, such as plasma flows over a tethering cable or a satellite antenna. One goal of the paper is to present our recent investigations on this problem.

† Email address for correspondence: ccai@mtu.edu

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

There are a variety of methods to study plasma flows. One approach is theoretical study, and there are fruitful results. However, due to the complex physics, various simplifications must be adopted to perform analytical investigations, and there are uncertainties in the final results. The second category of investigation methods are experiments which can provide practical and valuable benchmark test cases. Because complex plasma physics may be well beyond the laboratory capabilities, numerical simulations, which form the third category of investigation methods, stand out as invaluable tools.

There are many numerical simulation methods, and they can be divided into three categories. The first category of method solves for the macroscopic level magnetohydrodynamic equations and treats electrons and ions as one fluid. These methods become inaccurate in problems where the detailed kinetic processes (e.g. wave–particle interactions, wave trapping phenomena) affect the plasma macroscopic behaviours. The second category of method simulates the plasma flows at the mesoscopic level by using the gas kinetic theory, where electrons and ions are treated as discrete group of particles. In this category, the particle-in-cell (PIC) methods (for example, Liewer & Decyk (1989), Birdsall & Langdon (1991), Verboncoeur, Langdon & Gladd (1995), Blahovec, Bowers & Luginsland (2000) and Derouilla *et al.* (2017)) are very popular. The PIC method investigates electrons and ions by using macro-particles, and essentially the investigations are performed particle by particle. The PIC method is considered the most accurate. Based on different situations, specific simplifications may be further adopted to aid investigations, such as the electrostatic (ES) simulation method. There are also other plasma simulation methods, such as the hybrid method where ions are simulated as particles and electrons are treated as fluids due to their high mobility. It is always desirable to develop a comprehensive plasma flow simulation package which includes many solvers, and can simulate on parallel or serial computation machines. However, to maintain an open architecture and clean interfaces in one package is challenging, especially when parallel computation capabilities are desired.

There are two major purposes of this paper. The first one is to present our recent work on developing a parallel computing PIC simulation package with an open architecture and other design patterns. The other purpose is to present numerical investigations of plasma flows over a positive probe. Two benchmark test cases with analytical solutions are selected to validate the codes and demonstrate their capabilities. After that, we present two more advanced cases, dilute plasma flows over positively charged probes with electric field (E) $\times$ magnatic field (B) drifting velocities (Singh, Leung & Singh 2000; Heinrich & Cooke 2013). They are practical problems with space engineering applications. These problems are challenging to investigate, especially when an external magnetic field is present.

The rest of this paper is organized as follows. Section 2 presents the new package and its merits; § 3 includes two benchmark test cases and the simulation results are compared with theoretical results; § 4 presents investigations on plasma flows over a positively charged probe; and § 5 summarizes this work with a few conclusions.

2. Design patterns and package features

While this PIC package is designed for general plasma simulation purposes, the current version focuses on effective full kinetic ES-PIC simulations, and the electro-magnetic (EM)-PIC module will be added in the near future. The current package is especially suitable to investigate plasma flows where the electron kinetics

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

are crucial, and the induced magnetic fields are small. This section discusses some existing PIC codes in the literature and presents several novel design patterns and features incorporated in the package.

2.1. *Simulation package development*

The C++ language (Lippman, Lajoie & Moo 2005) has been widely adopted for large-scale software development, and it has many merits. One example is the object-orientated programming feature with the concepts of classes and objects. The inheritance feature of classes allows one simulation scheme to be derived from another simulation scheme. Simple member functions in the daughter classes can override the corresponding ones in their mother classes. In this work, the C++ language and several modern software design patterns (Gamma, Johnson & Vlissides 1995) are adopted to develop and manage different PIC simulation schemes within one simulation package.

There are many PIC simulation packages developed by using C++, for example, VORPAL (Nieter & Cary 2004) and iPIC3d (Markidis, Lapenta & Rizwan-uddin 2010). The VORPAL package is comprehensive with many features, for example, it uses scripts, load rebalancing in parallel computations, a template for arbitrary dimensions and it has a self-consistent visualization package. It includes particle and fluid solvers, developed by using other fundamental and mature libraries, and it can simulate an arbitrary-dimensional plasma. Common data structures and algorithms (e.g. interpolation schemes) are stored in the base framework class. VORPAL is sophisticated and has been well tested, for example, it has been used to study the generation of laser wake fields and the optical injection of electron beams into these wake fields (Giacone 2002). Another popular PIC implementation is iPIC3D which solves Maxwell's equations by using implicit schemes with high accuracy in time. Essentially, it simulates plasma flows with both fast-changing electric and magnetic fields. Our package is quite different from iPIC3D but closer to VORPAL, it has different open architecture designs and is optimized for electrostatic simulations at this stage.

In this work, a C++ class named the 'simulation engine' is developed (Liu, Cai & Zou 2012). It is introduced to organize different PIC related simulation schemes which can simulate different types of plasma flows. Essentially, this simulation engine class has the important 'singleton' (Gamma *et al.* 1995) design pattern, which means only one object of this class is created in the package. This object is an abstract root class for all simulation schemes, serving the purpose of organizing difference classes. All the other simulation classes are daughter classes derived from this root class. Among these daughter classes, there are inherent internal relations. Figure 1 illustrates different simulation engine classes, or different plasma simulation schemes. This arrangement is effective, for example, it allows two or multiple developers to derive different classes for two different projects. As such, their work is independent, neither endangers/pollutes the basic class nor interferes with their colleagues' classes. After they complete their projects, useful member functions and features in their classes can be summarized by their supervisor, incorporated or upgraded into the member functions in the mother classes. Evidently, the code architecture is open and convenient.

Benefiting from this simulation engine concept, the package can conveniently include many PIC simulation schemes. Examples include (i) three-dimensional plasma flows with Cartesian geometries; (ii) hybrid PIC schemes with the direct simulation

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

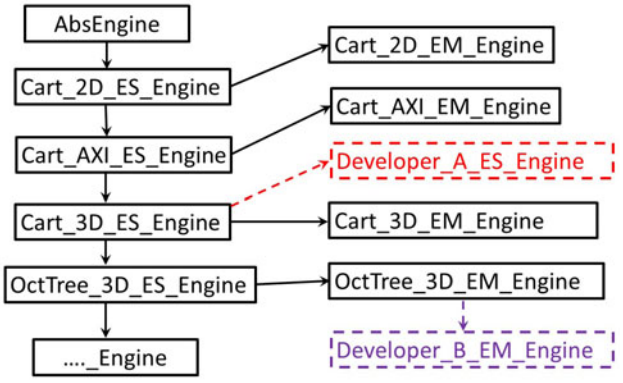


FIGURE 1. Illustrations of the inheritance relations among different simulation engine classes.

(*a*)

```
class AbsEngine
{
  public:
      inline virtual void Potential_Solver()=0;
      inline virtual void Particle_Move()=0;
     ..............
  private:
      .........
}
```

(*b*)

```
....; AbsEngine *Engine_PTR =NULL;   ....;
int main(int argc, char *argv[])
{   .....;
      if(mode ==1)
            Engine_PTR = new Cart_2D_ES_Engine();
    else if(mode ==2)
            Engine_PTR = new Cart_AXI_ES_Engine();
    else if(mode ==3)
            Engine_PTR = new Cart_3D_ES_Engine();
    ....;
   {   ...;   Engine_PTR→Particle_Move();   .....;
   }
    .......;
  return 0;
}
```

FIGURE 2. (*a*) Illustrations for a root class with pure virtual member functions; (*b*) the main code with the engine classes and objects.

Monte Carlo method (Bird 1994) for plasmas with neutral gaseous species; (iii) full PIC schemes for electrostatic plasmas; (iv) full PIC schemes for electro-magnetic (EM) plasma flows. Theoretically, the package can include more functionalities without any difficulty. All these schemes have common functionalities, and they can be driven with a singleton object initialized from the abstract engine class.

All simulations performed with this simulation package are controlled by a file that includes many key parameters. This approach is the same as many standard commercial finite element analysis solvers, like NASTRAN (MacNeal 1972), for computer aided engineering. These cards are scanned periodically while simulations continue, e.g. every 500 hundred steps. As such, users can modify the simulation parameters on the fly, there is no need to stop to read in new parameters and then restart the simulation from scratch again. The package writes out intermediate simulation results periodically and can restore the simulations by reading back the saved information.

Another novel feature of this root simulation engine class is that it adopts many pure virtual member functions that can effectively separate member functions' names and interfaces. Two examples of such pure virtual member functions are shown in figure 2 with the unique programming grammar '=0'. As is well known, a standard class member function usually has four crucial elements: the function name, its parameter list, its function body and the return result type. However, a pure virtual member function cannot have a concrete function body. The sole purpose of such

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

a pure virtual function is to define an interface for that member function in the mother and daughter classes. Those pure virtual functions for this abstract class of simulation engines are never called; instead, the corresponding immediate and derived daughter class' true member function with a concrete function body is called during simulations. Figures 1 and 2(*a*) indicate the first daughter class 'Cart_2D_ES_Engine' derived from the root class with pure virtual functions must have a member function 'Potential_Solver()' with a concrete function body. All the other daughter classes from this 'Cart_2D_ES_Engine' may not need to implement a different member function body. If necessary, different member functions can be overridden inside a daughter class. A common issue for software packages with multiple schemes is the mixture of function interfaces and bodies. This implementation of pure virtual member functions efficiently resolves this issue, the interface is concise, and the corresponding maintenance work is convenient.

Figure 2(*b*) shows the main code which includes common interfaces shared by all daughter simulation engines. In one simulation, only one object for a specific daughter simulation engine class will be created, depending on a specific control card in the input file.

2.2. *ES-PIC loop and data structure*

A typical ES-PIC simulation procedure includes the following steps:

(i) select asimulation type, for instance, three-dimensional full kinetic ES-PIC;
(ii) choose whether to perform simulations on a serial or parallel platform, the latter needs to specify how many processors to run;
(iii) provide the mesh sizes and domain lengths along the X−, Y− and Z− directions;
(iv) specify other simulation parameters, e.g. inflow conditions, external fields, species information, time step size, boundary conditions, etc.;
(v) initialize the flow field;
(vi) compute the initial field(s);
(vii) enter the major loop and run for the pre-determined number of time steps; and
(viii) post-process.

A package user can conveniently change a simulation by modifying the specific card in the input file; hence, even a user without much knowledge of PIC schemes can perform a simulation.

The data structure takes advantage of Cartesian mesh grids to separate particles and field data. Particle positions and velocities are stored in a single linked list for each species to allow continuous addition and removal of particles. The grid is Cartesian and it is straightforward to determine which cell a particle is located in. Field data are stored in an *IJK* ordered array for fast access. Special gather and deposit functions are implemented to mediate information between the two data structures. The data structures and main loop are illustrated in figure 3(*a*).

