EFFICIENT AND ACCURATE PARALLEL SIMULATIONS FOR STREAMER DISCHARGE IN THREE DIMENSIONS

LIN BO

NATIONAL UNIVERSITY OF SINGAPORE

2020

EFFICIENT AND ACCURATE PARALLEL SIMULATIONS FOR STREAMER DISCHARGE IN THREE DIMENSIONS

LIN BO

(B.Sc., Nanjing University, China)

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF MATHEMATICS
NATIONAL UNIVERSITY OF SINGAPORE
2020

Supervisor:
Professor BAO Weizhu, Main Supervisor
Assistant Professor CAI Zhenning, Co-Supervisor

Examiners:
Associate Professor CHU Delin
Assistant Professor LI Qianxiao
Professor LIU Yingjie, Georgia Institute of Technology

To my parents

DECLARATION

I hereby declare that the thesis is my original work and it has
been written by me in its entirety. I have duly
acknowledged all the sources of information which
have been used in the thesis.

This thesis has also not been submitted for any degree
in any university previously.

Lin Bo

October 26, 2020

Acknowledgements

To those who provided support to me during my PhD study. Your assistance was a milestone in the completion of this thesis.

I wish to express my deepest gratitude to my supervisor Professor BAO Weizhu and my co-supervisor Professor CAI Zhenning, for their professional mentorship and guidance. Prof. BAO is a rigorous, brilliant and knowledgeable applied mathematician. It is him who led me to this exciting research field and taught me how to be professional. His generous support and continuous encouragement paved the way to my research road. I learned a large amount of valuable ideas and experience from him in both research and life. Prof. CAI is a talented, efficient and skilled applied mathematician. He can always summarize my problem in a mathematical viewpoint and explain good ideas patiently to help me understand and solve different problems. I have benefited a lot from his professional and practical suggestions. Their rigorous academic attitude and enthusiasm to mathematics have branded me for life.

My sincere thanks also go to Professor ZHUANG Chijie and Professor ZENG Rong, who are my collaborators from Tsinghua University in China. Prof. ZHUANG is an efficient, hard-working and creative engineering researcher who is experienced in the topic of my thesis. He spent a lot of time discussing with me, which contributed to this work and broaden my knowledge in academic and career. I sincerely

appreciate his support when I visited him during my PhD candidature.

I wish to thank Professor CAI Yongyong for his invitation and support in Beijing Computational Science Research Center (CSRC) in China. He provided the funding and helped me to use Tianhe2-JK computing resource, which served as a fundamental tool in this project. I would express my gratitude to Dr. CHEN Lizhen and Dr. WANG Pengfei for their technical support during my usage of Tianhe2-JK. I also want to thank Prof. WANG Yan, Dr. BIAN Lei, Prof. KONG Huihui, Prof. WANG Pengde, Prof. YI Wenfan, Dr. LIU Lei and Dr. MA Ying for their assistance during my visit in CSRC. Without their support and care, this project could not have reached its goal.

I am fortunate to get involved in two wonderful groups and grateful to my colleagues, including but not limited to Prof. TANG Qinglin, Prof. ZHANG Yong, Dr. RUAN Xinran, Dr. SU Chunmei, Dr. ZHAO Quan, Dr. LIU Zhaoqiang, Dr. ZENG Fanhai, Dr. HUANG Weijie, Dr. YIN Jia, Mr. ZHANG Teng, Miss FENG Yue, Mr. GUO Yichen, Mr. WANG Boyi, Mr. GENG Xingri, Dr. KUANG Yang, Mr. YANG Siyao, Mr. WANG Haoxuan, Miss DONG Xiaoyu and Mr. WANG Geshuo. Thanks for their suggestions in seminars and help in life.

I also greatly appreciate NUS and Department of Mathematics for awarding me the scholarship and contributing to a great environment during my graduate study. I want to thank all of my peers in the office for their valuable discussions.

Last but not least, many thanks to my lovely family, my father LIN Zisheng and my mother CAI Shuling, for their unconditional love, trust and support. I owe my deepest gratitude to my girlfriend LIN Meixia, for her love, accompany and numerous encouragement throughout my undergraduate and graduate study. They kept me going on and this project would not have been possible without their love. I dedicate this thesis to them.

LIN Bo

July 2020

Contents

Summary

Streamer discharge is a non-thermal filamentary discharge which happens frequently in nature and has a large amount of industrial applications. Among different models to describe streamer discharge, the fluid model gives a good balance between the characterization of macroscopic quantities of streamer and computational capacity. It can be reduced to two dimensions in some particular cases and the reduced model has been widely studied numerically. However, the numerical simulation of the three-dimensional model is not commonly applied though the streamer discharge intrinsically happens in three dimensions. One possible reason is the large computational cost required for three-dimensional simulations. To ease this difficulty, it is worthwhile to propose accurate and efficient simulations for streamer discharge.

The aim of this thesis is to propose accurate and efficient numerical methods to solve the fluid model of streamer discharge in three dimensions. Details of schemes are illustrated, and the numerical results are reported to show the performance of the proposed methods in comparison with some other numerical methods. This thesis is mainly divided into three parts.

The first part proposes a new second-order semi-implicit scheme for temporal discretization and introduces a geometric multigrid preconditioned FGMRES solver for the resulting variable-coefficient elliptic equation after spatial discretization. The

new semi-implicit scheme can be solved explicitly, and it relaxes the dielectric relaxation time restriction, which is a common constraint in explicit schemes. Moreover, it requires solving the elliptic equation only once at each time step while the classical second-order explicit schemes typically need to do twice. Numerical results demonstrate its second-order convergence and ability to relax the dielectric relaxation time restriction. On the other hand, the introduced multigrid preconditioned FGMRES solver dramatically improves the efficiency of solving the elliptic equation with either constant or variable coefficients. Numerical results show no more than 4 iterations are required for the elliptic solver to converge to a relative residual of 10^{-8} during streamer propagation, and its time usage outperforms other Krylov subspace methods. The parallel efficiency and some properties of streamer are reported through numerical examples of streamer discharge. The first part focuses on the fluid model without photoionization, and the photoionization is studied in the second part.

The second part concentrates on the accurate simulation of the photoionization in streamer discharge based on the classical integral model derived by Zheleznyak *et al.* Two existing PDE-based approximations are reviewed, and they are applied to the calculation of photoionization rate with the help of the efficient multigrid preconditioned FGMRES solver introduced in the first part. After reviewing, a new method, which is the kernel-independent fast multipole method, is introduced to calculate the photoionization rate based on the integral form instead of PDE. The accuracy of this method is studied quantitatively for different domains and various pressures in comparison with other two PDE-based approximations. The comparison indicates the numerical error of the fast multipole method is much smaller than those of other PDE-based methods, with the reference solution given by direct numerical integration. Such accuracy can be achieved with affordable computational cost, and its performance in both efficiency and accuracy is quite stable for different domains and pressures. Meanwhile, the simulation using the fast multipole method exhibits good scalability, which shows its capability of three-dimensional simulations using distributed computing. The difference of the proposed method and other efficient

approximations are also studied in a three-dimensional dynamic problem where two streamers interact.

The third part of this thesis improves the efficiency of simulation by a local adaptive mesh refinement method and provides an implementation of spatial discretization in a non-regular domain using the embedded boundary method. On the one hand, the local adaptive mesh refinement skill exhibits good performance to capture the multiscale structure in streamer discharge dynamically with proper-selected refinement indicators. Therefore, the inclusion of adaptivity reduces the degree of freedom dramatically compared with the one using uniform mesh. On the other hand, numerical results indicate the second-order convergence of the embedded boundary method with proposed semi-implicit temporal discretization. The propagation of streamer is studied numerically in a pin-plane domain, and the effects of different anodes as well as the interaction of streamers are reported.

List of Symbols and Abbreviations

e	elementary charge
ε_0	vacuum dielectric permittivity
c	speed of light
t	time variable
$\mathbb{R}^d$	d-dimensional Euclidean space
$\vec{x} = (x_1, \cdots, x_d)^T$	spatial variable in $\mathbb{R}^d$ space
Δt or τ	time step size
Δx, Δy, Δz	mesh size in x, y, z direction
h	mesh size
l	level index in multi-level hierarchy
h_l	mesh size in level l
$r_{l,l+1}$ or r	ratio of mesh size between level l and $l+1$
N_e or n_e	density of electrons
N_p or n_p	density of positive ions
ϕ	electric potential
$\vec{E}$	electric field
S_i or $\tilde{S}_i$	effective ionization rate
S_{ph} or $\tilde{S}_{\text{ph}}$	photoionization rate

μ_e or $\tilde{\mu}_e$	mobility coefficients for electrons
μ_p or $\tilde{\mu}_p$	mobility coefficient for positive ions
D_e	diagonal matrix of diffusion coefficients for electrons
λ	dimensionless parameter in the Poisson equation
ξ	photoionization efficiency
p	gas pressure
p_{O_2}	partial pressure of oxygen in the gas
1D	one dimension
2D	two dimensions
3D	three dimensions
MPI	message passing interface
SP_3	three-group simplified P_N approximation
FMM	fast multipole method
EB	embedded boundary
AMR	adaptive mesh refinement

Chapter 1

Introduction

This chapter serves as an introduction of this thesis. Firstly, the background of streamer discharge is briefly reviewed, and then the mathematical models to describe streamer are introduced, especially the fluid model. Next, nondimensionalization of the fluid model and existing numerical methods to solve the model are discussed. The final part of this chapter summarizes the problems to study and shows the scope of this thesis.

1.1 Physical background

Gas discharge is frequently observed in nature, and it is an important process in industry which has a number of applications. In nature, besides lightning and small electrostatic discharge which are familiar to most people, gas discharge also occurs in high altitude such as sprites [16, 41], halos [124, 152] and gigantic jet [42, 162]. Some kinds of air discharges in nature are depicted in Figure 1.1, which is reprinted from [106]. In industry, gas discharge has been applied to precipitators [5, 69], polymer films [58], ozone generators [77, 142], water purification [73, 150] and others. On the other hand, gas discharge should be detected and prevented in some industrial applications, such as corona discharge in high-voltage electric power transmission [113] and partial discharge in transformers [74]. Figure 1.2 illustrates

the development of discharge in a conductor-tower lattice, which is reprinted from [29].

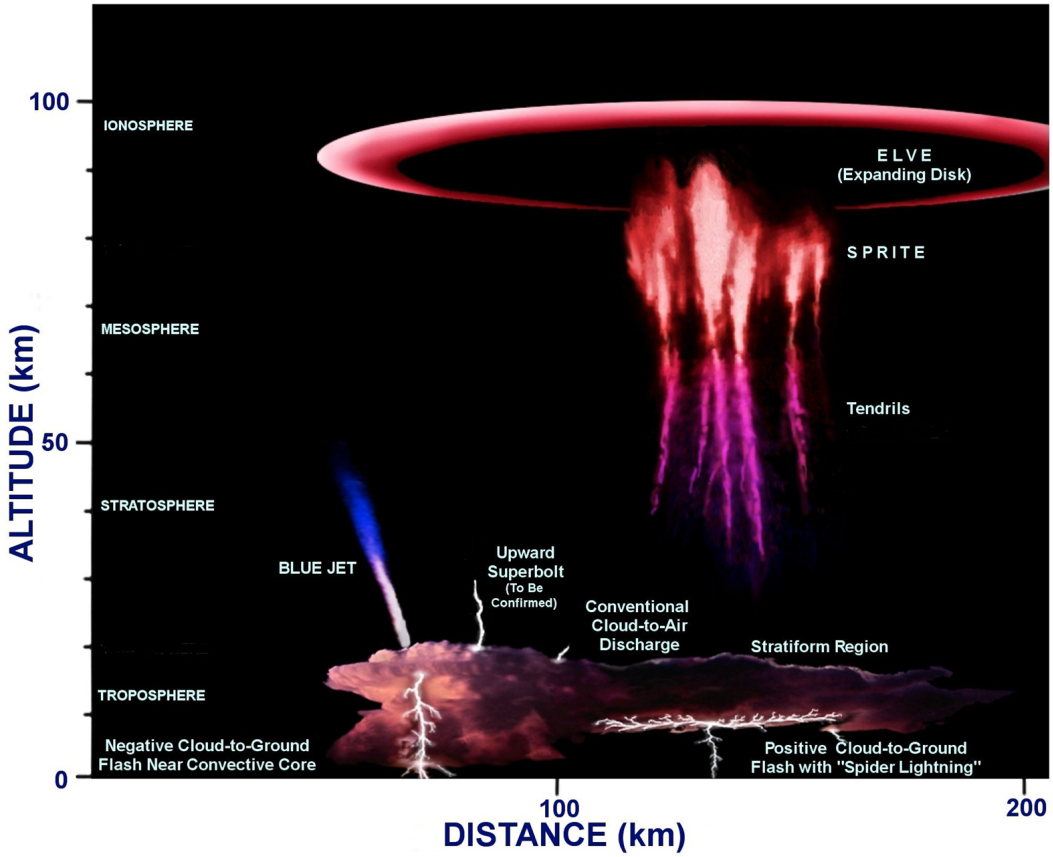


Figure 1.1: Depiction of a sprite, elve, blue jet, cloud-to-air lightning, and the parent lightning within a large mesoscale convective complex. Reprinted from [106] with permission.

Among different kinds of gas discharge, streamer discharge is the pivotal one [53], since it paves the way for lightning leaders [43] and serves as a building block of sprites [99]. Streamer discharge occurs as a consequence of electron avalanche without considering the thermodynamics. Exposed in a high electric field, free electrons are accelerated, and these accelerated electrons acquire adequate energy to ionize neutral particles when the electrons collide with them. This impact ionization generates new free electrons, which then proceed to ionize more neutral particles in a chain reaction. The chain reaction results in electron avalanche. This avalanche,

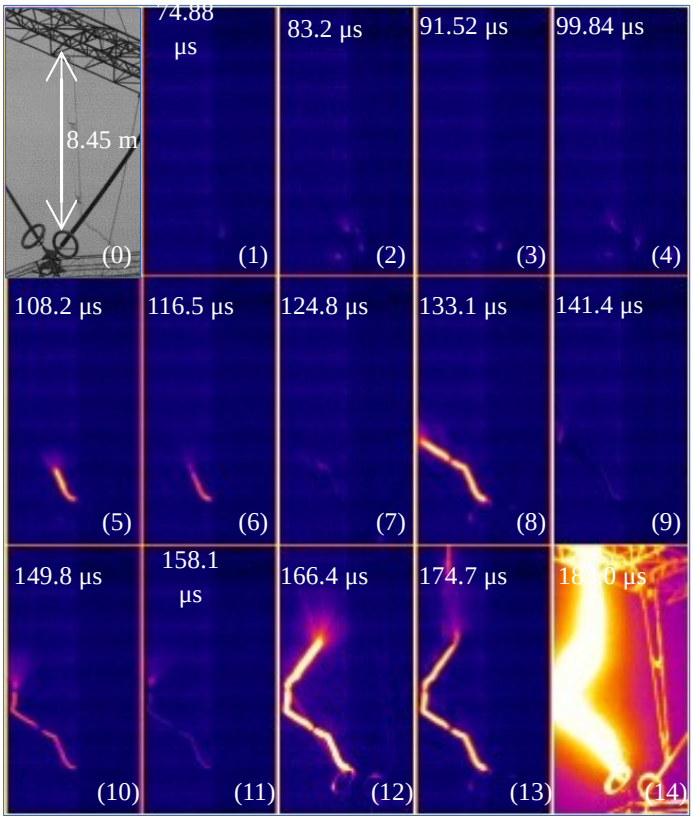


Figure 1.2: The time-resolved photographs of discharge development in a conductor-tower lattice. Reprinted from [29] with permission.

also known as Townsend discharge, forms a conductive plasma [102], and the plasma enhances the electric field at its tips. This enhanced field speeds up the avalanche in a particular direction, and the plasma grows along this direction to form filamentary discharge, which is the streamer discharge.

During the streamer discharge in the air, photoionization plays an important role, especially for positive streamers [6, 126, 129]. Streamers can be further divided into positive ones and negative ones. Positive streamers spread along the same direction of electric field, while negative ones grow in the reverse direction. A combination of these two streamers creates double-headed streamers, which propagate in two directions simultaneously. The propagation of the negative streamer is spontaneous due

to the fast propagation of free electrons at its front; however, the positive streamer propagates only if free electrons exist in front of it. These free electrons have a determining effect of the characteristics of positive streamers, and they are predominantly generated by photoionization in the air [100, 133]. As a result, photoionization is critical to positive streamer discharge in the air. Moreover, the photoionization is responsible for different behaviors of streamers at different pressure [100, 127], not only for the positive ones.

Although streamer discharge was observed and described before 1940 [101, 114], there is still no analytical theory to describe all its properties clearly in detail [53]. The unclear understanding of streamer and its active role in gas discharge motivates a large amount of researchers to study streamer discharges. One important and practical way in these studies is carrying out experiments. With the help of high-speed cameras such as intensified CCD cameras, researchers recorded the luminous channel of streamer in experiments and analyzed its velocities, size, branching and other properties [27, 66, 79, 131, 147, 171]. An example of the experimental device can be found in Figure 1.3, which is reprinted from [180]. These experiments provide macroscopic spatial structures of streamer and electrical properties of discharge, which helps the understanding of its propagation and guides the modeling. They also provide some key coefficients for modeling and simulation, which includes photoionization and ionization.

Another way to study the properties of streamer is numerical simulation. The typical properties of streamer discharge include densities of particles in the plasma as well as the electric field in the discharge region. These properties are difficult to be captured precisely during the discharge in whole region by experiments as streamer propagates on multiple temporal and spatial scales [53]. Therefore, numerical simulations complement the experimental studies on streamer discharges, especially for the quantities in the discharges. In this thesis, we focus on numerical simulations for streamer discharge. The models and the difficulties in solving the models will be discussed in the following chapter.

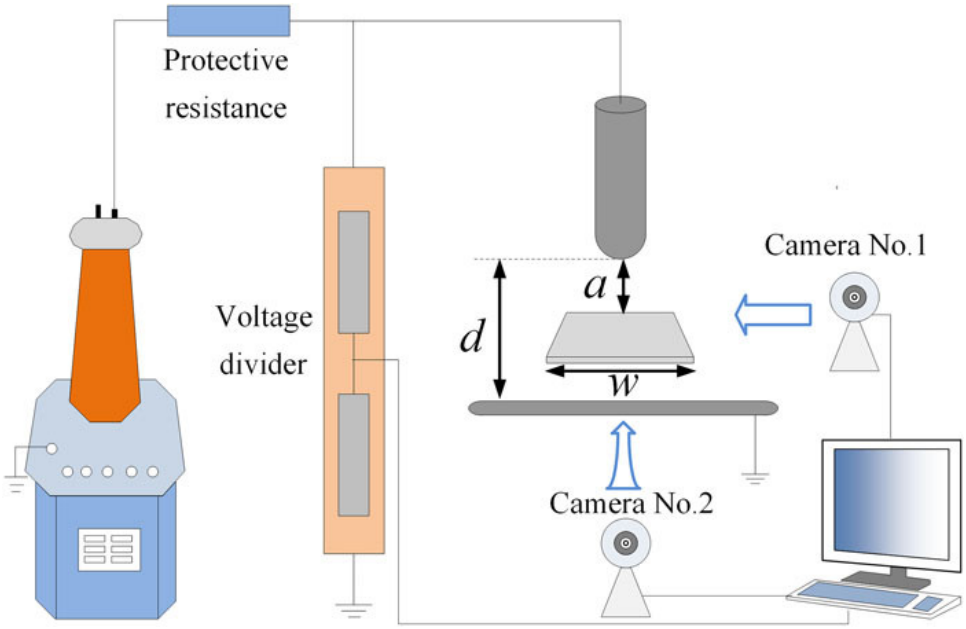


Figure 1.3: Schematic diagram of the experimental setup. Reprinted from [180] with permission.

1.2 Different models of streamer discharges

1.2.1 Electron avalanche and particle model

As discussed in Section 1.1, streamer discharge happens as a consequence of electron avalanche, where electron impact ionization serves as the main mechanism. When the applied voltage between two electrodes is sufficiently high and fixed, the current reaching the anode I is expressed as [138, 159]

$$I = I_0 \exp(\alpha d), \tag{1.1}$$

where I_0 is the initial electron current from cathode, d is the distance between two electrodes and α is the first Townsend ionization coefficient describing the number of new electron-ion pairs created by one electron in advancing one unit length (usually with unit) along the electric field. As a simple corollary, the ionization frequency ν_i is related to the coefficient α by [4]

$$\nu_i = \alpha |\vec{v}_e|, \tag{1.2}$$

where $|\vec{v}_e|$ is the electron velocity. We refer to [119] for the discussion of emergence of streamer from electron avalanche including the Raether-Meek criterion.

Taking consideration of the characteristic mechanism of the particle convection and electron impact ionization, the kinetic or particle model directly captures the movement of electrons and ions. The original collisionless particle model traces all particles by recording their positions and velocities, and accounts the electric field at a particular point by the sum of electric fields generated from every particle [21, 65]. This treatment of electric field is valid only in one dimension (1D), while the idea can be extended to high dimensions by summing the electric forces from all particles. The original particle model directly utilizes the Newton's Law and Maxwell's equations and makes very little approximation. However, as indicated in [21], the simple extension to higher dimensions is too slow. To ease the slow accumulation of Coulomb interactions among all particles, a background grid is introduced in solving the Maxwell's equations or Poisson equation. The idea is composed of three steps: mapping or assigning the particle density to the grid points; solving the electric field on the grid; interpolating the fields on grid to the location of particles. This method, which includes a grid for electric field, is called particle-in-cell (PIC) [15, 62] and has been widely applied in the particle model for gas discharge simulations [26, 82, 85]. Another improvement of the particle model is the inclusion of collisions as source, which captures the elastic collisions, ionization and excitation of electron-neutral and electron-ion pairs [15, 22]. These collisions are simulated mostly by Monte Carlo collision method (MCC). Together with PIC, the PIC-MCC methods gives an efficient implementation of the particle model with collision in three dimensions (3D) [70, 153].

However, the PIC-MCC method suffers from the exponential increasing number of particles during streamer discharge, and this unbounded increment burdens the time of simulation. To limit the total number of particles during simulation, weight of particle or super-particle is introduced [25, 170]. The original particles can be merged into one super-particle by changing the weights of particles. On the other

hand, one super-particle can be split into many particles if the weight of the super-particle is too high [153, 170]. Although the inclusion of super-particle extends the simulation time with present computing capability, it brings numerical fluctuations and stochastic heating [89]. As a result, the PIC-MCC method still needs to simulate a larger amount of particles during streamer discharge (e.g 10^7–10^8 particles in a domain of $5\,\mathrm{mm}^3$ in [153]), which brings a time-consuming numerical simulation in a domain of magnitude of cm in 3D.

1.2.2 A fluid model and its nondimensionalization

Considering the experiments in laboratory and industrial applications, it is important to use a fluid or density model instead of the particle model to focus on macroscopic properties of streamer in a larger spatial scale. On the one hand, the fluid model is suitable to approximate the electrons and ions inside the streamer channel, since they are dense enough to be approximated as continuous densities [91]. On the other hand, the fluid model could be simulated in a much efficient way compared with the particle model. The typical density of particle in streamer under atmospheric pressure is 10^{13}–$10^{14}\,\mathrm{cm}^{-3}$ [123], which is much larger than the current computing capability of using the particle model as 10^7–$10^8\,\mathrm{cm}^{-3}$.

The fluid model and its progress were recently reviewed in [103]. The model describes the movement of particle densities using convection-dominated partial differential equations (PDEs) with source, and it describes the coupling potential and corresponding electric field by the Poisson equation. At the same time, photoionization is counted as an additional source term in the equations of densities. This model has been shown reproducing the dynamics of streamer discharge successfully in [52, 131, 156].