The main ES-PIC scheme class is derived from a standard PIC scheme class. With initial fields calculated, the code injects particles of different species into the simulation domain at the injection boundaries, where particle velocities are rewound by a half-time step to utilize the leap-frog scheme. The particle velocities and positions are advanced by a time step with the Boris pusher (Boris 1970), and the advancements are based on the gathered local field values. The gathering process uses a second-order volume-weighting algorithm (Birdsall & Langdon 1991). The particle

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

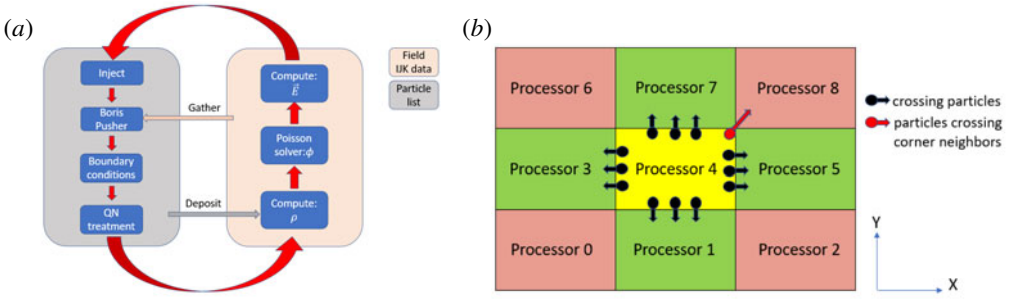


FIGURE 3. (*a*) Main loop and data structures for the simulation package; (*b*) two-dimensional illustration for transferring particles among sub-domains handled by different processors.

pusher has three modes: non-relativistic, relativistic and mixed. Aiming to save computation cost, only the particles that have high enough speeds will be pushed with a relativistic effect. After being pushed, particles interact with boundaries if they cross them. Then the quasi-neutral module is followed, where a user can select boundary treatments to ensure charge continuity in the system. This step is crucial to the stability of a full kinetic PIC simulation. By default, this module does almost nothing, but a detailed discussion will be presented in the next section with test cases. In the next step, the particles allocate their charges to adjacent grid nodes by using the same weighting scheme as the previously mentioned gathering process. The final step in the main loop is a Poisson equation solver, which utilizes the grid charge information to solve for the potential fields. The performance of the Poisson equation solver is crucial to an ES-PIC simulation because it is called in every step. The code adopts the conjugate gradient method (Golub, Charles & Loan 2012) for both serial and parallel computation modes. For the serial mode, the Poisson equation solver is pre-conditioned by using incomplete Cholesky factorization with zero fill (IC0). The parallel implementation of the solver is to be presented and discussed in the next section.

2.3. *Parallel computation schemes*

One issue which must be addressed is to choose the proper parallel scheme to implement the PIC scheme. One approach is to allocate particles evenly among multiple processors, regardless of their positions, to ensure balanced loads. However, because particles need local field information, which may be stored by another processor, the related communication cost is high. Another more popular approach divides the physical simulation domain into many sub-domains to be handled by different processors. Each processor handles a specific sub-domain field, which is further used to push particles located in this sub-domain. The second approach is adopted in this package for parallel computations because it is straightforward to implement with superior performance (Blahovec *et al.* 2000; Wolfheimer, Gjonaj & Weiland 2006). A defect of this approach is that particles may travel across different or even multiple sub-domains, thus transferring particles among processors is needed. The package adopts the message passing interface (MPI) method to communicate messages among processors, due to the good scalability and availability in most high performance computation facilities.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

The parallel ES-PIC loop treatments are the same as those for a serial computation mode except for two issues that need special attention. One issue is particle movements. Particles entering the neighbouring sub-domains must be handed over to another processor. The other issue is about the Poisson equation solver, where ghost layer information must be communicated during each iteration. Those information exchanges across different sub-domains are handled by different processors. The treatments for these two issues are discussed in what follows.

2.3.1. *Transferring particles among sub-domains handled by different processors*

Transferring particles among different processors can be achieved by using the MPI functions of 'send' and 'receive'. For a three-dimensional scenario, usually crossing-domain particles stop in one of their six neighbouring sub-domains, each shares one face with the current sub-domain. To properly transfer those particles into those sub-domains is not challenging. However, a few particles may re-locate into a sub-domain which only shares two edges or only a vertex with this specific initial sub-domain. It is challenging to treat those particles.

We can use a two-dimensional (2-D) scenario to help discussions. As shown by figure 3(*b*), there are 9 sub-domains handled by 9 different processors. A sub-domain (handled by Processor 4) has four side-neighbouring sub-domains (marked in green) with one shared edge, and the majority of particles move towards these four neighbouring sub-domains (this sub-domain also receives particles from these four neighbouring sub-domains). Actions of sending and receiving these particles are not challenging because they only involve communications with the four neighbours. However, there are also a few of particles that may travel to the four corner sub-domains (marked in pink). If we exchange particles with these corner-neighbour sub-domains as well, there will be a total of 8 neighbours to be considered in a 2-D case. For a 3-D scenario with 26 neighbours, such an option is less attractive. Another option is to simply send and receive particles with its 4 neighbours, then repeat the process until all crossing-domain particles reach their proper destination sub-domain. For example, the red particle is supposed to move from Processor 4 to Processor 8; however, if we simply transfer with the 4 green neighbours, it ends up in Processor 5 or 7. Then we must repeat the process again to complete the transfers from Processor 7 to 8. This solution is not ideal because it requires multiple transfer steps.

A three-step transfer scheme for a 3-D scenario is designed in this package to simplify this process, and it can maximize the transfer efficiency. This scheme transfers particles only along the main mesh directions (with 6 neighbours in three dimensions and 4 neighbours in two dimensions), but some minor manipulations are needed during the transfer process along each direction. In this scheme, transfer functions are only called once along the X, Y and Z directions. First, particles are checked to determine whether they need to cross the Z, Y or X sub-domain boundaries (pay attention to the specific orders for the transfer directions) after being pushed to their new positions. If so, they are copied to the corresponding transfer buffer: Z−, Z+; Y−, Y+; and X−, X+. The plus and minus signs indicate which neighbour along that direction to transfer particles to and receive particles from. For a particle crossing multiple boundaries, it will only be added to one transfer buffer to ensure no duplication of that particle. After all crossing-domain particles are sorted into one of the six transfer buffers, a transfer function along the X− direction is called. As the first step, this function starts the procedures of sending and receiving particles along this direction, and as the second step, the received particles

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

are further checked if they are indeed inside this sub-domain. If so, they are added to the linked-list particles for this sub-domain; if not, they are checked to determine whether they are outside of the Z− or the Y− boundaries (it is impossible to receive a particle which is outside of the X− boundaries here); and then they are added to the Z−, Z+ or Y−, Y+ transfer buffer (again pay attention to the orders for the transfer directions). Next, the transfer function along the Y+ direction is called, similar to the function along X, except, here, received particles are only checked to confirm whether they are outside of the Z boundaries; if so, then they are added to the Z− or the Z+ list. In the end, the transfer function long Z− is called, and there is no need to check the received particles.

Figure 3(*b*) is used again to explain why we add the crossing particles to the transfer buffers in a reverse order, first Y then X, not first X then Y. For the 2-D scenario, the crossing particles should be added in the orders of Y and X, followed by calling transfer functions along X, then Y. Then we consider the red particle again and see what will happen if we do not follow this order. If we add this particle in the first X then Y order (instead of the first Y then X), then because it crosses both boundaries, it will eventually be added to the Y+ transfer buffer. We emphasize again we do not copy this particle to both the X and Y buffers to ensure there is no duplication of this particle. Then in the transfer function along X, it will not be transferred to any of its X neighbours because it is not in the X+ or X− buffer. Next, in the transfer function along Y, it will be transferred to Processor 7, which is not its new destination sub-domain. If we follow the prescribed order Y and X, it will be added to X+ transfer buffer, and transferred to Processor 5 in the transfer function along X. After receiving this particle, Processor 5 will check whether this particle is outside of its Y boundary, then subsequently add it to the Y+ buffer. In the next sub-step, Processor 5 will transfer this particle to Processor 8 in the transfer function along Y. Such a scheme dramatically simplifies the particle transfer algorithm and is convenient to implement. Additionally, because a particle is only added to one transfer buffer at one time, it cannot be deleted or duplicated accidentally. As explained, this process reduces the debugging effort.

2.3.2. *A Poisson equation solver with multiple processors*

The Poisson equation solver's performance is crucial to a parallel ES-PIC code. The conjugate gradient (CG) method is popular for parallel computation since it involves many vector computations and matrix multiplications. Nevertheless, choosing a good parallel pre-conditioner is not trivial. The IC(0) pre-conditioner for example (Huang *et al.* 2009) is usually adopted for serial computations, and it has superior performance for serial as compared to parallel computations. Several different parallel pre-conditioners have been tested for better performance, and the factorized sparse approximate inverse (FSAI) (Kolotilina, Yu & Yeremin 1993) pre-conditioner is adopted in this package due to two reasons. First, once an FSAI pre-conditioner is constructed, it can be used repeatedly through the whole simulation. This treatment is suitable for the ES-PIC scheme because the Poisson equation only updates its right-hand side term within each time step. Secondly, the FSAI pre-conditioner has shown good parallel computing performance for a symmetric positive matrix (Benzi & Tuma 1999), and the Poisson equation usually leads to such a matrix. For a parallel Poisson equation solver, information exchanges with neighbouring sub-domains are crucial. The CG method involves many iterations before a converged solution can be obtained, resulting in multiple information exchanges between neighbouring sub-domains during each time step. Due to this constant synchronization during the

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

solving the potential equation step, the parallel ES-PIC mode does not benefit much from dynamic load balancing adopted in most electro-magnetic PIC codes. It is worth mentioning that, in the current package, a Poisson equation solver is implemented with GPU and written in CUDA. This treatment can help with future development.

Other novel features of this package include: (i) it may re-load particles and re-start unfinished simulations; (ii) different boundary condition treatments: reflective, periodic, injection, outflow, current return, sinks, conducting walls and dielectric; (iii) multiple diagnostic tools; (iv) multiple output file formats (e.g. DAT and VTI) that can be written in parallel; (v) a resizeable array class that can automatically handle particle storage issue. VORPAL developed a post-processing and visualization module so it can be self-consistent. To minimize the development work, we adopt the freeware package ParaView (Ahrens, Geveci & Law 2005; Ayachit 2015) for visualization. To regulate an arbitrary dimension simulation, VORPAL adopted templates that need sophisticated treatments, e.g. on particle weighting and movement. In our implementations, for simplicity, all those functions are directly incorporated into related simulation engine classes.

3. Benchmark test cases and discussion

We perform four full kinetic ES-PIC simulation cases related to positively charged probes immersed in plasma. The first two cases are used to validate the code and its implementation. Case I deals with non-drifting plasmas with a magnetic field, and Case II involves space charges collected by a positively charged probe without a magnetic field. These two problems have theoretical results and are suitable for code validation. Cases III and IV are about numerical investigations of practical plasma flow problems, and they are associated with more complicated $\boldsymbol{E} \times \boldsymbol{B}$ drifting plasma flows. These four cases correspond to the work done by Singh, Vashi & Leung (1994), Laframboise (1966), Singh, Leung & Vashi (1997) and Heinrich & Cooke (2013).