Under assumptions [88, 103] of local field approximation, electrostatic field, etc., the fluid model or density model is valid to approximate the gas density on the specific region in the streamer [52]. The specific expression of fluid model varies with different further approximations [52, 100, 123, 160, 167]. In this thesis, we focus

on the numerical methods for streamer discharge simulations in 3D, therefore we use a minimal model that incorporates the essential mechanism of the phenomena [120,182]. The minimal three-dimensional model for simulating streamer discharges consists of two convection-dominated transport equations coupled with a Poisson equation for the electrical potential and the field:

$$\begin{cases} \dfrac{\partial N_e}{\partial t} - \nabla\cdot\left(\mu_e\vec{E}N_e\right) - \nabla\cdot(D_e\nabla N_e) = S_i + S_{\text{ph}}, \\ \dfrac{\partial N_p}{\partial t} + \nabla\cdot\left(\mu_p\vec{E}N_p\right) = S_i + S_{\text{ph}}, \qquad\qquad \vec{x}\in\Omega, \quad t>0, \\ -\Delta\phi = \dfrac{e}{\varepsilon_0}\left(N_p - N_e\right), \qquad \vec{E} = -\nabla\phi, \end{cases} \tag{1.3}$$

where $\vec{x} = (x, y, z)^T$ and $\Omega \subset \mathbb{R}^3$ is a bounded domain; e and ε_0 are the elementary charge and the vacuum dielectric permittivity, respectively; $N_e := N_e(\vec{x}, t)$ and $N_p := N_p(\vec{x}, t)$ are the densities of electrons and positive ions, respectively; μ_e and μ_p are the mobility coefficients for electrons and positive ions, respectively; $D_e = \text{diag}(D_{e,x}, D_{e,y}, D_{e,z})$, where $D_{e,x}$, $D_{e,y}$ and $D_{e,z}$ are the diffusion coefficients in x, y, z directions, respectively; ϕ and $\vec{E} = (E_x, E_y, E_z)^T$ denote the electric potential and electric field, respectively. Here the photoionization rate S_{ph} is discussed later in Chapter 3. S_i is the effective ionization rate defined as

$$S_i = \bar{\alpha}\mu_e N_e|\vec{E}|, \qquad \text{with} \quad \bar{\alpha} := \bar{\alpha}(|\vec{E}|) = 5.7p\exp(-260p/|\vec{E}|)\,\text{cm}^{-1}, \tag{1.4}$$

while $\bar{\alpha}$ is the effective ionization coefficient taken from [46], p is the air pressure with unit Torr, n_e and $\vec{E}$ are the solution of (1.3). In this thesis, coefficients in (1.3) are taken from [46, 97]: $\mu_e = 2.9\times10^5/p$ cm^2/(V·s), $\mu_p = 2.6\times10^3/p$ cm^2/(V·s), and $D_e = \text{diag}(2190, 2190, 1800)$ cm^2/s. Two physical constants are taken as: $e = 1.6021766\times10^{-19}$ C and $\varepsilon_0 = 8.8541878\times10^{-14}$ F·cm^{-1}.

To study model (1.3) in mathematical viewpoint, nondimensionalization is done as follows: Define the scaled variables

$$\tilde{t} = \frac{t}{t_0}, \qquad \tilde{\vec{x}} = \frac{\vec{x}}{x_0},$$

and

$$n_e = \frac{N_e}{N_0}, \qquad n_p = \frac{N_p}{N_0}, \qquad \tilde{\phi} = \frac{\phi}{\phi_0}, \qquad \tilde{\vec{E}} = \frac{\vec{E}}{E_0}, \qquad \tilde{S}_i = \frac{S_i}{S_0}, \qquad \tilde{S}_{ph} = \frac{S_{ph}}{S_0},$$

where t_0, x_0, N_0, ϕ_0, E_0 and S_0 are some scalar constants with unit to be determined later. Plugging all scaled variable into (1.3) and (1.4), we can obtain

$$\begin{cases} \dfrac{\partial n_e}{\partial \tilde{t}} - \nabla_{\tilde{\vec{x}}} \cdot \left(\dfrac{\mu_e E_0 t_0}{x_0} \tilde{\vec{E}} n_e \right) - \dfrac{t_0 D_e}{x_0^2} \Delta_{\tilde{\vec{x}}} n_e = \dfrac{S_0 t_0}{N_0} \left(\tilde{S}_i + \tilde{S}_{\text{ph}} \right), \\ \dfrac{\partial n_p}{\partial \tilde{t}} + \nabla_{\tilde{\vec{x}}} \cdot \left(\dfrac{\mu_p E_0 t_0}{x_0} \tilde{\vec{E}} n_p \right) = \dfrac{S_0 t_0}{N_0} \left(\tilde{S}_i + \tilde{S}_{\text{ph}} \right), \\ -\Delta_{\tilde{\vec{x}}} \tilde{\phi} = \dfrac{x_0^2 N_0 e}{\phi_0 \varepsilon_0} \left(n_p - n_e \right), \qquad \tilde{\vec{E}} = -\dfrac{\phi_0}{E_0 x_0} \nabla_{\tilde{\vec{x}}} \tilde{\phi}, \\ \tilde{S}_i = \dfrac{\bar{\alpha} \mu_e N_0 E_0}{S_0} n_e |\tilde{\vec{E}}|. \end{cases} \tag{1.5}$$

If we choose the constants with unit to mimic the streamer discharge between two parallel electrodes, then it is natural to choose $x_0 = d$, where d is the length between two electrodes as shown in (1.1). Similarly, let ϕ_0 be the corresponding potential difference between electrodes. In particular, if zero potential is assumed for one electrode, ϕ_0 is the potential on the other electrode. With these two selected constant x_0 and ϕ_0, let

$$\frac{\phi_0}{E_0 x_0} = 1, \qquad \frac{\mu_e E_0 t_0}{x_0} = 1,$$

which means scaling constant t_0 and E_0 are

$$E_0 = \frac{\phi_0}{x_0}, \qquad t_0 = \frac{x_0}{\mu_e E_0} = \frac{x_0^2}{\mu_e \phi_0}.$$

To simulate streamer discharge, we would like to take $N_0 = 10^{14}\,\text{cm}^{-3}$, which is the typical electron density in the streamer. Then taking $S_0 = \frac{N_0}{t_0} = \frac{N_0 \mu_e \phi_0}{x_0^2}$ be the typical rate of source and (1.5) becomes

$$\begin{cases} \dfrac{\partial n_e}{\partial \tilde{t}} - \nabla_{\tilde{\vec{x}}} \cdot \left(\tilde{\mu}_e \tilde{\vec{E}} n_e \right) - \tilde{D}_e \Delta_{\tilde{\vec{x}}} n_e = \tilde{S}_i + \tilde{S}_{\text{ph}}, \\ \dfrac{\partial n_p}{\partial \tilde{t}} + \nabla_{\tilde{\vec{x}}} \cdot \left(\tilde{\mu}_p \tilde{\vec{E}} n_p \right) = \tilde{S}_i + \tilde{S}_{\text{ph}}, \\ -\lambda \Delta_{\tilde{\vec{x}}} \tilde{\phi} = n_p - n_e, \qquad \tilde{\vec{E}} = -\nabla_{\tilde{\vec{x}}} \tilde{\phi}, \\ \tilde{S}_i = \tilde{\alpha} n_e \tilde{\mu}_e |\tilde{\vec{E}}|, \end{cases} \tag{1.6}$$

where $\lambda = \frac{\phi_0 \varepsilon_0}{x_0^2 N_0 e}$ is the dimensionless parameter in the Poisson equation that is typically less than 1 for streamer discharge, $\tilde{\mu}_e = 1$, $\tilde{\mu}_p = \frac{\mu_p}{\mu_e} \approx 8.9655 \times 10^{-3}$ are

the dimensionless mobility parameter for electrons and positive ions, respectively; $\tilde{D}_e = \frac{D_e}{\mu_e \phi_0}$ is the dimensionless diffusion coefficient for electrons and $\tilde{\alpha} = \bar{\alpha} x_0$ is the dimensionless ionization coefficient. To avoid abuse usage of tilde symbols in the remaining thesis, we would like to reuse notations t, $\vec{x}$, ∇, Δ, ϕ and $\vec{E}$ in (1.6) instead of $\tilde{t}$, $\tilde{\vec{x}}$, $\nabla_{\tilde{\vec{x}}}$, $\Delta_{\tilde{\vec{x}}}$, $\tilde{\phi}$ and $\tilde{\vec{E}}$, respectively. Similarly, we reuse $\Omega \subset \mathbb{R}^3$ to denote the dimensionless bounded domain. In this thesis, we would like to take typical potential $\phi_0 = 52\,\mathrm{kV}$, typical length $x_0 = 1\,\mathrm{cm}$ and typical pressure $p = 760\,\mathrm{Torr}$ unless otherwise stated. Calculated from these typical values, we can get the dimensionless parameters $\tilde{D}_e \approx \mathrm{Diag}(1.1037\times10^{-4}, 1.1037\times10^{-4}, 9.0716\times10^{-5})$, $\lambda \approx 2.8737\times10^{-4}$, and $\tilde{\alpha} = 4332\exp(-3.8/|\vec{E}|)$.

Before reviewing existing methods for the fluid model in the following section, it should be simply commented here that besides the particle model and the fluid model, some researchers made a hybrid model of these two models [90, 91]. The idea is simulating the major channel of streamer by the fluid model and tracing the limited number of particle at its tip by the particle model without introducing super-particle. This method could follow the run-away of individual electrons with high energies by the inclusion of particle model [92], however, it requires accurate and efficient simulation of the fluid model in the major channel.

1.3 Literature review

Continuous efforts have been made to simulate streamer discharges through the fluid model over the past few decades. The early studies of these numerical simulation dated back to 1965-1975. In 1965, Ward [169] first studied electrical breakdown of streamer discharge numerically using a finite difference method. Later in 1971, Davies *et al.* [44] improved the numerical error of the finite difference method by using the characteristic line method, which was applied to the transport equations. Moreover, the authors presented a more accurate calculation of the electric field using an image charge method. These two works were further improved by Kline [76]

as well as Yoshida and Tagashira [174] to include photoionization and the diffusion of electrons in the transport equations. The early works discussed above pave the way for using numerical simulation to study the properties of the streamer discharge. However, these works mostly focused in one dimension, and thus were limited to the case when the cross section of streamer was constant. The numerical methods should be applied to higher dimensions, which can simulate different shapes of streamers.

To overcome the limitation of the one-dimensional simulations of streamer discharge, a two-dimensional simulation based on the finite difference method was proposed. In 1976, Yoshida and Tagashira [175] applied the finite difference method to axisymmetric three-dimensional model with a splitting strategy for equations of electrons. The finite difference method was then tested and compared in the fluid model in [121], which performed best with flux-corrected transport (FCT, [17, 177]) techniques. Therefore, in the 1980s and 1990s, FCT techniques were widely used. It was combined with the finite difference method and finite element method (FEM) to overcome the numerical oscillations that occur when classical linear schemes are used to solve convection-dominated equations [56, 122]. Compared with the finite difference method, FEM can be easily applied to general discharge regions including point-to-plane region [108]. The idea of FEM was first introduced in the Poisson equation with cubic spline basis in [81]. Subsequently, the combination of FCT (FEM-FCT, [86, 87]) was applied to the transport equations since 2000 in [56, 115]. Although FEM can easily handle general simulation domain and achieve high-order accuracy, it is not locally conservative, which violates the total current law in the simulation. Moreover, the FEM requires a global linear solver for the transport equations, which could be a little time-consuming and cannot be applied to distributed computing easily.

Later, the finite volume method (FVM) became popular since 2000s due to the property of local conservation [120, 176], and it was shown to be faster than FEM in [50]. Although it needs wider stencils compared with FEM, these stencils are constructed locally. Because of this local property, FVM can be easily upgraded

to a parallel version [137, 154]. Motivated by the successes of FVM and FEM, the discontinuous Galerkin (DG) method, which uses a finite element discretization with discontinuous basis functions and incorporates the ideas of numerical fluxes and slope limiters from the high-resolution FDM and FVM, was used to simulate the streamers [181,183,184]. Compared with FVM, DG requires shorter stencils and could be upgraded to higher order more easily. However, if a second-order scheme is desired, FVM is easy to implement and requires less storage of variables. As a result, a parallel version of FVM is adopted in this thesis.

The previous improvements in the numerical methods achieved great progress in streamer simulations [7], especially in two-dimensional cases where the streamer is assumed to be axisymmetric. However, compared with 2D simulations, studies of real three-dimensional simulations are much fewer [109, 137, 145, 154, 166, 167].

Although efficient compared with the particle model, the fluid model still takes a long time for one simulation, especially in three dimensions where the streamer discharge inherently happens. The difficulty in three-dimensional simulations lies in the fine meshes needed to simulate rapid variations in the solution. Streamer discharges propagate at dramatic speeds, e.g., at $10^8\,\mathrm{cm/s}$, as shown in Fig. 7 of [28]. During this rapid transient process, the electric field in the discharge channel, which is one of the key parameters dominating the development of a streamer, varies significantly both temporally and spatially. After streamer inception, the electric field at its head is greatly enhanced due to the net charge accumulation, which further accelerates the ionization and charge accumulation. Thus, a sharp charge density profile forms at the streamer's head. Capturing the structures of the charge carriers in a simulation requires a very high-resolution spatial grid. Typically, the order of magnitude for the grid size adopted in previous simulations has been characterized by micrometres [14,154], which is tiny compared with the characteristic length of the problem at the scale of, e.g., centimetres. Consequently, the maximal allowed time step is restricted to the order of several picoseconds or even smaller when explicit schemes are used.