This type of problem has a long history of research and it is quite challenging to analyse theoretically. Laframboise & Sonmor (1993) provided a detailed review on the subject. In numerical studies, both electrons and ions must be treated as particles with electron kinetic theory; as results, computational costs increase due to the low electron mass. Another issue with full kinetic ES-PIC simulations is the difficulty in keeping the simulation domain quasi-neutral, and this issue has been discussed by Singh *et al.* (1997), Onishi (1998), Pritchett (2000) and Brieda (2018).

These four plasma flows happen within different regimes with different flow patterns. All cases have small plasma beta, i.e. the induced magnetic field is much smaller than the external magnetic field; thus, the use of the ES-PIC simulation method is well justified. They are confirmed by using finer grids, smaller macro-particle to real particle ratios, larger simulation domain sizes and smaller time steps. Since these problems are electrostatic, we adopted the following stability criteria for grid size and time step choices: $\Delta x \leqslant \lambda_D$ and $\Delta t < 0.2\omega_p$ for all the four simulation cases. Here, λ_D and ω_p represent the Debye length and electron plasma frequency, respectively. In practice, our choices of time steps are stricter than the criteria above. This section discusses the first two validation cases.

3.1. *Test Case I: a positive probe in non-drifting magnetized plasma*

We first perform a series of simulations with a positive probe immersed in non-drifting magnetized plasmas. The probe potential ranges from 5 to 40 V. We then compare

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

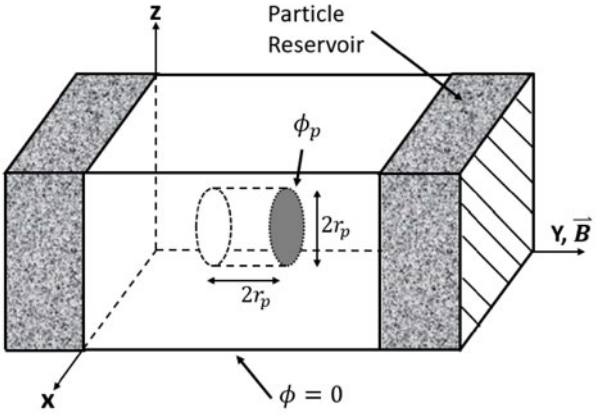


FIGURE 4. Test Case I: simulation set-up. Domain size: NX = NZ = 157, NY = 221, where NX, NY, NZ denote the number of nodes along each axis; mesh size $\Delta x = \Delta y = \Delta z =$ 0.0105 m, approximately one Debye length.

the simulated current collection with the values predicted by the Parker & Murphy (1967) theory. Parker and Murphy considered the conservation of canonical angular momentum of an electron in a cylindrically symmetric force field, and derived an upper limit for the collected current by the positive probe

$$I_{PM} = 2\pi {r_0}^2 j_0, \tag{3.1}$$

where r_0 is the maximum radius from which current can be collected, and j_0 is the electron thermal current in the plasma. The relation between r_p and r_0 is given as

$$r_0^2/{r_p}^2 \leqslant 1 + (8\Phi_0/(m_e {\omega_e}^2 {r_p}^2))^{0.5}, \tag{3.2}$$

where Φ_0, m_e and ω_e denote electron potential energy, electron mass and electron cyclotron frequency respectively.

The simulation set-up is illustrated in figure 4. The following plasma parameters are used in this test case: ambient plasma density $n_0 = 1 \times 10^{11}$ m^{-3}, the ion to electron mass ratio $m_i/m_e = 100$, magnetic field $B_y = 0.3$ G, the ratio of probe radius to Debye length $r_p/\lambda_D = 15$, electron and ion temperatures $T_e = T_i = 0.2$ eV. With these parameters, the electron and ion gyro-radii are approximately $0.34r_p$ and $3.38r_p$, respectively. A small ion to electron mass ratio is used to accelerate the simulations. Additional tests are performed with a bigger ion to electron mass ratio, while the electron gyro-radius is the same. These tests confirmed that the smaller mass ratio treatment does not change the collected currents.

Test Case I uses similar parameters and treatments to the work done by Singh *et al.* (1994). Singh placed particle reservoirs at each magnetic end of the simulation domain. As such, electrons and ions can move into the simulation box; there is no need to compute the precise entrance angles. The reservoir contains equal numbers of ions and electrons, both with Maxwellian velocity distributions. At the beginning of each time step, randomly positioned particles leave the reservoir and enter the simulation box. After each time step, the reservoir is randomly refreshed to maintain enough particle sources. The reservoirs are connected to two of the cubic simulation

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

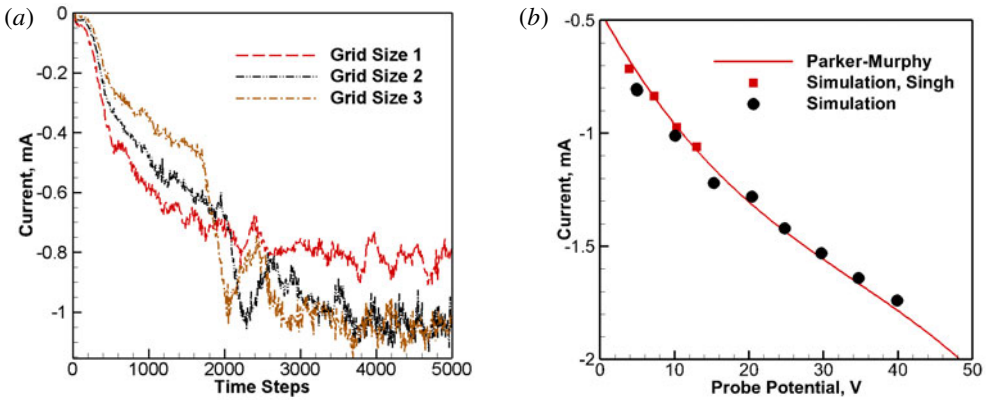


FIGURE 5. Test Case I: (*a*) probe current collection history for the case of 10 V probe surface potential; (*b*) the I–V relation predicted by the Parker–Murphy theory and simulations.

box faces, whose normal directions are parallel to the magnetic field. As such, the entering electrons can move along the magnetic field lines and fill the simulation box quickly. It is feasible to put reservoirs on the remaining four simulation domain faces as well, but such a treatment can increase the computational cost. Boundary treatments are needed to preserve electrons exiting the boundary faces that have no reservoirs attached. This is because electrons have gyro-motions and can exit and re-enter the boundary faces frequently. In the computation, an exiting electron is simply deleted from the simulation if no treatment is performed. This can be improved by setting those boundaries to be reflective, or simply by re-injecting those escaped electrons into the simulation domain.

The simulation was performed with 96 processors, 5000 time steps and lasted approximately 34 min. Figure 5(*a*) shows the current collecting history for the 10 V probe potential case. Three different grid sizes were used to confirm grid convergence, where Grid Set 2 with a mesh size 0.0105 m is adopted for our final simulations, while Grid Sets 1 and 3 are coarser and finer, respectively. The time-averaged current value in the steady state is approximately 1.1 mA, very close to the value of 0.980 mA which is predicted by the Parker–Murphy theory. Figure 5(*b*) shows the current–voltage curve over a range of different probe potentials. The solid line is from asymptotic values predicted by the Parker–Murphy theory. The triangles represent our PIC simulation results, and the squares represent the results obtained by Singh. It should be mentioned that, in Singh's work, they used a mesh in a cylindrical coordinate system, while in our work, we used a cubic mesh. The former can fit a spherical probe surface with higher accuracy, and we believe this is the reason why their results have relatively better agreement compared to our results. However, as shown, the good agreement among the three sets of data indicates that our PIC package implementations are still reliable.

3.2. *Test Case II: unmagnetized space charges collected by a spherical probe*

In this test case, we use the theoretical result by Laframboise (1966) to validate our code. Laframboise's theoretical study solves the Poisson–Vlasov equation for electrons. The method computes the electron densities at special points in the distribution

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

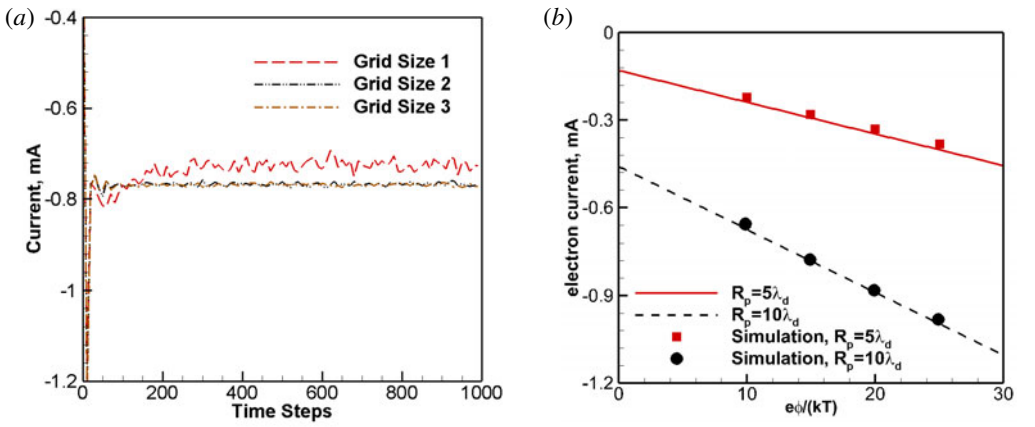


FIGURE 6. Test Case II: (*a*) electron current development history for probe $R_p = 10\lambda_D$, $e\Phi/kT = 15$ and $T_i/T_e = 1.0$; (*b*) electron current collections: simulation (symbols) and Laframboise's theoretical (lines) results.

function, and this distribution function is determined by the conservation of angular momentum and energy. A nonlinear system of integral equations were derived and solved numerically to obtain the current collections by the positive probe. Unlike the first test case, there is no magnetic field, and the probe shape is spherical. The same as Case I, our simulation domain adopts cubes. The plasma parameters are the same for the previous test case except the probe radius R_p are used. We perform two series of simulations, with $r_p/\lambda_D = 5$ and 10. The simulation technique is similar to Case I except the reservoirs are attached to every boundary face. This is because without a magnetic field, particle collections do not have favourable directions.

The simulation is performed with 96 processors, 1000 time steps and lasts approximately 9 min. Figure 6(*a*) shows the electron current collection history for one series of simulations with $R_p = 10\lambda_D$. Three different grid sizes were used to show grid convergence. Grid Set 2 with a size of 0.0105 m is chosen for simulations, while Grid Sets 1 and 3 have double or half the size. As seen from figure 6(*a*), the simulation reaches steady states relatively faster with smaller fluctuations when compared with Case I, which has an external magnetic field. Figure 6(*b*) shows that our simulation results agree well with Laframboise's theoretical results. Since our code adopts Cartesian meshes, the spherical probe is approximated by cubes. A total of 157 mesh points are used along the X, Y and Z directions and the mesh sizes are close to one Debye length. This explains why, in figure 6(*b*), the results for the larger probe scenario, $R_p = 10\lambda_D$, show better agreement than the smaller probe case: the spherical probe is better approximated by using more mesh points.

4. Positive probes in $E \times B$ drifting magnetized plasma

Cases III and IV are about plasma flows with $E \times B$ drifting velocities, and they are the major goals of our PIC package development work. Essentially, all the simulation parameters for these two cases are the same as those in the past papers, and for convenience, these parameters are copied in this paper. This section includes the results and discussions of the two cases and provides implementation recommendations on maintaining system quasi-neutrality.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

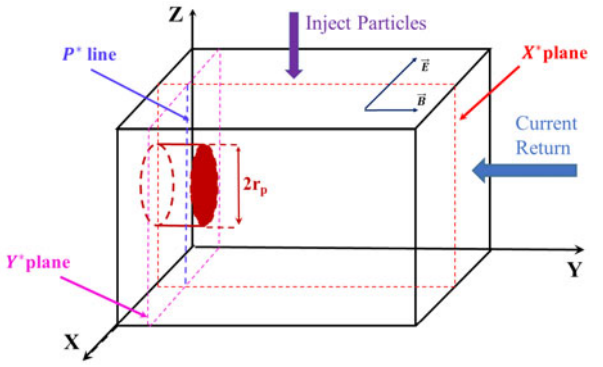


FIGURE 7. Case III: simulation domain illustration. Domain size: NX = NZ = 157, NY = 289, where NX, NY, NZ denote the number of nodes along each axis; mesh size $\Delta x = \Delta y = \Delta z = 0.01$ m.

$n_e = n_i$ (m^{-3})	1×10^{11}
$T_e = T_i$ (eV)	0.2
λ_D (m)	0.0105
r_e (m)	$\approx 5\lambda_D$
r_i (m)	$\approx 215\lambda_D$
r_p (m)	$\approx 15\lambda_D$
m_i/m_e	1836
B_y (G)	0.3
E_x (V m^{-1})	−1.76
ω_{pe} (10^7 rad s^{-1})	1.78
ω_{ce} (10^6 rad s^{-1})	5.28
U_{drift} (10^4 m s^{-1})	5.76
V_p (V)	10

TABLE 1. Simulation parameters for Case III (Singh *et al.* 1997).

4.1. *Case III: a positive probe in an* $\boldsymbol{E} \times \boldsymbol{B}$ *drifting magnetized plasma*

Case III was investigated by Singh *et al.* (1997). The simulation parameters are listed in table 1, and figure 7 illustrates the simulation set-up. Also shown in the same figure, the external magnetic field is along the Y-axis with a magnitude of 0.3 G, and the external electric field along the negative X direction is set to be 1.728 V m^{-1}. A relative motion of $0.3V_{th,e}$ or $13V_{th,i}$ between the plasma and simulation box is created, where $V_{th,e}$ and $V_{th,i}$ are the electron and ion thermal velocities respectively. This flow regime is characterized by $r_e < r_p$, $r_i \gg r_p$ and $\omega_{pe} > \omega_{ce}$, where r_e, r_i, r_p, ω_{pe} and ω_{ce} are gyro-radii of electrons and ions, probe radius, plasma frequencies and electron gyro-frequencies respectively. Ions in this flow regime are effectively un-magnetized due to the large gyro-radius to probe radius ratio.

For convenience, two special planes and one line are defined to aid result discussions for the last two test cases. The X^* plane is defined as the plane with a fixed X value, it is normal to the X-axis and passes the probe centreline. The Y^* plane is defined as the one perpendicular to the probe centreline, and it evenly bisects the extruded probe, i.e. $y = R_p/2$. The $p*$ line is the intersection line formed by these two planes.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

The simulations adopted the following parameters: mesh size $\mathrm{d}x = \mathrm{d}y = \mathrm{d}z = 0.01$ m, which is approximately one Debye length; time step $\Delta t = 3.56 \times 10^{-8}$ s. The simulation begins with a plasma flow being injected from the top boundary face. It drifts downward and eventually leaves the simulation domain from the bottom boundary face. The left boundary face (i.e. $Y = 0$) connected to the probe is set up as a symmetric plane to reduce the simulation cost. A symmetric plane means that when a particle hits it from the domain, it will not be removed from the simulation; instead, it bounces back into the simulate domain, with its normal velocity components reversed but its parallel velocity components unchanged. We compare the results with those from cases without such a symmetric plane treatment and confirm that the symmetry boundary is valid and can reduce the computation cost by half. Particles can also exit from the other four boundaries and be deleted from the simulations. The electrostatic potential is updated every time step by solving Poisson's equation and the gradients are adopted to update the electric field in the simulation box. All boundary faces adopt a Dirichlet boundary condition with a zero potential, except that the left boundary face adopts a Neumann boundary condition $\partial\phi/\partial n = 0$ due to the symmetry. The probe surface potential is set to 10 V.

To sustain a stable simulation, the system neutrality requirement must be maintained. The injection introduces an equal number of electrons and singly charged ions to establish a plasma flow into the simulation box, at a rate calculated by: $U_{\mathrm{drift}} \cdot \boldsymbol{n}\, \mathrm{d}t\, \mathrm{d}s$. However, because lighter electrons have higher velocities, they usually exit the simulation boundaries with a much higher rate than ions. As consequence, the simulation boundaries change to charge positive and further exacerbate the simulations. Another source of electron depletion is from the current collecting probe. A positive probe continues to collect electrons in the simulation and causes the system to be positively charged. In this test case, we adopt Singh's treatment, where those particles leaving the boundary faces are rejected back to the simulation box boundary faces; and those electrons collected by the positive probe are returned to the system at the right boundary facing the probe end. In this case, we did not adopt the reservoir treatments (i.e. with extra buffer volumes outside the simulation domains) which were used in Cases I and II. The simulation is performed with 96 processors, 2000 time steps and lasts approximately 3 h. With those flow parameters, the plasma flow takes approximately 700 time steps to reach the bottom boundary; hence, the simulation has sufficient time to reach a steady state.

The simulation results are very close to those obtained by Singh *et al.* (1997), with several slight differences. This is due to subtle implementation details, e.g. different types of meshes.

Here, we briefly discuss the flow field characteristics and offer several implementation recommendations to simulate $\boldsymbol{E} \times \boldsymbol{B}$ drifting plasma flows around a positive probe with similar parameters. We first discuss the potential sheath structures near the probe. Figure 8 shows the potential contours inside the X^* and Y^* planes. Figure 8(*a*) shows that the sheath structures extend much further along the magnetic field direction and bend towards the wake region. Figure 8(*b*) demonstrates fan-shape structures around the probe with a negative potential wake region. These observations are consistent with the previous simulation results obtained by Singh *et al.* (1997). The negative potential in the wake is mostly due to the absence of ions in that region, and it is well illustrated by figure 9. On the other hand, electrons can enter the wake region even with the obstructions from the probe, as shown in figure 10. Both the potential structure and species density profile display asymmetry patterns in the Y^* plane, due to the presence of magnetic field.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

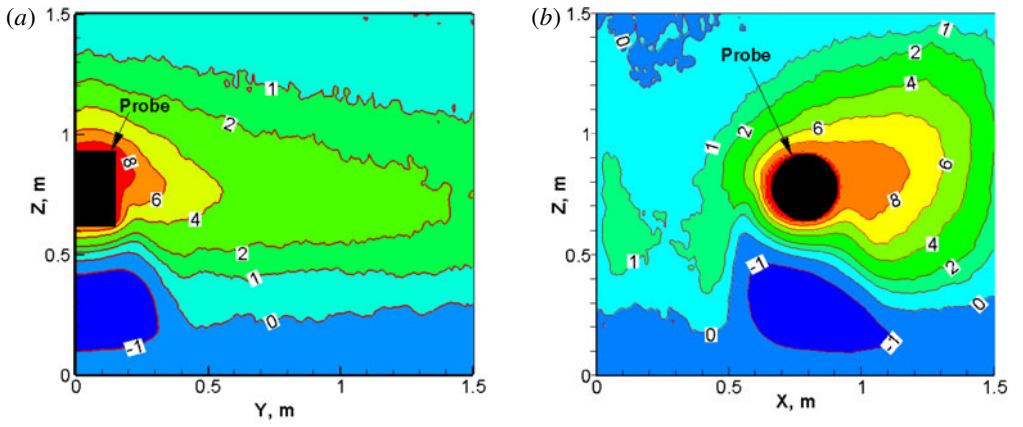


FIGURE 8. Case III: potential (units: Volt) contours in the (*a*) X^* plane; and (*b*) Y^* plane. Probe surface potential: 10 V.

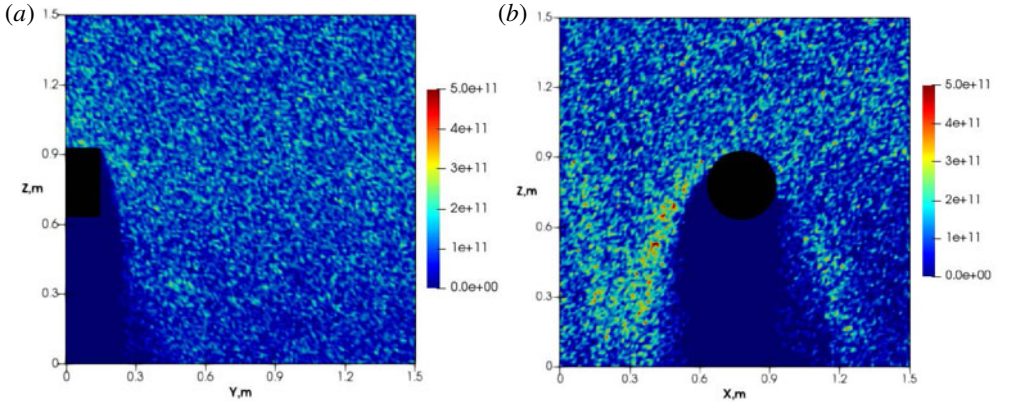


FIGURE 9. Case III: snapshots of ion number density n_i in the (*a*) X^* and (*b*) Y^* planes. Unit: m^{-3}. Ions are expelled from the probe and have difficulty entering the wake region.

Figure 8(*b*) illustrates the potential patterns, and it is evident that the lower-left region has larger gradients than the lower-right region. As such, the corresponding left wing is more effective in blocking ions and forms a slightly higher ion accumulation region than the right side. This result is correctly shown in figure 9(*b*). Due to the same reason, the left lower region of the probe can attract more electrons than the right lower region, this fact is also correctly illustrated by figure 10.