In addition, because the Poisson equation and transport equations for the charge carriers are coupled together, the time step is further restricted by the dielectric relaxation time, i.e., $\varepsilon_0/e(\mu_p N_p + \mu_e N_e)$ in (1.3), which is also typically on the order of several picoseconds. The dimensionless dielectric relaxation time τ_{diel} in (1.6) is similarly given as

$$\tau_{\text{diel}} = \frac{\lambda}{\max_{\vec{x}\in\Omega}(\tilde{\mu}_p n_p + \tilde{\mu}_e n_e)}. \tag{1.7}$$

It should be noted that the dielectric relaxation restriction on time step could be attributed to the explicit treatment of potential in the transport equations in (1.6), and this restriction does not depend on the spatial resolution. There are several viewpoints and derivations for (1.7). In [9], this restriction is taken to prevent net charge $(n_p - n_e)$ from getting wrong sign. In [110, 155], (1.7) is derived from Ampere's law and regarded as the typical time scale for electric screening, which describes the rate-of-change of electric field due to the moving of charged particles. These explanations mainly come from physical viewpoint, and a more analytical explanation is proposed in [36] using scaling analysis.

For these reasons, even two-dimensional simulations require long computational times, let alone three-dimensional simulations which have thousands times the number of degrees of freedom (DOFs) in 2D simulations, for even a small domain in 3D. Thus, it seems parallel computing is the only possible way to efficiently construct large-scale three-dimensional simulations for streamer discharges.

Recently, Teunissen and Ebert reported on 3D streamer simulations using the parallel adaptive Afivo framework [154], which features adaptive mesh refinement (AMR), geometric multigrid methods for the Poisson equation and OpenMP parallelism. AMR performs well, however, further improvement could be made by replacing the OpenMP parallelism with message-passing interface (MPI) libraries to fully exploit the power of clusters, especially for simulations of very long streamers. Another advance in the MPI-based simulation was reported by Plewa, Eichwald, and Ducasse *et al.* [137], who used the successive over-relaxation iterative solver in

the red and black strategy (R&B SOR) as the Poisson solver, and tested the parallelization and the scalability with cell numbers ranging from 8-512 million and core numbers from 20-1600. Their use of high-performance computing clusters with an MPI implementation reduced the computational time. Taking advantage of MPI, Marskar reported some surprising results in general discharge regions in 3D [109], which uses AMR and features embedded boundary method (EB) for the general boundaries.

These previous works suggest an efficient and accurate simulator should be proposed to shorten the computation time in 3D, which should have advantages in the following aspects: (1) an efficient time integration scheme, which may reduce the time marching steps; (2) a fast algebraic elliptic solver, which accelerates the solution of the Poisson equation or elliptic equations that dominates the total computing time; (3) a good parallelization, which allows to utilize the full power of modern clusters; (4) adaptive mesh strategies, which can reduce the number of DOFs; (5) a suitable treatment on boundaries, which broadens simulation to more geometries. These aspects should be considered when designing a simulator with or without considering the accurate simulation of photoionization.

As discussed in Section 1.1, photoionization plays an important role in positive streamer in the air, and therefore accurate calculation is required numerically. At an early stage of the numerical simulations of the streamer discharge [44,169], photoionization was considered and applied only on the boundary of the fluid model. Later in [76], photoionization was added as an additional source term, and this model is now widely accepted for the simulation of the streamer discharge and is adopted in this thesis.

Although the photoionization plays an important role in streamer discharge, the calculation of it is not accurate or efficient enough. The classical model for oxygen-nitrogen mixture derived by Zheleznyak *et al.* in [179] is widely utilized in the simulation of positive streamers [129,148], and was improved in [71,130] to gain better accuracy and has been extended to a stochastic version in [25]. However, direct

calculation of the classical integral model requires a large amount of computation, especially in 3D where streamer discharges inherently happen. This unaffordable time seems to form a barrier for simulating streamer considering photoionization in 3D.

To ease the numerical difficulty and reduce the computational cost, some approximation methods are proposed in [18,98,105,144] based on the kernel expansion and conversion to Helmholtz equations. These approximations are designed to calculate photoionization efficiently and make it practical to simulate streamers with photoionization in higher dimensions, especially positive streamers. These approximations give satisfactory simulations when the absorption coefficient for radiation is high enough, however, as indicated in [23], these efficient models becomes less accurate and more time-consuming as the absorption coefficient decreases. As a result, modeling of photoionization directly based on solving the radiative transfer equation (RTE) was suggested in [23] for 2D simulations. The extension of this idea to 3D might be computationally expensive due to the inclusion of solving a five-dimensional steady PDE at each time step.

Less than two decades ago, the kernel-independent fast multipole method (FMM) was proposed to compute particle interactions efficiently and accurately [172, 173]. It can be easily applied to the convolutional integrals [107], and its computational complexity is comparable to the method of fast Fourier transform (FFT). Compared with FFT, FMM can be applied to more general computational domains and has better parallel efficiency in distributed computations. In addition, it can be directly applied on a broad class of different integral forms compared with the kernel-dependent FMM which requires kernel expansion and efficient translation for different specific kernels [30, 59–61, 94]. Motivated by the good performance of the kernel-independent FMM, this thesis extends its application to the computation of photoionization rates, and focuses on the following properties: (1) accuracy and robustness for different pressures; (2) good efficiency; and (3) extensibility to other integral models.

1.4 Contributions

As is pointed out in Section 1.3, although there has been much effort devoted to numerical study and implementation of the fluid model, there still lacks a fast numerical methods in three dimensions with satisfactory accuracy for different pressures. This motivates us to propose robust three-dimensional accurate and efficient parallel simulations for streamer discharges. Specifically, this thesis focuses on

- Proposing an efficient and accurate MPI-based parallel simulator for streamer discharges in three dimensions without photoionization. First, a new second-order semi-implicit scheme is proposed for the temporal discretization of the model that relaxes the dielectric relaxation time restriction. Moreover, it requires solving the Poisson-type equation only once at each time step, while the classical second-order explicit scheme typically need to do so twice. Second, a geometric multigrid preconditioned FGMRES solver is introduced that dramatically improves the efficiency of solving the Poisson-type equation. Last but not least, the satisfactory parallel efficiency of the code is demonstrated.

- Extending the application of the kernel-independent fast multipole method to efficiently evaluate the photoionization based on the classical integral model. The accuracy and computational cost of this method is studied quantitatively for different domains and various pressures in comparison with other existing models based on PDEs. The capability of three-dimensional simulations using parallel computing is reported.

- Generalizing the computational domain to general geometries by embedded boundary method. Furthermore, local adaptive refinement skills are included to reduce the number of DOFs and make simulations more efficient. The convergence and stability of the embedded boundary method is studied, and different refinement indicators are compared in adaptive refinement method. The streamer propagation as well as interaction between streamers in pin-plane domains is studied.

1.5 Organization of the thesis

This thesis is organized as follows.

In Chapter 2, a 3D parallel simulator is proposed for streamer discharge between plane-plane electrodes without photoionization. First, A new second-order semi-implicit temporal scheme is described, and the spatial discretization using FVM is briefly stated. Then a multigrid preconditioned FGMRES elliptic solver is introduced. Numerical comparisons illustrate the convergence and stability of different schemes, show the scalability of parallel implementation and compare the performance of different algebraic elliptic solvers. Simulation results of a double-headed streamer propagation as well as the interaction between two streamers are reported.

Chapter 3 focuses on accurate and efficient simulation of the photoionization in 3D based on classical integral model. The classical integral method and its associated PDE-based approximations are reviewed. After reviewing, the fast multipole method on a general numerical integral form is introduced. The quantified performance of the fast multipole method and comparisons with other approximations for computing photoionization as well as streamer discharges are presented in numerical experiments.

Chapter 4 is devoted to improving the efficiency of the parallel simulator with adaptivity and extending it to deal with general boundaries of simulation domain. The local adaptive mesh refinement and embedded boundary method are introduced for transport equations and elliptic equations individually, and the refinement criteria is briefly discussed. Numerical examples demonstrate the performance of these methods including the convergence and scalability, after which the improved simulator is applied to streamer discharge with different shapes of electrodes as well as streamer interaction in a pin-plane domain.

Chapter 5 draws a conclusion of the thesis and discusses some possible future work.

Chapter 2

A Parallel Simulator for the Fluid Model

This chapter focuses on numerical methods for the fluid equation (1.6) without considering photoionization. Firstly, the problem setting and boundary conditions are briefly introduced. Secondly, the temporal discretization is discussed, including a new semi-implicit scheme. Thirdly, the finite volume method is taken for spatial discretization, and then a multigrid preconditioned FGMRES algebraic solver is reported to solve elliptic equations. Finally, numerical experiments illustrate the performance of proposed method, and the proposed simulator is applied to study the propagation of streamers.

2.1 Problem setting and boundary conditions

To model the streamer discharge between two parallel plates, a cubic domain $\Omega = (x_0, x_1) \times (y_0, y_1) \times (z_0, z_1)$ is considered in this chapter. Dirichlet boundary conditions are applied for the potential, ϕ, on the upper and lower plate electrodes (i.e., $\phi|_{z=z_1} = 1$ where the constant applied potential is chosen, and $\phi|_{z=z_0} = 0$); and homogeneous Neumann boundary conditions, which are $\frac{\partial \phi}{\partial x}|_{x=x_0,x_1} = 0$ and $\frac{\partial \phi}{\partial y}|_{y=y_0,y_1} = 0$, are applied on other four sides. Initial conditions for n_e and n_p are given as

$$n_e(\vec{x}, t = 0) = n_{e,0}(\vec{x}), \qquad n_p(\vec{x}, t = 0) = n_{p,0}(\vec{x}), \qquad \vec{x} \in \bar{\Omega}. \tag{2.1}$$

For simplicity, the plasma is initially assumed to be electrically neutral everywhere, which gives $n_{e,0}(\vec{x}) = n_{p,0}(\vec{x}) = n_0(\vec{x})$ with $n_0(\vec{x})$ a given function. Homogeneous Neumann boundary conditions are applied at all the boundaries for n_e, and at all inflow boundaries for n_p.

The chapter does not consider the non-local photoionization in the fluid model. As a result, we assume $\tilde{S}_{\text{ph}} = 0$ in (1.6) and leave the calculation of photoionization in Chapter 3. However, as indicated in Chapter 1, the free electrons from photoionization play an important role in the propagation of streamer, especially the positive streamer. Therefore, in order to include the free electrons without photoionization, we would like to add a small homogeneous pre-ionization into $n_0(\vec{x})$ before the simulation. This idea is widely used for the simulations without calculating the photoionization [14, 129, 137, 154].

It should be noted that though this chapter discusses numerical methods on a cubic domain, these methods are not limited to the cubic domain. The implementation and extension of the proposed methods for the streamer discharge in general domains will be shown later in Chapter 4.

2.2 Temporal discretization

In this section, we focus on temporal discretization. After reviewing some existing explicit and implicit schemes, we present a new second-order semi-implicit scheme [97].

Let $t_0 = 0$, $\tau_n > 0$ be the time step at n-th step, and $t_{n+1} = t_n + \tau_n$ for $n \geq 0$. We use $n_e^n = n_e^n(\vec{x})$, $n_p^n = n_p^n(\vec{x})$, $\phi^n = \phi(\vec{x})$ and $\vec{E}^n = \vec{E}^n(\vec{x})$ to denote the associated quantities to be approximated at time t_n.

2.2.1 Explicit schemes

To avoid solving nonlinear algebraic equations, explicit schemes are frequently used for temporal discretization, among which the forward Euler scheme is used to

discretize the model (1.6) with (1.4) as

$$\begin{cases} \dfrac{n_e^{n+1}-n_e^n}{\tau_n} - \nabla\cdot(\tilde{\mu}_e\vec{E}^n n_e^n) - \nabla\cdot(\tilde{D}_e\nabla n_e^n) = \tilde{\alpha}(|\vec{E}^n|)\tilde{\mu}_e|\vec{E}^n|n_e^n, \\ \dfrac{n_p^{n+1}-n_p^n}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^n n_p^n) = \tilde{\alpha}(|\vec{E}^n|)\tilde{\mu}_e|\vec{E}^n|n_e^n, \qquad \vec{x}\in\Omega. \\ -\lambda\Delta\phi^n = n_p^n - n_e^n, \qquad \vec{E}^n = -\nabla\phi^n, \end{cases} \tag{2.2}$$

At each time step, the potential ϕ^n is first calculated by the Poisson equation, and then n_e^{n+1} and n_p^{n+1} are obtained subsequently. The Poisson equation need to be solved once at each time step.