The ion and electron flow patterns are quite different. Figures 11 and 12 show the electron and ion velocity vectors inside the X^* and Y^* planes. These vectors are coloured by electric potential magnitude to help illustrate the potential sheath effects. Vector plots for electrons are averaged over 30 time steps to reduce the randomness caused by electron gyro-motions. In both planes, the ion flows are mostly unaffected by the probe and magnetic fields, which is expected because ions are not magnetized with the given simulation parameters. Essentially, ions drift downward and only divert away from locations very close to the probe with strong electric fields. The electrons, on the other hand, are strongly influenced by the probe and magnetic fields. Parallel to the magnetic fields, the electric field accelerates electrons towards

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

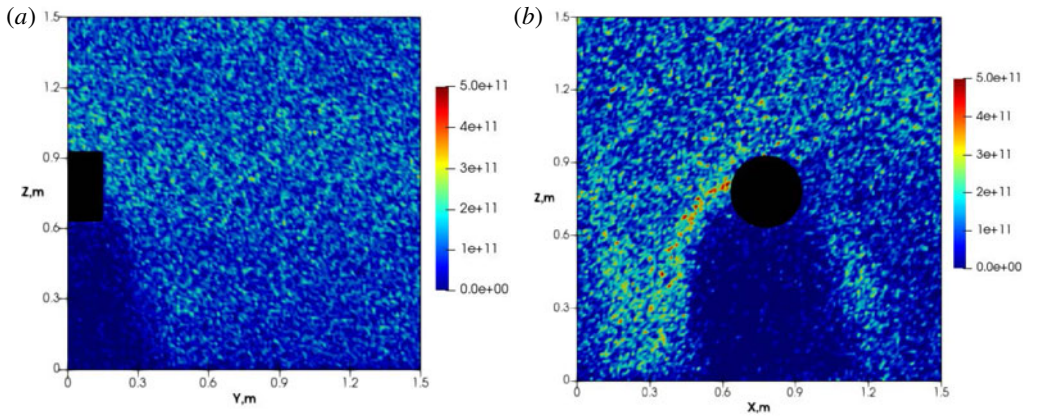


FIGURE 10. Case III: snapshots of electron number density n_e in the (*a*) X^* and (*b*) Y^* planes. Unit: m^{-3}. They are attracted to the probe and can enter the wake region.

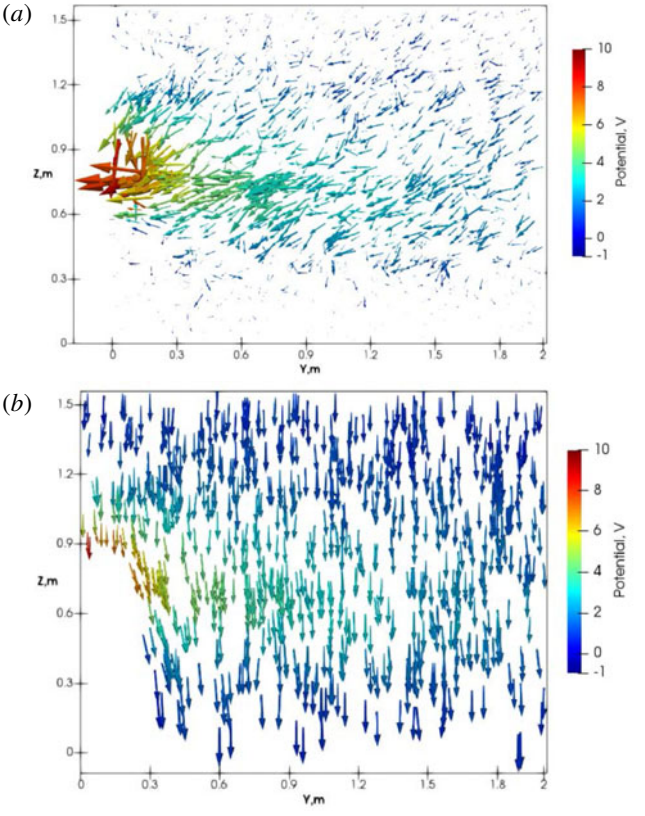


FIGURE 11. Case III: (*a*) electron velocity vectors in the X^* plane, averaged over 30 time steps; (*b*) the corresponding ion velocity vectors in the same plane.

the probe, as shown in figure 11(*a*). In the X^* plane, ions are less affected in the potential sheath region, as shown in figure 11(*b*). This explains that the extended potential sheath along the magnetic field is due electron depletion in this region.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

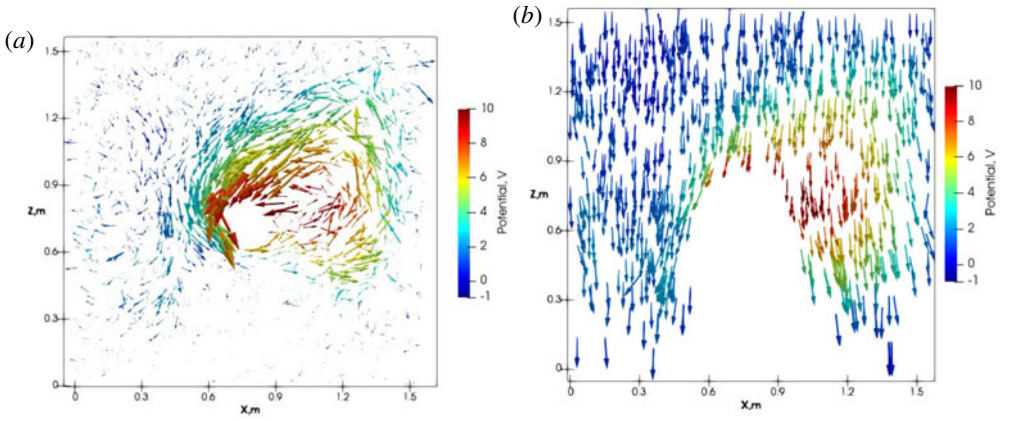


FIGURE 12. Case III: (*a*) electron velocity vectors in the Y^* plane, averaged over 30 time steps; (*b*) the corresponding ion velocity vectors in the same plane.

In the Y^* plane, figure 12(*a*) shows that electrons experience strong circulating forces around the probe, due to the local radial electric field and the resulting local $\boldsymbol{E} \times \boldsymbol{B}$ drifting effect. This circulation zone has a large cross-sectional area with asymmetrical features. Figure 12(*b*) shows the corresponding ion flow vectors in the Y^* plane. Ions have local $\boldsymbol{E} \times \boldsymbol{B}$ drift motions as well, but only near the left side of the probe. This is convenient to understand, because the external convection electric fields point to the negative X-axis direction, the local down-drift on the left side of the probe is combined with the external electric field at the same side. On the right side of the probe, however, the local drifting effects must counteract the large downward drift, and less ions stay in that area.

Particles are heated near the probe in the potential sheath region, especially electrons. Figure 13 presents the electron kinetic energy inside the two planes. In the X^* plane, electrons gain kinetic energy when they enter the potential sheath, forming a high kinetic energy zone, shown as figure 13(*a*). In the Y^* plane perpendicular to the magnetic field, there are electrons with high speeds around the probe, as shown in figure 13(*b*). Electrons around the probe gain a sufficiently high energy to circulate around the probe, counter-acting the downward drifting motions. We recall that the downward drifting velocity is around $0.3V_{th,e}$, thus it is easier for high energy electrons to move upwards even with the external downward $\boldsymbol{E} \times \boldsymbol{B}$ drifting velocity. Figure 13(*b*) also shows that high energy electrons are present near the boundary faces, at the right boundary face, i.e. the electrons circulating zone near the probe, electrons constantly deplete. These observations recommend that the simulation box should have a larger mesh size along the X direction.

Figure 14(*a*) shows three potential profiles extracted from three lines: $Z = 0.5$ m (below the probe), $Z = 1.1$ m (in the ram side) and along the probe centreline $Z = 0.0$ m, at a final quasi-steady state. A significant portion of the last profile passes through the high potential sheath region. In the ram side above the probe, the potential rises and gradually drops along the magnetic lines, as shown with the $Z = 1.1$ m line. In the wake region, where only electrons exist, the potential is low and eventually rises to the regular potential value along the magnetic field direction, as shown with the $Z = 0.5$ m line. These lines eventually reach similar potential values at a location of approximately $Y = 1.5$ m.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

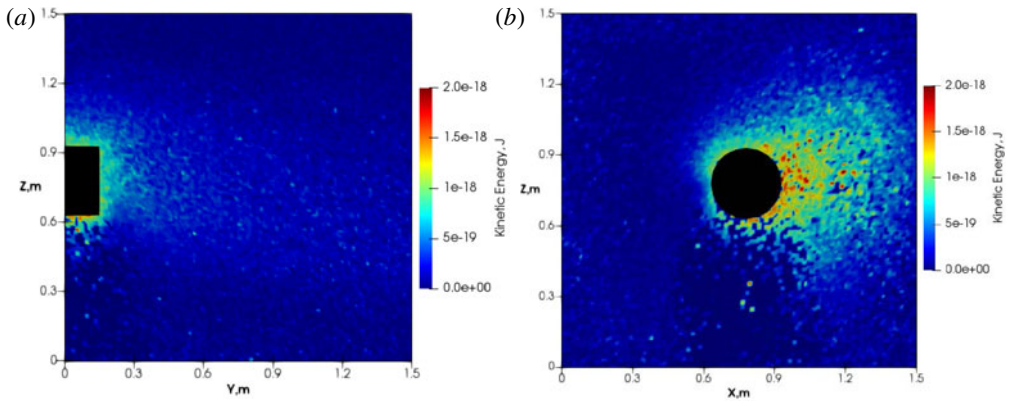


FIGURE 13. Case III: electron kinetic energy in the (*a*) X^* and the (*b*) Y^* planes.

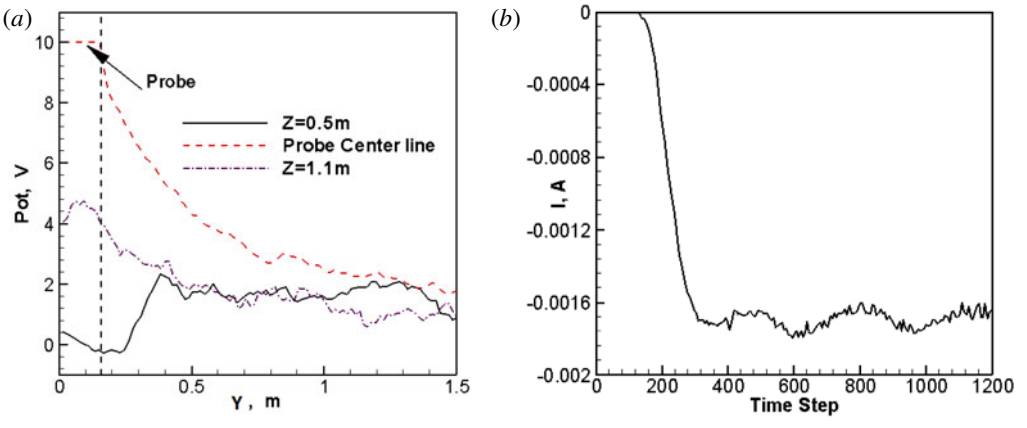


FIGURE 14. Case III: (*a*) averaged potentials extracted along $Z = 0.5$ m, probe centreline $Z = 0.0$ m and $Z = 1.1$ m, at a final quasi-steady flow state; (*b*) collected current development history.

Figure 14(*b*) shows the probe current collection development history. The current slightly oscillates but in a quasi-steady state, so it indicates that our boundary treatments are proper for establishing quasi-neutrality in the simulations. As such, the above results are reliable. The collected current is around 3.4 mA (1.7 mA × 2 since we only simulate half of the domain), which is larger than the 2.1 mA obtained from Singh's simulations. We have performed several tests with different mesh sizes, simulation domain sizes, macro–real particle ratios, etc., and all the results converge to the save value of 3.4 mA. The discrepancy may come from several subtle implementation differences. First, we adopt a uniform Cartesian grid with a mesh size smaller than one Debye length. In comparison, Singh adopted a non-uniform cylindrical mesh with a largest mesh size of 10 Debye length at the boundaries. In addition, the simulation box in our study is cubic with six boundary faces. Singh's simulation domain is a large cylinder with four boundary faces. Most likely due to these reasons, their past results are inferior. Lastly, there are many minor implementation differences, e.g. the random number generator can impact the statistical behaviours of the plasma flows.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

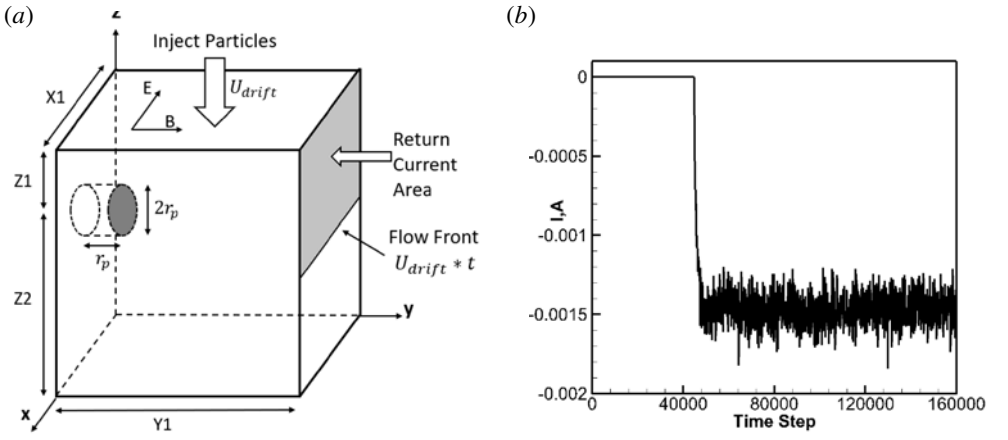


FIGURE 15. Case IV: (*a*) simulation domain illustration, $X1 = 90\lambda_D$, $Y1 = 300\lambda_D$, $Z1 = 40\lambda_D$, $Z2 = 100\lambda_D$; (*b*) probe current collection history.

$n_e = n_i$ (m^{-3})	2×10^{12}
$T_e = T_i$ (eV)	6
λ_D (m)	0.0128
$r_p = 6.6r_i = 200r_e$ (m)	$\approx 2\lambda_D$
m_i/m_e	918
B_y (T)	0.046
E_x (V m^{-1})	−9900
$\omega_{ce} = 100\omega_{pe}$ (rad s^{-1})	8×10^9
U_{drift} (10^5 m s^{-1})	2.16
V_p (V)	200

TABLE 2. Simulation parameters for test Case IV (Heinrich & Cooke 2013).

4.2. *Test Case IV: a positive probe in an* $\boldsymbol{E} \times \boldsymbol{B}$ *drifting under-dense plasma*

The last test case is the same as one past study (Heinrich & Cooke 2013). It is an under-dense plasma flow over a positively charged probe which is characterized by: $r_e < r_p$, $r_i < r_p$ and $\omega_{pe} < \omega_{ce}$. Different from Test Case III, ions are strongly magnetized in this case. Table 2 presents the flow parameters, and the simulation domain is illustrated in figure 15(*a*).

In this case, we do not adopt the reservoir treatment as used in Cases I and II. Several simulation techniques are similar to Case III, and a few of the differences are listed as follows.

Firstly, two side boundary faces connected to the symmetric plane, i.e. $X = 0$ and $X = X1$, are set to be periodic. This treatment means there is a row of positive probes, and the major goal of such a treatment is to reduce the complex boundary condition effects. Both ions and electrons have gyro-motions due to the strong magnetic field, and they exit from one of these two boundary faces and re-enter the other boundary face frequently. By setting these two boundaries to be periodic, the neutrality condition is conveniently satisfied. Neumann boundary conditions for Poisson's equation, $\partial\phi/\partial n = 0$, are adopted for all the boundary faces except for the top face. We have attempted to implement the Dirichlet boundary conditions as used

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

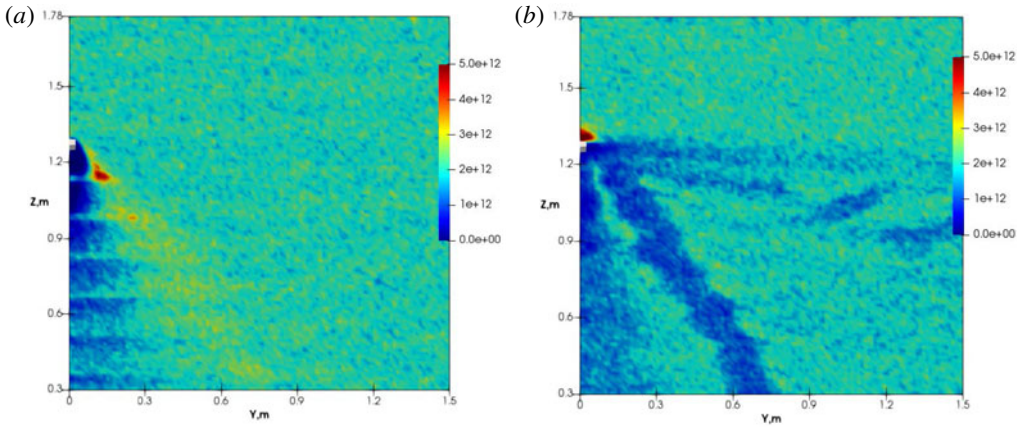


FIGURE 16. Case IV: (*a*) n_i, (*b*) n_e, in the X^* plane. Unit: m^{-3}.

in Test Case III, but the Neumann boundary conditions illustrate superior stability in this test case.

Secondly, the neutrality scheme is different from Case III. There is no particle loss at the two periodic boundary faces; thus there is no need for extra treatments. The symmetric face is still used to return the collected electrons. Because this face has a large size along the Z- direction, if the complete face area is used to return the collected particles, most electrons simply leave the bottom boundary later due to their large downward drifting motions, and this is not efficient. Instead, we allow the electron collection front to develop naturally with time. This is achieved by using a plasma front-marching surface moving downward with the drifting velocity, as shown in figure 15(*b*). The simulation is performed with 96 processors, and it lasts 39 hours with 160 000 time steps.

Figure 15(*b*) shows the current collection development history. As shown, after a certain simulation step, a relatively steady stage is reached with small current fluctuations. The collected current is approximately 3 mA (1.5 mA × 2), close to the predicted value of 4 mA by Heinrich & Cooke (2013). The small difference is because Heinrich adopted the entire right boundary face to inject electrons into the simulation box, resulting in enhanced electron collections parallel to the magnetic field. The defect of this treatment is that the simulation system may end up with more electrons than ions, and it breaks down the system quasi-neutrality with numerical errors.

Figure 16(*a*,*b*) presents the ion and electron number density distributions in the X^* plane. The positive probe is located at approximately $Z = 1.27$ m with a small diameter of 0.0254 m. Figure 16(*a*) shows that the ion flows form two distinctive patterns passing through the positive probe: a wake region and an ion-rich wing. The wing is formed due to the strong parallel electric fields near the probe, and it expels ions away from the probe. The ion-rich wing has a number density as high as 2.5 times the injecting plasma density immediately downstream of the probe, and the plasma flow gradually dissipates further downstream. Figure 16(*b*) shows the electron flow patterns in the same plane with several evident flow patterns. First, an electron trapping region appears around the probe with a number density as high as 5.5 times the injecting plasma density. Second, an electron depletion wing parallel to the magnetic fields along the Y axis emerges. Finally, a wake region and

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

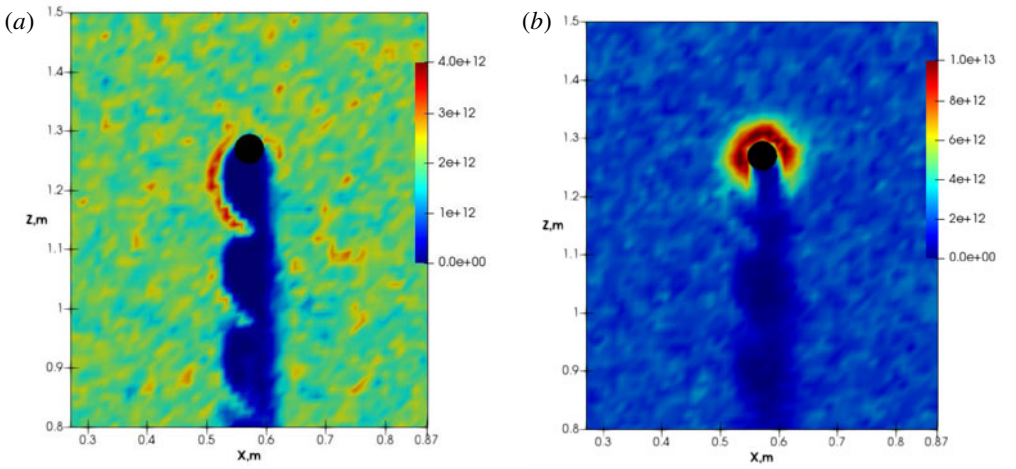


FIGURE 17. Case IV: (*a*) n_i and (*b*) n_e in the Y^* plane. Unit: m^{-3}.

another electron depletion wing are present in the ion-rich wing region. These flow patterns are consistent with the results by Heinrich & Cooke (2013), except the second electron depletion wing is in the ion-rich wing region. These discrepancies only appear in the combined charge density profile, and only net positive charges are visible in the ion-rich wing region, obscuring the presence of an electron depletion region. The related plasma physics is rather complex and we confirm their work and recommend their paper (Heinrich & Cooke 2013) for detailed explanations.