It is easy to see the scheme (2.2) is only first order in time, and it has been upgraded to second order by Heun's method, as is used in [14, 154]. The first stage of Heun's method is to solve ϕ^n, n_e^* and n_p^* from n_e^n and n_p^n,

$$\begin{cases} \dfrac{n_e^*-n_e^n}{\tau_n} - \nabla\cdot(\tilde{\mu}_e\vec{E}^n n_e^n) - \nabla\cdot(\tilde{D}_e\nabla n_e^n) = \tilde{\alpha}(|\vec{E}^n|)\tilde{\mu}_e|\vec{E}^n|n_e^n, \\ \dfrac{n_p^*-n_p^n}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^n n_p^n) = \tilde{\alpha}(|\vec{E}^n|)\tilde{\mu}_e|\vec{E}^n|n_e^n, \qquad \vec{x}\in\Omega, \\ -\lambda\Delta\phi^n = n_p^n - n_e^n, \qquad \vec{E}^n = -\nabla\phi^n, \end{cases} \tag{2.3}$$

and then evolve the solution through one more stage to obtain n_e^{**} and n_p^{**}:

$$\begin{cases} \dfrac{n_e^{**}-n_e^*}{\tau_n} - \nabla\cdot(\tilde{\mu}_e\vec{E}^* n_e^*) - \nabla\cdot(\tilde{D}_e\nabla n_e^*) = \tilde{\alpha}(|\vec{E}^*|)\tilde{\mu}_e|\vec{E}^*|n_e^*, \\ \dfrac{n_p^{**}-n_p^*}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^* n_p^*) = \tilde{\alpha}(|\vec{E}^*|)\tilde{\mu}_e|\vec{E}^*|n_e^*, \qquad \vec{x}\in\Omega. \\ -\lambda\Delta\phi^* = n_p^* - n_e^*, \qquad \vec{E}^* = -\nabla\phi^*, \end{cases} \tag{2.4}$$

The final solution at t_{n+1} is constructed by

$$n_e^{n+1} = \frac{1}{2}\left(n_e^n + n_e^{**}\right), \qquad n_p^{n+1} = \frac{1}{2}\left(n_p^n + n_p^{**}\right), \qquad \vec{x}\in\Omega. \tag{2.5}$$

This temporal scheme possesses a second-order time accuracy.

We emphasize that the second-order explicit scheme shown in (2.3)–(2.5) requires solving the Poisson equation twice at each time step (from t_n to t_{n+1}). Moreover, it was suggested in [9, 154] that these explicit schemes need to satisfy the dielectric relaxation time constraint (1.7), i.e.,

$$\tau_n \le \frac{\lambda}{\max_{\vec{x}\in\Omega}(\tilde{\mu}_p n_p^n + \tilde{\mu}_e n_e^n)}, \qquad n \ge 0. \tag{2.6}$$

2.2.2 Implicit schemes

To relax the dielectric relaxation time constraint (2.6), semi-implicit schemes were introduced [165, 168]. In [168], Villa et al. proposed a first-order semi-implicit scheme with a rigorous asymptotic preserving property. Here we present the scheme with a slight modification as

$$\begin{cases} \dfrac{n_e^{n+1}-n_e^n}{\tau_n} - \nabla\cdot(\tilde{\mu}_e\vec{E}^{n+1}n_e^n) - \nabla\cdot(\tilde{D}_e\nabla n_e^n) = \tilde{\alpha}(|\vec{E}^{n+1}|)\tilde{\mu}_e|\vec{E}^{n+1}|n_e^n, \\ \dfrac{n_p^{n+1}-n_p^n}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^{n+1}n_p^n) = \tilde{\alpha}(|\vec{E}^{n+1}|)\tilde{\mu}_e|\vec{E}^{n+1}|n_e^n, \qquad \vec{x}\in\Omega. \\ -\lambda\Delta\phi^{n+1} = n_p^{n+1}-n_e^{n+1}, \qquad \vec{E}^{n+1} = -\nabla\phi^{n+1}, \end{cases} \tag{2.7}$$

In [168], a fully implicit source term $\tilde{\alpha}(|\vec{E}^{n+1}|)\tilde{\mu}_e|\vec{E}^{n+1}|n_e^{n+1}$ was adopted; however, our simplification of the source term in (2.7) does not affect the proof of the asymptotic preserving property. A comparison of (2.7) and (2.2) shows that the main difference between the semi-implicit scheme and the explicit schemes lies in whether the electric field is treated implicitly. As demonstrated in [168], when the reference states of n_e, n_p and $\vec{E}$ are bounded, the time step τ_n is no longer restricted by the dielectric relaxation time.

Although (2.7) is a semi-implicit discretization of (1.6), thanks to the structure of (2.7), it can be solved explicitly by rewriting the Poisson equation as a variable coefficient elliptic equation or Poisson-type equation [168]. A subtraction of the first two equations in (2.7) yields

$$\frac{(n_p^{n+1}-n_e^{n+1})-(n_p^n-n_e^n)}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^{n+1}n_p^n) + \nabla\cdot(\tilde{\mu}_e\vec{E}^{n+1}n_e^n) + \nabla\cdot(\tilde{D}_e\nabla n_e^n) = 0. \tag{2.8}$$

Then, we plug the expression of $(n_p^{n+1}-n_e^{n+1})$ in (2.8) into the Poisson equation in (2.7), and obtain an elliptic equation:

$$-\nabla\cdot\left(\left(\lambda+\tau_n\left(\tilde{\mu}_p n_p^n+\tilde{\mu}_e n_e^n\right)\right)\nabla\phi^{n+1}\right) = n_p^n - n_e^n - \tau_n\nabla\cdot(\tilde{D}_e\nabla n_e^n). \tag{2.9}$$

After solving the variable coefficient elliptic problem (2.9), we obtain ϕ^{n+1}. Then, we can calculate $\vec{E}^{n+1} = -\nabla\phi^{n+1}$, and evolve the first two equations in (2.7) to obtain n_e^{n+1} and n_p^{n+1}.

The elliptic equation (2.9) has another important implication for stability with respect to the dielectric relaxation time. If we further plug $(n_p^n - n_e^n)$ in the right-hand side of (2.9) by the Poisson equation as $-\lambda\Delta\phi^n$, the remaining equation can be rewritten as

$$-\lambda\nabla\cdot\left(\nabla\left(\frac{\phi^{n+1}-\phi^n}{\tau_n}\right)\right) = \nabla\cdot\left(\left(\tilde{\mu}_p n_p^n + \tilde{\mu}_e n_e^n\right)\nabla\phi^{n+1}\right) - \nabla\cdot(\tilde{D}_e\nabla n_e^n), \quad (2.10)$$

which can be regarded as the backward Euler scheme of following differential equation for ϕ

$$-\lambda\nabla\cdot\left(\nabla\left(\frac{\partial\phi}{\partial t}\right)\right) = \nabla\cdot\left(\left(\tilde{\mu}_p n_p^n + \tilde{\mu}_e n_e^n\right)\nabla\phi\right) - \nabla\cdot(\tilde{D}_e\nabla n_e^n). \quad (2.11)$$

Since backward Euler scheme is unconditionally stable, we do not need additional stability condition on (2.10). On the other hand, by similar procedure of (2.8)–(2.10), we could derive a similar equation as (2.10) by the explicit scheme (2.2), which just replaces ϕ^{n+1} on the right-hand side by ϕ^n. Therefore, the explicit scheme (2.2) can be regarded as forward Euler discretization of (2.11), and the dielectric relaxation time constraint (2.6) is the stability condition of this discretization.

Scheme (2.7) is only first order accurate in time, which will be numerically demonstrated later in Section 2.6.1. Here we propose a new second-order semi-implicit scheme for (1.6). Our scheme can be regarded as a predictor-corrector method. First, we calculate a prediction $n_e^{n+1/2}$, $n_p^{n+1/2}$, $\phi^{n+1/2}$ and $\vec{E}^{n+1/2}$ at time $t_n + \tau_n/2$ using the first-order semi-implicit scheme (2.7), i.e.,

$$\begin{cases} \dfrac{n_e^{n+1/2}-n_e^n}{\tau_n/2} - \nabla\cdot(\tilde{\mu}_e\vec{E}^{n+1/2}n_e^n) - \nabla\cdot(\tilde{D}_e\nabla n_e^n) = \tilde{\alpha}(|\vec{E}^{n+1/2}|)\tilde{\mu}_e|\vec{E}^{n+1/2}|n_e^n, \\ \dfrac{n_p^{n+1/2}-n_p^n}{\tau_n/2} + \nabla\cdot(\tilde{\mu}_p\vec{E}^{n+1/2}n_p^n) = \tilde{\alpha}(|\vec{E}^{n+1/2}|)\tilde{\mu}_e|\vec{E}^{n+1/2}|n_e^n, \qquad \vec{x}\in\Omega. \\ -\lambda\Delta\phi^{n+1/2} = n_p^{n+1/2} - n_e^{n+1/2}, \qquad \vec{E}^{n+1/2} = -\nabla\phi^{n+1/2}, \end{cases} \quad (2.12)$$

Then, we get a correction of n_e and n_p using a midpoint scheme, which yields (2.13)

$$\begin{cases} \dfrac{n_e^{n+1}-n_e^n}{\tau_n} - \nabla\cdot(\tilde{\mu}_e\vec{E}^{n+1/2}n_e^{n+1/2}) - \nabla\cdot(\tilde{D}_e\nabla n_e^{n+1/2}) = \tilde{\alpha}(|\vec{E}^{n+1/2}|)\tilde{\mu}_e|\vec{E}^{n+1/2}|n_e^{n+1/2}, \\ \dfrac{n_p^{n+1}-n_p^n}{\tau_n} + \nabla\cdot(\tilde{\mu}_p\vec{E}^{n+1/2}n_p^{n+1/2}) = \tilde{\alpha}(|\vec{E}^{n+1/2}|)\tilde{\mu}_e|\vec{E}^{n+1/2}|n_e^{n+1/2}, \qquad \vec{x}\in\Omega. \end{cases} \tag{2.13}$$

The potential $\phi^{n+1/2}$ and electric field $\vec{E}^{n+1/2}$ are already predicted at time $t_n+\tau_n/2$ by solving the following variable coefficient elliptic equation derived from (2.12):

$$-\nabla\cdot\left(\left(\lambda+\frac{\tau_n}{2}\left(\tilde{\mu}_p n_p^n+\tilde{\mu}_e n_e^n\right)\right)\nabla\phi^{n+1/2}\right) = n_p^n-n_e^n-\frac{\tau_n}{2}\nabla\cdot(\tilde{D}_e\nabla n_e^n), \tag{2.14}$$

consequently, $\phi^{n+1/2}$ and $\vec{E}^{n+1/2}$ can be reused in (2.13) without additional calculation. Therefore, the elliptic equation is solved only once at each time step.

The basic idea for reducing the computational cost is to mimic the underlying mechanism of the second-order implicit midpoint rule [67, Chapter 3], in which the right-hand side appears only once at each time step. To avoid solving nonlinear systems, this mechanism is applied only to the electric field; the other parts are implemented following the explicit midpoint method. Comparing the first-order scheme (2.7) with our second-order scheme (2.12)–(2.13), and focusing on the treatment of the electric field, the difference is similar to the difference between the backward Euler method and the implicit midpoint method. However, it is well known that the backward Euler method is L-stable, while the implicit midpoint method is not. Hence, due to the strong relation between L-stability and the asymptotic preserving property [55], when using (2.12)–(2.13), we will probably lose the asymptotic preserving property while gaining one additional numerical order. Nevertheless, due to its implicit nature, the scheme in (2.12)–(2.13) is indeed more stable than the explicit ones, as will be shown numerically in Section 2.6.1.

It is worth noting that both (2.9) and (2.14) are variable coefficient elliptic problems in which the coefficients vary at every time step during the streamer simulations. Thus, the coefficient matrix must be computed and assembled in each time step, whereas it needs to be calculated only once in the constant case. When a preconditioned iterative elliptic solver is used, the preconditioner must also be renewed

in each step to solve the variable coefficient elliptic equation (again, this needs to be done only once in the constant case). The situation is similar if a direct solver is used. Therefore, in streamer simulations, solving a variable coefficient elliptic equation generally consumes more time than solving a Poisson equation with constant coefficients.

However, it is still not true to conclude that the second-order explicit scheme in (2.3)–(2.5) is faster than the second-order semi-implicit scheme in (2.12)–(2.13). As we will show in Section 2.6.3, the semi-implicit scheme achieves better performance than explicit schemes in many Krylov elliptic solvers even under the same time steps. Moreover, the semi-implicit schemes remove the dielectric relaxation time restriction, which may allow a larger time step on many occasions to further shorten the total computational time.

2.3 Spatial discretization

Finite volume method is applied for spatial discretization. The computational domain is decomposed by a uniform grid with M_x, M_y, M_z partitions in the x, y, z directions, respectively. Therefore, the grid size is characterized by $\Delta x = (x_1 - x_0)/M_x$, $\Delta y = (y_1 - y_0)/M_y$, $\Delta z = (z_1 - z_0)/M_z$. The grid cells are denoted by $I_{i,j,k} = [x_0 + i\Delta x, x_0 + (i+1)\Delta x] \times [y_0 + j\Delta y, y_0 + (j+1)\Delta y] \times [z_0 + k\Delta z, z_0 + (k+1)\Delta z]$, where $0 \le i \le M_x - 1$, $0 \le j \le M_y - 1$, and $0 \le k \le M_z - 1$. The finite volume method is used for the spatial discretization, and we define

$$(n_e)^n_{i,j,k} = \frac{1}{|I_{i,j,k}|} \int_{I_{i,j,k}} n_e^n(x, y, z) \mathrm{d}x\mathrm{d}y\mathrm{d}z. \tag{2.15}$$

Other notations, such as $(n_p)^n_{i,j,k}$ and $\phi^{n+1/2}_{i,j,k}$ are similarly defined. We adopt the classical second-order central scheme for (2.14). Let $P^n_{i,j,k}$ be the discrete coefficient of the elliptic problem (2.14) defined by

$$P^n_{i,j,k} = \lambda + \frac{\tau_n}{2} \left(\tilde{\mu}_p (n_p)^n_{i,j,k} + \tilde{\mu}_e (n_e)^n_{i,j,k} \right), \tag{2.16}$$

and denote

$$P^n_{i\pm1/2,j,k}=\frac{1}{2}(P^n_{i\pm1,j,k}+P^n_{i,j,k}),\ P^n_{i,j\pm1/2,k}=\frac{1}{2}(P^n_{i,j\pm1,k}+P^n_{i,j,k}),\ P^n_{i,j,k\pm1/2}=\frac{1}{2}(P^n_{i,j,k\pm1}+P^n_{i,j,k}).$$

Then, (2.14) is discretized as follows:

$$\begin{aligned}&-\frac{P^n_{i+1/2,j,k}\Delta_{+x}\phi^{n+1/2}-P^n_{i-1/2,j,k}\Delta_{-x}\phi^{n+1/2}}{(\Delta x)^2}\\&-\frac{P^n_{i,j+1/2,k}\Delta_{+y}\phi^{n+1/2}-P^n_{i,j-1/2,k}\Delta_{-y}\phi^{n+1/2}}{(\Delta y)^2}\\&-\frac{P^n_{i,j,k+1/2}\Delta_{+z}\phi^{n+1/2}-P^n_{i,j,k-1/2}\Delta_{-z}\phi^{n+1/2}}{(\Delta z)^2}\\=&n^n_p-n^n_e-\frac{\tau_n}{2}\left(\frac{\tilde{D}_{e,x}}{(\Delta x)^2}\delta^2_x n^n_e+\frac{\tilde{D}_{e,y}}{(\Delta y)^2}\delta^2_y n^n_e+\frac{\tilde{D}_{e,z}}{(\Delta z)^2}\delta^2_z n^n_e\right)\end{aligned}\tag{2.17}$$

where $\tilde{D}_{e,x}$, $\tilde{D}_{e,y}$ and $\tilde{D}_{e,z}$ are the diagonal entries of the dimensionless diffusion matrix $\tilde{D}_e$; the subscripts are neglected for the numerical solutions at $I_{i,j,k}$, e.g., n^n_p denotes $(n_p)^n_{i,j,k}$, $\Delta_{+x}\phi^{n+1/2}$ and $\Delta_{-x}\phi^{n+1/2}$ denote the forward and backward differences of $\phi^{n+1/2}_{i,j,k}$ in the x direction, respectively:

$$\Delta_{+x}\phi^{n+1/2}=\phi^{n+1/2}_{i+1,j,k}-\phi^{n+1/2}_{i,j,k},\ \Delta_{-x}\phi^{n+1/2}=\phi^{n+1/2}_{i,j,k}-\phi^{n+1/2}_{i-1,j,k},\tag{2.18}$$

and similar notations are applied for $\Delta_{\pm y}\phi^{n+1/2}$ and $\Delta_{\pm z}\phi^{n+1/2}$; $\delta^2_x n^n_e$ denotes the second-order central difference of $(n_e)^n_{i,j,k}$ in the x direction:

$$\delta^2_x n^n_e=(n_e)^n_{i+1,j,k}-2(n_e)^n_{i,j,k}+(n_e)^n_{i-1,j,k},\tag{2.19}$$

and similar notations are used for $\delta^2_y n^n_e$ and $\delta^2_z n^n_e$.

Afterwards, $\vec{E}$ can be calculated numerically by the central difference from the numerical solution of ϕ, and $|\vec{E}|$ can be evaluated accordingly. In some cases [7,137] where the mobility and diffusion coefficients depend on $|\vec{E}|$, interpolations can be applied to obtain $|\vec{E}|$ on the cell surfaces.

For the transport equations in (2.12)–(2.13), the second-order MUSCL scheme combined with the Koren limiter is applied [78,164]. Ghost cells are used for all the boundary conditions of n_e and n_p. This part of the spatial discretization is classical,

and we omit the details here. Generally, we expect second-order accuracy from this spatial discretization for smooth solutions.

Due to the explicit treatment in the temporal discretization of the drift and diffusion terms, all temporal schemes in Section 2.2 with the above FVM discretization should satisfy the following stability condition

$$\tau_n \sum_{\alpha=x,y,z} \left(\frac{C_\gamma \tilde{\mu}_e |(E_\alpha)^*|_{\max}}{\Delta\alpha} + \frac{2\tilde{D}_{e,\alpha}}{(\Delta\alpha)^2} \right) \leq 1, \tag{2.20}$$

where $(E_\alpha)^*$ denotes $E_\alpha^{n+1/2}$ for scheme (2.12)–(2.13), E_α^n for explicit schemes, and E_α^{n+1} for scheme (2.7); E_x, E_y and E_z are the components of the electric field $\vec{E} = (E_x, E_y, E_z)^T$, and the subscript "max" denotes the maximum value among all cells.

We have two remarks on (2.20). Firstly, the problem is convection dominated, and typically the restriction on the time step determined by the convection term is stricter than that of the diffusion term. Secondly, the drift velocity of electrons $\tilde{\mu}_e|\vec{E}|$ is typically one or two orders of magnitude larger than that of positive ions $\tilde{\mu}_p|\vec{E}|$, and therefore only the stability condition for n_e needs to be considered.

The constant C_γ in (2.20) depends on both time integration and space discretization methods [33, 34, 146]. The stability of the schemes can be analyzed by fixing the electric field and neglecting the source terms. With the MUSCL finite volume method as the space discretization, von Neumann analysis (e.g. [84, Chapter 20]) indicates the second-order methods in this paper (including the proposed semi-implicit method and the second-order explicit scheme) are linearly stable without limiters under the condition (2.20) with $C_\gamma = 1$. On the other hand, by Harten's theorem [63], it can be shown that the first-order methods (2.2) and (2.7) are stable under (2.20) with $C_\gamma = 2$ provided that a proper slope limiter is applied, e.g., the Koren limiter which is used in this chapter.

As a summary, we have discussed two constraints for the time step, one from the dielectric relaxation time (2.6), and the other from the convection and diffusion

(2.20). For explicit schemes (2.2) and (2.3)–(2.5), both conditions have to be satisfied; while for the proposed second-order semi-implicit (2.12)–(2.13) and the first-order semi-implicit scheme (2.7), it will be shown numerically in Section 2.6.1 that the constraint from the dielectric relaxation time can be relaxed. Which constraint is more restrictive depends on the problem setting and the spatial discretization. For the problems in which (2.20) gives a stricter constraint, all the aforementioned schemes require similar time steps. However, for those problems where the dielectric relaxation constraint is tighter, semi-implicit schemes allow larger time steps, so that we can better match the errors of temporal and spatial discretization to maximize the computational efficiency.

2.4 Multigrid preconditioned FGMRES elliptic solver

To solve (2.17), an iterative solver [97] is preferable to a direct solver. Although some state-of-the-art direct solvers retain the matrix sparsity to some degree, in three-dimensional simulations, a parallel direct solver still generally requires enormous amounts of memory, which is unaffordable when the number of DOFs becomes large and is therefore inapplicable. One example of using the parallel direct solver MUMPS can be found in [137].

In [75], the geometric multigrid method was shown to be faster than the SOR method for solving the Poisson equation in 2D streamer discharges. Moreover, the convergence rate of the SOR method depends on the relaxation factor; however, maintaining its optimality at each time step is difficult because the coefficients in elliptic problem (2.17) vary.

We use a geometric multigrid as a preconditioner rather than a solver because the geometric multigrid preconditioned Krylov subspace solver may be more stable and efficient than using a geometric multigrid alone. In [149], a multigrid method was shown to be divergent for high-order FEMs when used as a solver, but convergence was achieved when the multigrid was combined with the conjugate gradient method.

By investigating the eigenvalues of the iteration matrix, it was found in [128] that that while isolated large eigenvalues limit the convergence of a multigrid solver, the eigenvectors belonging to these large eigenvalues can be captured in a Krylov subspace constructed by GMRES within a few iterations, which accelerates the convergence of a multigrid solver. It was also shown in [149, 151] that the multigrid preconditioner combined with the conjugate gradient method is faster and more stable than the multigrid solver alone.

2.4.1 Preconditioned FGMRES solver

Using a geometric multigrid as the preconditioner, we find that the geometric multigrid-preconditioned flexible generalized minimal residual (FGMRES) solver is the best among various Krylov subspace solvers, as discussed in Section 2.6.3. The flowchart of the preconditioned FGMRES is shown in Algorithm 1 [140]. Hereafter, the notation $\|\cdot\|$ denotes 2-norm.

As shown in line [5] of Algorithm 1, for different basis vectors v_j, different preconditioning matrices $\bar{M}_j$ can be selected, which provides the "flexibility" reflected in the solver's name. The price is that the preconditioned vectors z_j in line [5] must be stored to form the matrix Z_m, resulting in a larger memory cost than is achieved by the classical generalized minimal residual (GMRES) method which stores only the vectors v_j. However, the flexibility resulting from different preconditioners helps to improve the robustness of the GMRES algorithm, as shown in [140]. In our FGMRES implementation, we initially set $m = 30$ and selected the multigrid as the preconditioner in line [5].

2.4.2 Multigrid preconditioner

A geometric multigrid preconditioner is chosen to accelerate the convergence of the FGMRES solver.

Algorithm 1: FGMRES with preconditioning to solve $Ax = b$.

[1] **1**. Initial guess x_0. Define a $(m+1) \times m$ zero matrix $\bar{H}_m = (h_{i,j})$, where m is a given number indicating the dimension of Krylov subspace and $h_{i,j}$ denotes the (i,j)-th entry of $\bar{H}_m$. Let $k \leftarrow m$.

[2] **2**. Arnoldi process:

[3] Compute $r_0 \leftarrow b - Ax_0$, $\beta \leftarrow \|r_0\|$ and $v_1 \leftarrow r_0/\beta$;

[4] **for** $j = 1, \cdots, m$ **do**

[5] Preconditioning: $z_j \leftarrow \bar{M}_j v_j$;

[6] Compute $\omega \leftarrow Az_j$;

[7] **for** $i = 1, \cdots, j$ **do**

[8] Gram-Schmidt process: $h_{i,j} \leftarrow (\omega, v_i)$, $\omega \leftarrow \omega - h_{i,j} v_i$;

[9] **end**

[10] Compute $h_{j+1,j} \leftarrow \|\omega\|$, $v_{j+1} \leftarrow \omega / h_{j+1,j}$;

[11] Compute $r_j \leftarrow \min_y \|\beta e_j - \bar{H}_j y\|$, where $\bar{H}_j$ is the upper left $(j+1) \times j$ sub-matrix of $\bar{H}_m$ and $e_j = [1, 0, \cdots, 0]^T$ with a total of $j+1$ entries;

[12] Check the stopping criterion. If satisfied, let $k \leftarrow j$ and go to line [14];

[13] **end**

[14] Define a matrix $Z_k \leftarrow [z_1, \cdots, z_k]$.