Figure 17(*a*,*b*) presents ion and electron number density profiles in the Y^* plane. For both species, the repeating wake patterns are similar, indicating that they are caused by the local $E \times B$ drifting effect. The magnetic field is along the positive Y-axis, which is perpendicular to the paper and leaving the reader. The strong local radial electric field causes both species to accumulate around the probe, counter-clockwise with the presence of the strong external magnetic field. Figure 17(*a*) further shows that the wake is asymmetric. On the left side of the probe, the wake shows evident wave patterns, while on the right side, no clear wave patterns are visible. This is due to the fact that the external convection electric field is along the X-axis direction (pointing from right to left), and the local electric field counteracts the external electric field at the right side of the probe. Figure 17(*b*) highlights electron trapping regions around the probe.

Figure 18(*a*,*b*) shows potential profiles. The high probe potential is well confined within a small region. Figure 18(*a*) shows the potential structures in the X^* plane, and the positive potential region coincides with the ion-rich wing region. The potential structure extends along the magnetic field due to electron depletion, similar to test Case III. Figure 18(*b*) shows the potential structures in the Y^* plane. Unlike Case III, with a negative potential region in the wake, the overall potential behind the probe is slightly positive, because both ions and electrons are highly magnetized and can enter this region. The asymmetry patterns are not evident in this case.

Figure 19 shows ion and electron velocity vectors in the X^* plane. The vectors are scaled by the velocity magnitudes and coloured by their number densities. Figure 19(*a*) indicates two interesting areas, one is the ion-rich wing with light yellow colours, and the other is the wake region of ion flows, pointing to the left. The ion flow in the ion-rich wing points to the right is due to the expelling effect from the positively charged

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

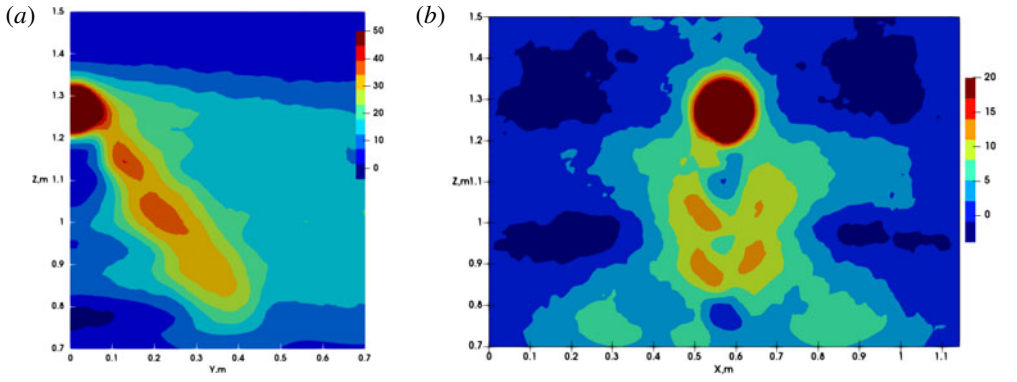


FIGURE 18. Case IV: potential contours in the (*a*) X^* and (*b*) Y^* planes. Unit: Volts.

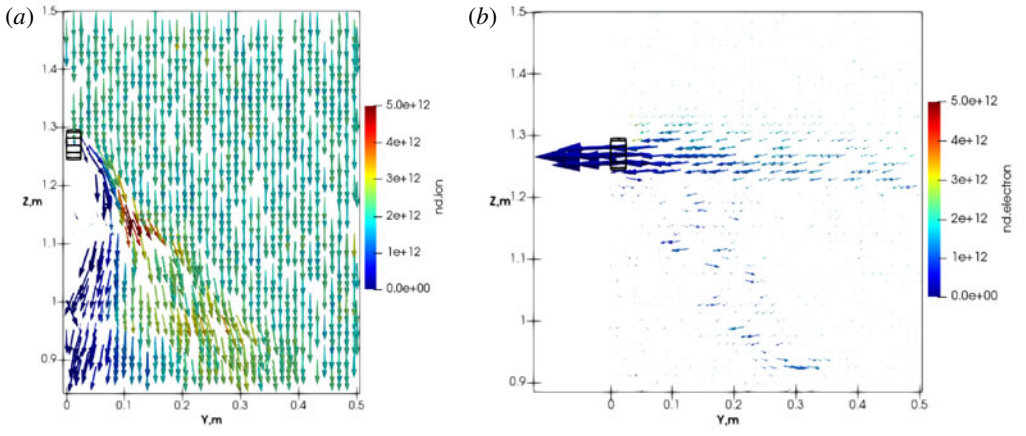


FIGURE 19. Case IV: (*a*) ion and (*b*) electron velocity vectors in the Y^* plane, coloured by their number densities.

probe. The wake region flow pointing to the left is due to the existence of the ion-rich wing, which introduces a local left-pointing parallel electric field. Figure 19(*b*) shows the corresponding electron velocity vectors in this plane. Parallel to the magnetic field, electrons are easily collected and accelerated towards the probe due to their small mass, forming a current collecting tube. Another interesting feature is, in the ion-rich wing region, electrons accelerate along the parallel direction, due to the local parallel electric field created by the ion-rich wing.

Figure 20(*a*,*b*) shows the velocity vectors for ions and electrons in the Y^* plane. Ion flows are likely affected by both the local $E \times B$ drifts and the ion gyro-motions. To better illustrate the trapping effect around the probe, velocity vectors for electrons are calculated by first subtracting the downward drifting velocity caused by the external electric fields. The trapping region has a diameter two times larger than the probe diameter, and the electron number density is higher in this region, creating a shielding effect.

Figure 21(*a*,*b*) shows the ion and electron number densities and average temperatures, along the P^* line. The grey bar represents the probe. The left side of the probe is the wake region, and the right side is the ram region. It can be seen from

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

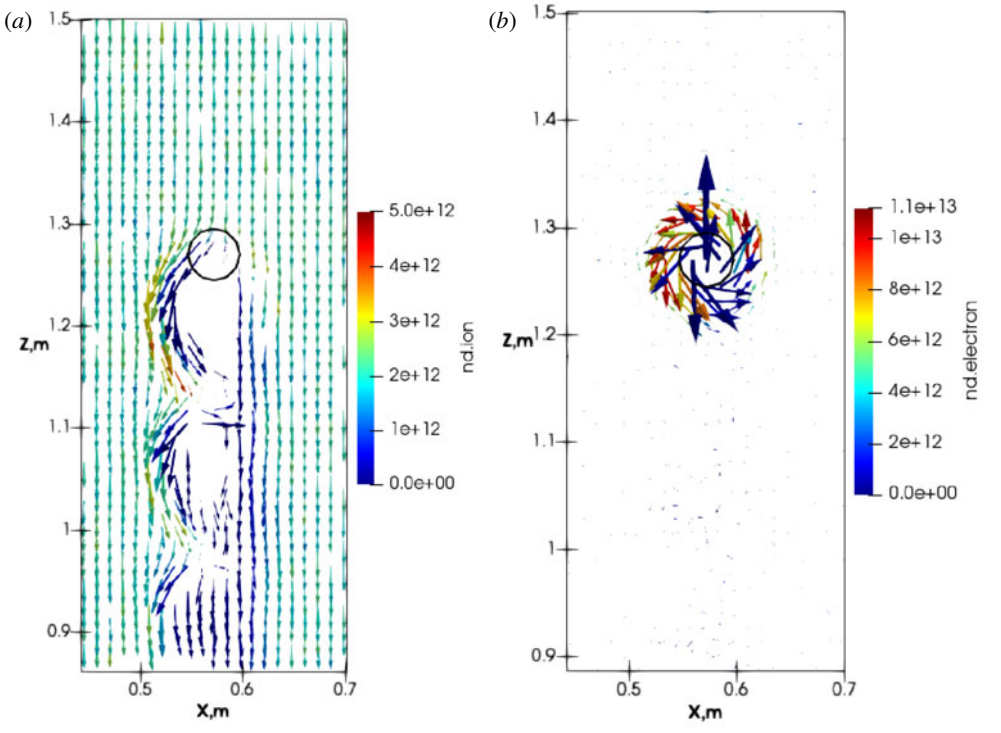


FIGURE 20. Case IV: velocity vectors for: (*a*) ions, and (*b*) electrons, in the Y^* plane. Coloured by their number densities; electron downward drift motions are removed to illustrate the trapping effect.

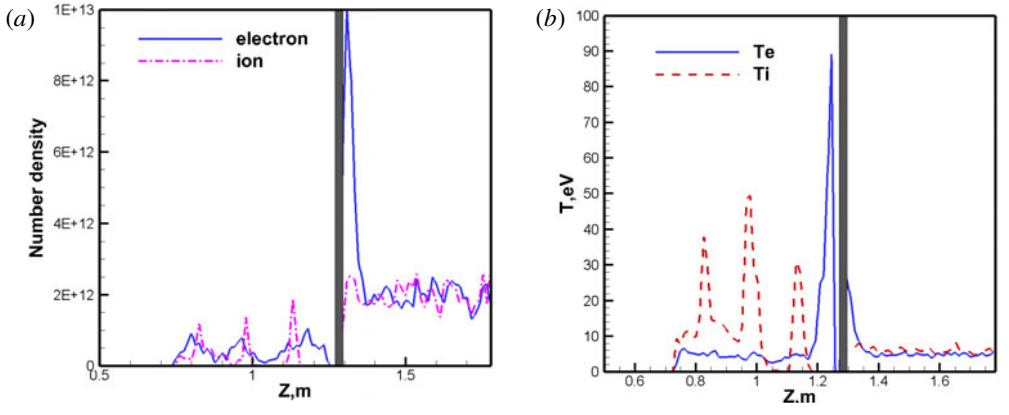


FIGURE 21. Case IV: (*a*) number density (unit: m^{-3}), and (*b*) average temperatures, along the P^* line.

figure 21(*a*) that the electron number density rises near the probe on the ram side mainly because they impinge from the top and are attracted to the positive probe but ions are expelled from the probe; the overall electron number density is much higher than the ion number density. In the probe downstream wake regions, both species have lower number densities. Figure 21(*b*) shows the corresponding ion and

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

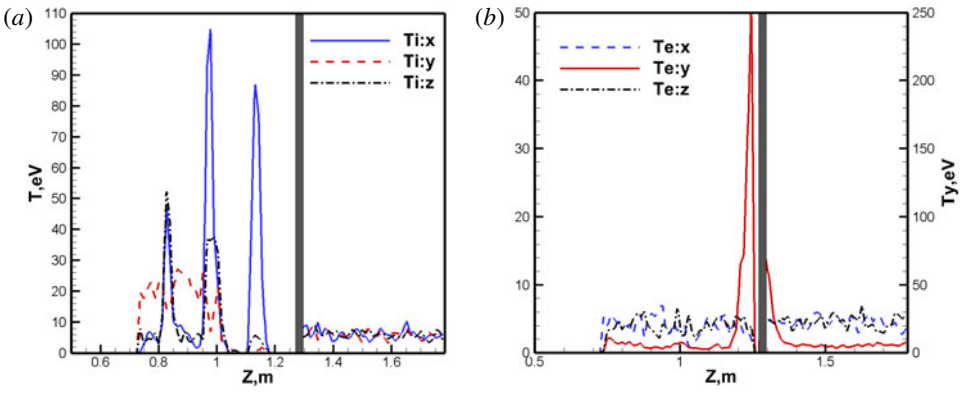


FIGURE 22. Case IV: species temperature component profiles along the P^* line: (*a*) ions, (*b*) electrons.

electron average temperatures. In the wake, electrons have higher temperatures than the ions. There are three spikes in the average ion temperature profiles, which are consistent with the spikes in the corresponding ion density shown in figure 17(*a*), and ion velocities shown in figure 20(*a*). In the wake, the highest averaged electron temperature is approximately 90 eV, which is almost twice the highest ion temperature, of approximately 50 eV).

The average temperature differences in figure 21(*b*) shall be further investigated. Figure 22(*a*,*b*) shows the detailed temperature components for each species, and they can effectively explain the T_e spikes in figure 21(*b*). Ions are heated mostly along the X− direction due to the groups of impinging ions shown in figure 19(*a*). The major contribution to the average ion temperature T_i is from T_{ix}, due to the relatively large velocity differences along the X− direction. For electrons, the temperature component T_{ey} provides the largest contribution, approximately 250 eV, to the average electron temperature. Our simulation results are consistent with the results obtained by Heinrich & Cooke (2013), and they offered good explanations for these phenomena with the Parker–Murphy theory. Essentially in the wake region, there are two groups of electrons travelling along the magnetic field lines. One group of electrons are bounced back from the symmetric plane and move away from the plane along the Y+ direction; while the other group of electrons moves towards the symmetric plane along the Y− direction. There are two big spikes in the local V_y velocity distribution function, and the velocity deviations from the averaged velocity are much larger, or the local T_y is much larger than the other two components.

In this work, numerical tests are performed by using different positive probe potential values, and a current–voltage (I–V) curve is obtained for this flow regime. Figure 23(*a*) presents the final I–V curves in semi-log scales. When plotted in log–log scales, the curve shows a linear relation, as seen in figure 23(*b*). A linear curving-fitting formula can be obtained as: $I = 10^{-0.44}\phi^{0.392}$ mA. One interesting result is obtained by examining the simulated currents with those values predicted by the Parker–Murphy theory. For drifting plasma flows, usually the measured or simulated currents can be several times larger than the theoretically predicted values. One example is the magnetic limited electron currents collected by the tethering cable during the TSS-1R experiment in space (Janeski, Scales & Hall 2014; Cooke & Katz 1998; Cai 2017). In figure 23(*c*), the Y-axis represents the simulation results,

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

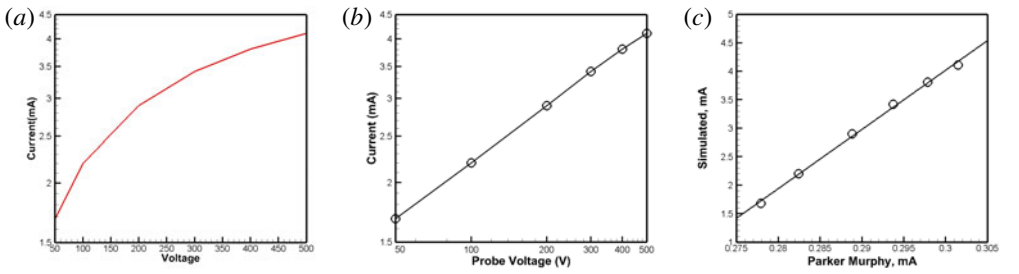


FIGURE 23. Case IV: I–V curve from simulations with a probe potential ranging from 50 to 500 V. (*a*) In semi-log scales; (*b*) in log–log scales; (*c*) simulated currents versus those from the Parker–Murphy theory.

and the X-axis represents the theoretical values predicted by the Parker–Murphy theory. A linear relation can be obtained and shown as figure 23(*c*), and the curve-fitting expression is: $I_{\text{sim}} = 103.85 I_{\text{pm}} - 27.14$ mA. It should be mentioned that those coefficients are obtained numerically. With this PIC simulation package, it is feasible to numerically study the current augment ratio. The work described in this paper provides an alternative but more accurate option than various past theoretical predictions.

5. Conclusions

In this paper, the development of a versatile serial, and parallel PIC package for plasma simulations is presented. It is written in C++ and many novel features are discussed. The package has an open architecture, and specific modern software design patterns, such as the singleton, are adopted. A pure virtual abstract class is incorporated, which separates the interfaces and function bodies for the member functions. Four cases related to positive probes immersed in plasma are studied, and physically correct results are obtained, showing good agreement with the past theoretical and simulation results. The simulation results illustrate the high efficiency of the package, and it can provide insights into simulations of plasma flows over a positively charged probe. The work can inspire PIC simulation package development.

REFERENCES

AHRENS, J., GEVECI, B. & LAW, C. 2005 *ParaView: An End-User Tool for Large Data Visualization, Visualization Handbook*. Elsevier.

AYACHIT, U. 2015 *The ParaView Guide: A Parallel Visualization Application*. Kitware.

BENZI, M. & TUMA, M. 1999 A comparative study of sparse approximate inverse pre-conditioners. *Appl. Numer. Maths* **30**, 305–340.

BIRD, G. 1994 *Molecular Gas Dynamics and Direct Simulation of Gas Flows*. Clarendon Press.

BIRDSALL, C. & LANGDON, A. 1991 *Plasma Physics Via Computer Simulation*. Taylor & Francis.

BLAHOVEC, J. D., BOWERS, L. & LUGINSLAND, J. 2000 3D ICEPIC simulations of the realistic Klystron oscillator. *IEEE Trans. Plasma Sci.* **28**, 821–829.

BORIS, J. P. 1970 The acceleration calculation from a scalar potential. *Tech. Rep.* MATT-152. Plasma Physics Laboratory, Princeton University, NJ.

BRIEDA, L. 2018 Model for steady-state fully kinetic ion beam neutralization studies. *IEEE Trans. Plasma Sci.* **46** (3), 556–562.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press

CAI, C. 2017 An improved electron pre-sheath model for TSS-1R current enhancement computation. *Aerospace* **4**, 1.

COOKE, D. & KATZ, I. 1998 TSS-1R electron currents: magnetic limited collection from a heated pre-sheath. *Geophy. Res. Lett.* **25**, 735–756.

DEROUILLA, J., BECK, A., PEREZ, F., VINCI, T., CHIARAMELLO, M., GRASSI, A., FLE, M., BOUCHARD, G., PLOTNIKOV, I., AUNAI, N. *et al.* 2017 SMILEI: a collaborative, open-source, multi-purpose particle-in-cell code for plasma simulation. *Phys. Plasma* **2**, 16. arXiv:1702.05128v1.

GAMMA, E., JOHNSON, R. & VLISSIDES, J. 1995 *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

GIACONE, R. 2002 Generation of nonlinear plasma wake fields in the colliding laser pulses injection schemes. *Bull. Am. Phys. Soc.* **47**, 282.

GOLUB, G. H., CHARLES, F. & LOAN, V. 2012 Matrix computations. *Comput. Fluids* **57**, 67–75.

HEINRICH, J. & COOKE, D. 2013 Dynamics of positive probes in under-dense, strongly magnetized, $E \times B$ drifting plasma: particle-in-cell simulations. *Phys. Plasmas* **20**, 093503.

HUANG, T., TUMA, M., ZHANG, Y., LI, L., SHAO, W. & LAI, S. 2009 Modified incomplete Cholesky factorization for solving electromagnetic scattering problems. *PERB* **13**, 41–58.

JANESKI, J., SCALES, W. & HALL, C. 2014 Investigation of the current collected by a positively biased satellite with application to electro-dynamic tethers. *J. Geophys. Res.* **119**, 7824–7840.

KOLOTILINA, L., YU, L. & YEREMIN, A. 1993 Factorized sparse approximate inverse pre-conditioning. I. Theory. *SIAM J. Matrix Anal.* A **14**, 45–58.

LAFRAMBOISE, J. 1966 Theory of spherical and cylindrical Langmuir probes in a collisionless, Maxwellian plasma at rest. PhD dissertation, University of Toronto, Canada.

LAFRAMBOISE, J. & SONMOR, L. 1993 Current collection by probes and electrodes in space magneto-plasmas: a review. *J. Geophys. Res.: Space Phys.* **98** (A1), 337–357.

LIEWER, P. & DECYK, V. 1989 A general concurrent algorithm for plasma particle-in-cell simulation codes. *J. Comput. Phys.* **85**, 302.

LIPPMAN, S., LAJOIE, J. & MOO, B. 2005 *C++ Primer*. Addison-Wesley Professional.

LIU, H., CAI, C. & ZOU, C. 2012 An object-oriented serial implementation of a DSMC simulation package. *Comput. Fluids* **57**, 67–75.

MACNEAL, R. 1972 The NASTRAN theoretical manual. *Tech. Rep.* NASA.

MARKIDIS, S., LAPENTA, G. & RIZWAN-UDDIN 2010 Multi-scale simulations of plasma with IPIC3D. *Math. Comput. Simul.* **80**, 1509–1519.

NIETER, C. & CARY, J. 2004 VORPAL: a versatile plasma simulation code. *J. Comput. Phys.* **196**, 448–473.

ONISHI, T. 1998 Electron current collection by a positively charged tether using a particle-in-cell method. PhD thesis, Massachusetts Institute of Technology.

PARKER, L. & MURPHY, B. 1967 Potential buildup on an electron-emitting ionospheric satellite. *J. Geophys. Res.* **72**, 1631.

PRITCHETT, P. L. 2000 Particle-in-cell simulations of magnetosphere electrodynamics. *IEEE Trans. Plasma Sci.* **28** (6), 1976–1990.

SINGH, N., LEUNG, W. & SINGH, G. 2000 Enhanced current collection by a positively charged spacecraft. *J. Geophys. Res.* **105** (A9), 20935–20947.

SINGH, N. L., LEUNG, W. & VASHI, B. 1997 Potential structure near a probe in a flowing magneto-plasma and current collection. *J. Geophys. Res.* **102**, 195–208.

SINGH, N. L., VASHI, B. & LEUNG, L. 1994 Threedimensional numerical simulation of current collection by a probe in a magnetized plasma. *Geophy. Res. Lett.* **21**, 833–836.

VERBONCOEUR, J., LANGDON, A. & GLADD, N. 1995 An object-oriented electromagnetic PIC code. *Comput. Phys. Commun.* **87**, 199.

WOLFHEIMER, F., GJONAJ, E. & WEILAND, T. 2006 Parallel particle-in-cell (PIC) codes. *Tech. Rep.* In Proceedings of ICAP.

https://doi.org/10.1017/S0022377820000410 Published online by Cambridge University Press