[15] **3**. Form iterative solution $x_k \leftarrow x_0 + Z_k y_k$, where $y_k = \arg\min_y \|\beta e_k - \bar{H}_k y\|$.

[16] **4**. Restart: If the stopping criterion is not satisfied, let $x_0 \leftarrow x_k$ and $k \leftarrow m$; go to line [2].

Our implementation of the geometric multigrid preconditioner uses a full multigrid (FMG) [75] for the first time step, and V-cycle multigrid afterwards. In the first time step, no previous information is available, and therefore we simply make a zero initial guess, with the expectation that FMG will achieve faster convergence. Subsequently, the potential ϕ calculated during the previous time step is adopted as the initial guess. Given a good initial guess, the cheaper V-cycle multigrid gives

better performance.

To introduce the multigrid preconditioner, we first provide a simple review of the multigrid solver [161, Chapter 2]. In the following, assume that the elliptic equation on grid level l is discretized as follows:

$$A_l x_l = b_l. \tag{2.21}$$

A diagram showing the procedure of the two-layer V-cycle multigrid method is given in Figure 2.1, where the subscripts 2 and 1 denote the second layer (the fine layer) and the first layer (the coarse layer) respectively. The restriction and prolongation are shown using a 2D example of 4×4 and 2×2 meshes. When solving the equation $A_1 d_1 = r_1$ shown at the bottom of the V-cycle, this multigrid procedure can be called recursively, resulting in a multi-layer multigrid solver.

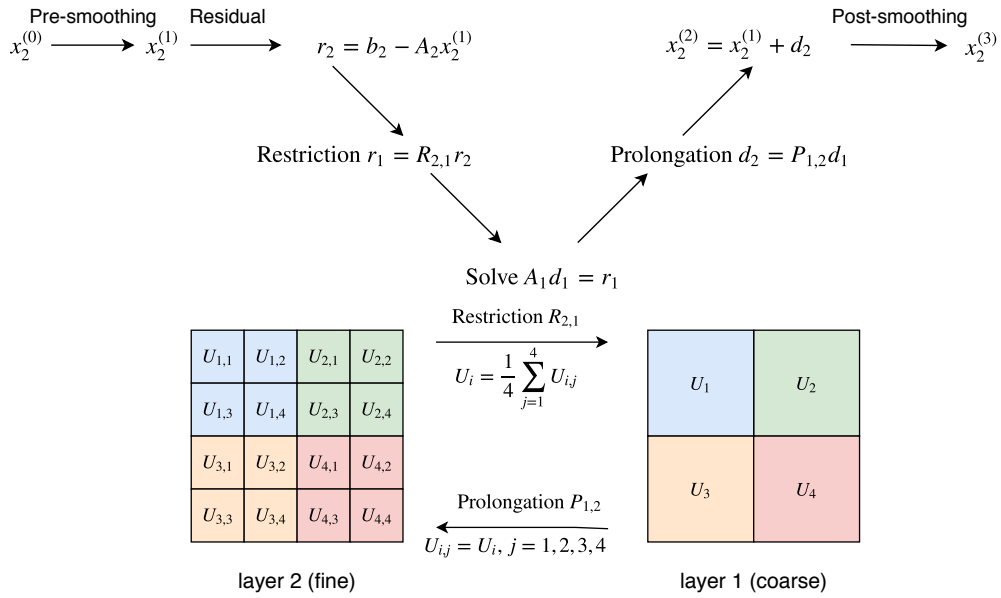


Figure 2.1: Diagram of V-cycle geometric multigrid for uniform mesh in two layers.

Some smoothers commonly used in sequential computation include the Gauss-Seidel method and the successive over relaxation (SOR) method [151]. However, when parallelized, the efficiency of these methods is impaired due to their sequential nature. Therefore, we adopt the Chebyshev smoother in our implementation, which

is a polynomial smoother based on Chebyshev polynomials. The performance of polynomial smoothers (including Chebyshev polynomials) and the parallel Gauss-Seidel smoother were compared in [1]; the results show that polynomial smoothers are preferable in a parallel environment. In general, given a polynomial $p_n(x)$ of degree n, the associated polynomial smoother in the pre-smoothing of the multigrid algorithm is

$$x_2^{(1)} = x_2^{(0)} + p_n(A_2)(b_2 - A_2 x_2^{(0)}), \tag{2.22}$$

which smooths out the error as follows:

$$x_2^{(1)} - A_2^{-1} b_2 = q_{n+1}(A_2)(x_2^{(0)} - A_2^{-1} b_2),$$

where $q_{n+1}(x) = 1 - x p_n(x)$. The Chebyshev smoother uses the following polynomials [19]:

$$q_{n+1}(x) = T_{n+1}\left(\frac{\lambda_{\max}(A_2) + \lambda^*(A_2) - 2x}{\lambda_{\max}(A_2) - \lambda^*(A_2)}\right) \Big/ T_{n+1}\left(\frac{\lambda_{\max}(A_2) + \lambda^*(A_2)}{\lambda_{\max}(A_2) - \lambda^*(A_2)}\right), \tag{2.23}$$

where $\lambda_{\max}(A_2)$ is the largest eigenvalue of A_2 and is usually replaced by an approximation of the largest eigenvalue in practice; $\lambda^*(A_2)$ is selected manually, both of which will be discussed later in this section; and $T_{n+1}(x)$ is the Chebyshev polynomial of degree $n+1$. $p_n(x)$ can be obtained accordingly. By introducing the two matrices $P_2 = p_n(A_2)$ and $Q_2 = q_{n+1}(A_2)$, we can re-write the pre-smoothing operation in (2.22) as follows:

$$x_2^{(1)} = Q_2 x_2^{(0)} + P_2 b_2. \tag{2.24}$$

Moreover, the Chebyshev iteration can be further improved to become a "preconditioned Chebyshev iteration" by introducing another preconditioner on top of it, for which we refer the readers to [19] for details.

The post-smoothing operation in Figure 2.1 is applied to $x_2^{(2)}$ in the same way as (2.24). Therefore, the V-cycle multigrid in Figure 2.1 is formulated as follows:

$$x_2^{(3)} = x_2^{(0)} + M_2(b_2 - A_2 x_2^{(0)}), \tag{2.25}$$

where $M_2 = Q_2P_2 + P_2 + Q_2P_{1,2}A_1^{-1}R_{2,1}(I_2 - A_2P_2)$ and I_2 is the identical matrix on the second layer. The derivation of (2.25) and the corresponding equation for the general multi-layer V-cycle method are shown as follows: one step of the V-cycle geometric multigrid solver with two layers in Figure 2.1 can be expressed as follows:

$$x_2^{(3)} = Q_2[Q_2x_2^{(0)} + P_2b_2 + P_{1,2}A_1^{-1}R_{2,1}(b_2 - A_2Q_2x_2^{(0)} - A_2P_2b_2)] + P_2b_2, \tag{2.26}$$

which represents the iteration between $x_2^{(0)}$ and $x_2^{(3)}$. Simplifying (2.26) we obtain

$$x_2^{(3)} = [Q_2^2 - Q_2P_{1,2}A_1^{-1}R_{2,1}A_2Q_2]x_2^{(0)} + [Q_2P_2 + P_2 + Q_2P_{1,2}A_1^{-1}R_{2,1}(I_2 - A_2P_2)]b_2. \tag{2.27}$$

Then we denote $M_2 = Q_2P_2 + P_2 + Q_2P_{1,2}A_1^{-1}R_{2,1}(I_2 - A_2P_2)$. Note that here, M_2 is the same as the one defined in (2.25). The matrix in front of $x_2^{(0)}$ in (2.27) is

$$\begin{aligned} Q_2^2 - Q_2P_{1,2}A_1^{-1}R_{2,1}A_2Q_2 &= I_2 + Q_2^2 - I_2 - Q_2P_{1,2}A_1^{-1}R_{2,1}A_2Q_2 \\ &= I_2 + (Q_2 + I_2)(Q_2 - I_2) - Q_2P_{1,2}A_1^{-1}R_{2,1}A_2Q_2 \\ &= I_2 + (Q_2 + I_2)(-P_2A_2) + Q_2P_{1,2}A_1^{-1}R_{2,1}A_2(P_2A_2 - I_2), \end{aligned} \tag{2.28}$$

where $Q_2 = I_2 - P_2A_2$ is used in the last line of (2.28) by denoting of $Q_2 = q_{n+1}(A) = (I_2 - p_n(A)A)$ and $P_2 = p_n(A)$. Then,

$$\begin{aligned} &I_2 + (Q_2 + I_2)(-P_2A_2) + Q_2P_{1,2}A_1^{-1}R_{2,1}A_2(P_2A_2 - I_2) \\ =&I_2 - (Q_2P_2 + P_2)A_2 + Q_2P_{1,2}A_1^{-1}R_{2,1}(A_2P_2 - I_2)A_2 \\ =&I_2 - [(Q_2P_2 + P_2) + Q_2P_{1,2}A_1^{-1}R_{2,1}(I_2 - A_2P_2)]A_2 = I_2 - M_2A_2. \end{aligned} \tag{2.29}$$

Now it is clear that the V-cycle geometric multigrid method (2.27) is

$$x_2^{(3)} = [I_2 - M_2A_2]x_2^{(0)} + M_2b_2 = x_2^{(0)} + M_2(b_2 - A_2x_2^{(0)}), \tag{2.30}$$

which is a Richardson iteration with the multigrid preconditioner M_2. (2.30) shows (2.25). If more than one layer is taken in a multigrid, we can obtain the preconditioner matrix recursively. The only difference is that we use the multigrid again to obtain the solution on the coarse mesh rather than the inverse of the matrix.

Therefore, the multigrid preconditioner matrix M_l for l layers V-cycle multigrid can be expressed recursively as follows:

$$\begin{aligned} M_1 &= A_1^{-1}, \\ M_l &= Q_l P_l + P_l + Q_l P_{l-1,l} M_{l-1} R_{l,l-1}(I_l - A_l P_l), \qquad (l \geq 2). \end{aligned} \tag{2.31}$$

In our implementation, the quadratic-polynomial preconditioned Chebyshev smoother is applied to both pre-smoothing and post-smoothing, using the one-step local symmetric successive over relaxation method (SSOR) as the preconditioner. The values of $\lambda_{\max}(A_l)$ and $\lambda^*(A_l)$, which are required in the Chebyshev iteration (see (2.23)), are estimated by

$$\lambda_{\max}(A_l) \approx 1.1\lambda_{\max}(H_m), \qquad \lambda^*(A_l) = 0.1\lambda_{\max}(H_m),$$

where H_m is the upper Hessenberg matrix $(h_{i,j})_{m\times m}$ obtained by applying the Arnoldi process (without preconditioning) to A_l, as shown in Algorithm 1. We use $m = 10$ for the eigenvalue estimate. A sequential direct solver is applied to the coarsest mesh.

When the multigrid is used in the "Preconditioning" step in line 5 of Algorithm 1, we apply the multigrid preconditioner $\bar{M}_j$ to a vector v_j by setting a zero initial $x_2^{(0)} = \vec{0}$ and $b_2 = v_j$ in the multigrid solver, and run the V-cycle once. The output will be the preconditioned vector $z_j = \bar{M}_j v_j$, turning a multigrid solver into a preconditioner for Krylov subspace solvers. More details of the preconditioning process can be found in, e.g., [1, 151].

2.5 MPI based parallel implementation

As a parallel simulator, we would like to brief the implementation of MPI parallelization in this chapter. It follows the standard procedure of solving PDE on a uniform structured mesh.

Most High Performance Computing (HPC) platforms support MPI, and a variety of implementations are available. MPI supports parallel computing using thousands

of cores, to fully utilize the power of modern clusters. MPI is responsible for the communication between different processes by sending and receiving messages. A group of processes is defined in a communicator; each of which has a unique rank. Processes communicate with one another using messages that include the process rank and a tag. Moreover, MPI supports collective communication, to optimally broadcast messages from one process to all the other processes.

In our implementation, we partition the 3D grid into Cartesian subgrids of equal size in each direction. Each process stores only the portion of the solutions defined in one of the subgrids. To reduce the communication latency, for each subgrid, the number of cells in each direction is approximately equal. For example, suppose we have a uniform mesh of $256 \times 256 \times 320$ cells, which is to be distributed to 80 processes. We first decompose the entire grid into $4 \times 4 \times 5$ subgrids; each process will own a cubic subgrid with a size of 64^3.

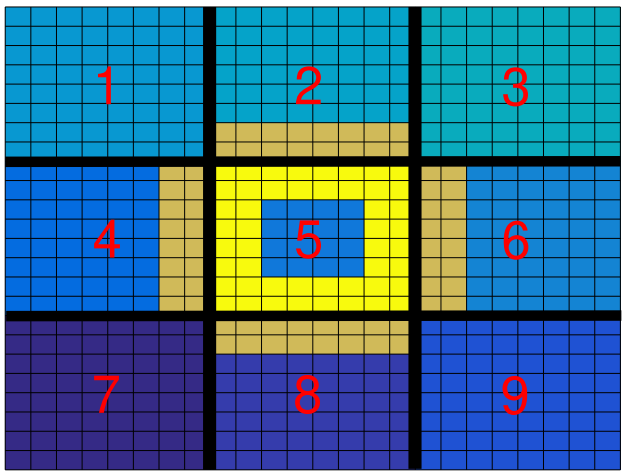


Figure 2.2: Explanation of communication for n_e, n_p in a 2D case.

To apply the MUSCL scheme, each process requires inter-process communication to retrieve the values of n_e and n_p on the two adjacent layers of cells from its neighbouring processes. For illustrative purposes, we only show the communication in 2D cases in Figure 2.2. Figure 2.2 includes 9 processes labeled by their ranks.

Here, we focus only on the 5th process, whose subgrid is located in the interior of the domain, and stores an 8×8 subgrid. To update the solutions on the top two rows of the subgrid using the MUSCL method, a 2×8 stripe of cells of unknowns from process 2 (indicated by the dark yellow) is required. Similarly, 2×8 or 8×2 strips of cells of unknowns from processes 4, 6, 8 are required to update the solutions on the other light-yellow grid cells. Similarly, to update the solution in processes 2, 6, 8 and 4, four stripes of 2×8 or 8×2 cells of unknowns from process 5 must be sent to the corresponding processes. Generally, for a 3D uniform grid, an interior subgrid whose size is $M \times N \times P$ should receive $4(MN + NP + MP)$ cells of unknowns from the surrounding processes and should send the same number of unknowns to them. Therefore, the local communication/computation ratio can be characterized as following:

$$\frac{\text{communication}}{\text{computation}} = \frac{4(MN + NP + MP)}{MNP} \leq \frac{12}{\min\{M, N, P\}}, \tag{2.32}$$

showing that the communication cost is one order of magnitude lower than the computation for interior processes. We use ghost cells to address the boundary conditions; therefore, the communication required for processes handling boundary conditions is less than the communication required for interior processes.

The communication for solving the potential ϕ is similar as Figure 2.2, but the communication stripes have a width of only one cell in most layers of the multigrid. The only exception is that at the coarsest layer of the multigrid preconditioner where a direct solver is used, gathering and broadcasting operations are still needed for a small amount of data.

2.6 Numerical comparisons

In this section, we first adopt a 1D dimensionless model to illustrate the convergence order and stability of our second-order semi-implicit scheme. Then, we use a 3D problem to show the scalability of our simulator, and make a comparison between several algebraic elliptic solvers.

2.6.1 Comparison on convergence and stability

The following dimensionless model problem in 1D, which has a form similar to (1.6), is adopted to show that the proposed semi-implicit scheme is second-order accurate in time, and is more stable than the explicit schemes:

$$\begin{cases} \partial_t n_e - \partial_x(\tilde{\mu}_e E n_e) - \tilde{D}_e \partial_{xx}(n_e) = S\exp(-K/|E|)\tilde{\mu}_e|E|n_e, \\ \partial_t n_p + \partial_x(\tilde{\mu}_p E n_p) = S\exp(-K/|E|)\tilde{\mu}_e|E|n_e, \qquad x \in I, \quad t > 0, \\ -\lambda\partial_{xx}\phi = n_p - n_e, \ E = -\partial_x\phi, \end{cases} \tag{2.33}$$

where $I = (0,1)$. Dirichlet boundary conditions $\phi(0,t) = 1$ and $\phi(1,t) = 0$ are applied for $\phi = \phi(x,t)$, while homogeneous Neumann boundary conditions are applied at all boundaries for $n_e = n_e(x,t)$ and at inflow boundary for $n_p = n_p(x,t)$. Parameters in (2.33) are given by $\tilde{\mu}_e = 1$, $\tilde{\mu}_p = 0.09$, $\tilde{D}_e = 10^{-4}$, $S = 1000$ and $K = 4$, while constant λ will be given later. Constant time steps $\tau_n = \tau$ are used, and the computation is performed until $T = 0.05$. It should be mentioned that the dimensionless dielectric relaxation time constraint for (2.33) is same as (1.7), which is

$$\tau \le \tau_{\text{diel}} = \frac{\lambda}{\max(\tilde{\mu}_p n_p + \tilde{\mu}_e n_e)}. \tag{2.34}$$

Different temporal schemes introduced in Section 2.2 are implemented with the same spatial discretization.

Study of convergence In this testing example, we set $\lambda = 10^{-3}$ in (2.33). The initial value is $n_e(x, t=0) = n_p(x, t=0) = 10^{-6} + 0.1\exp(-100(x-0.5)^2)$. For all the calculations in this example, we fix the ratio of the time step to the grid size at $\tau/\Delta x = 0.25$. The finite volume method with unlimited linear reconstruction is applied for spatial discretization. The "exact solution" for $(n_e)_{\text{ref}}$ and $(n_p)_{\text{ref}}$ are calculated by second-order explicit scheme (2.3)–(2.5) with $\tau = 0.005/2^8$ which is sufficiently small. The numeric results are given in Tables 2.1 and 2.2.

Table 2.1: L^2-norm error of the second-order semi-implicit scheme (2.12)–(2.13) in the 1D testing problem.

Δt	0.005	0.005/2	0.005/2^2	0.005/2^3	0.005/2^4	0.005/2^5
$\|n_e - (n_e)_{\text{ref}}\|$	3.1660×10^{-4}	8.3832×10^{-5}	2.1650×10^{-5}	5.5097×10^{-6}	1.3876×10^{-6}	3.4508×10^{-7}
order	–	1.9171	1.9531	1.9743	1.9894	2.0075
$\|n_p - (n_p)_{\text{ref}}\|$	2.5303×10^{-4}	6.3421×10^{-5}	1.6028×10^{-5}	4.0412×10^{-6}	1.0134×10^{-6}	2.5158×10^{-7}
order	–	1.9962	1.9844	1.9877	1.9956	2.0101

Table 2.2: L^2-norm error of the first-order semi-implicit scheme (2.7) in the 1D testing problem.

Δt	0.005	0.005/2	0.005/2^2	0.005/2^3	0.005/2^4	0.005/2^5
$\|n_e - (n_e)_{\text{ref}}\|$	1.7405×10^{-3}	9.9821×10^{-4}	5.3935×10^{-4}	2.8098×10^{-4}	1.4349×10^{-4}	7.2514×10^{-5}
order	–	0.8021	0.8881	0.9408	0.9696	0.9846
$\|n_p - (n_p)_{\text{ref}}\|$	1.5186×10^{-3}	8.8187×10^{-4}	4.8021×10^{-4}	2.5122×10^{-4}	1.2856×10^{-4}	6.5044×10^{-5}
order	–	0.7841	0.8769	0.9347	0.9665	0.9830

The results in Tables 2.1 and 2.2 clearly demonstrate that the proposed semi-implicit scheme (2.12)–(2.13) is indeed second order time accurate, while the previously used semi-implicit scheme (2.7) is only first order time accurate, though second-order spatial discretization is used.

It is worth emphasizing that the proposed second-order semi-implicit scheme needs to solve the elliptic equation only once during each time step, the same as the first-order semi-implicit scheme. To gain second order time accuracy, the only additional cost is an explicit stage for n_p and n_e at each time step, which is relatively cheap compared with solving the elliptic equation.

Study of stability in terms of the dielectric relaxation time restriction To show that the semi-implicit scheme (2.12)–(2.13) is able to alleviate the dielectric relaxation time restriction, and is thus more stable than explicit schemes, we perform the calculation with $\lambda = 10^{-5}$ in (2.33). The initial value is $n_e(x, t = 0) =$

$n_p(x, t = 0) = 10^{-6} + \exp(-100(x - 0.5)^2)$. Koren limiter is applied in finite volume discretization.

Note the goal of this example is to check the instability raised by the dielectric relaxation time restriction. As discussed after (2.20), the constraints of the time step include the dielectric relaxation time constraint and the CFL-type stability condition. In this 1D setting, they can be represented, respectively, by (2.34) and

$$\tau \le \tau_{\text{CFL}} = \frac{\Delta x}{C_\gamma \tilde{\mu}_e |E|_{\max} + 2\tilde{D}_e/\Delta x}. \tag{2.35}$$

To get a good estimation of τ_{diel} and τ_{CFL} for this test problem, we first perform the simulation on a very fine mesh $\Delta x = 1/12800$ with $\tau = 1/128000$, using second-order explicit scheme (2.3)–(2.5), and record the maximum values of $|E|$ and $(\tilde{\mu}_p n_p + \tilde{\mu}_e n_e)$ throughout the simulation. Such results are considered to have sufficient accuracy, so that we can use these values to get a precise estimation of τ_{diel} defined in (2.34), and the result is $\tau_{\text{diel}} = 9.1736 \times 10^{-6}$. To estimate τ_{CFL}, we insert the estimated value of $|E|_{\max}$ into (2.35), with C_γ set to be 2 and Δx chosen as a relatively larger cell size $\Delta x = 1/400$, which yields $\tau_{\text{CFL}} = 4.7448 \times 10^{-4}$. The numerical tests presented below will be carried out on the uniform grid with $\Delta x = 1/400$. Thus we have $\tau_{\text{CFL}} \approx 52\tau_{\text{diel}}$, meaning that a much better efficiency can be achieved if we can break the dielectric relaxation time constraint.

Five different time steps, i.e., $0.5\tau_{\text{diel}}$, τ_{diel}, $3\tau_{\text{diel}}$, $10\tau_{\text{diel}}$ and $50\tau_{\text{diel}}$, are used to test the stability. All these five time steps are less than τ_{CFL}, and stability condition (2.35) is always satisfied for all simulations before numerical blow-up occurs. We consider a simulation to be unstable if $n_e > 10$ is detected in this example. According to our experiments, this condition always leads to a quick numerical blow-up of the solution. In addition to the proposed semi-implicit scheme (2.12)–(2.13), we implemented three other temporal discretizations for comparisons: the first-order explicit scheme (2.2), the first-order semi-implicit scheme (2.7), and second-order explicit scheme (2.3)–(2.5).

The results in Table 2.3 clearly show that the two semi-implicit methods remain stable when the time step exceeds τ_{diel} and reaches $50\tau_{\text{diel}}$. In contrast, the two

Table 2.3: Stability of different temporal discretizations on a 1D testing problem.

Temporal scheme	$\tau = 0.5\tau_{\text{diel}}$	$\tau = \tau_{\text{diel}}$	$\tau = 3\tau_{\text{diel}}$	$\tau = 10\tau_{\text{diel}}$	$\tau = 50\tau_{\text{diel}}$
2^{nd} order semi-implicit (2.12)–(2.13)	stable	stable	stable	stable	stable
1^{st} order semi-implicit (2.7)	stable	stable	stable	stable	stable
2^{nd} order explicit (2.3)–(2.5)	stable	stable	unstable	unstable	unstable
1^{st} order explicit (2.2)	stable	stable	unstable	unstable	unstable

explicit methods exhibit instability. As indicated, the stability condition (2.35) is still fulfilled for all the time steps and schemes. Therefore the instability is caused by the violation of the dielectric relaxation time constraint, and the semi-implicit schemes truly allow larger time steps on this occasion.

Although we focus on the stability in this example, we can calculate the L^2 errors for all stable results in Table 2.3 using the numerical solution on the fine mesh as the reference solution. The errors of stable results for all time steps and schemes are at the order of 10^{-4} (ranging from 8.1734×10^{-5} to 9.0213×10^{-3}), indicating that all stable results in Table 2.3 still make reasonable predictions even when a relatively large time step $\tau = 50\tau_{\text{diel}}$ is used with a coarse mesh size $\Delta x = 1/400$.

As indicated in Section 2.3, the dielectric relaxation time constraint is not always the tightest time step constraint generally. However, the newly proposed semi-implicit method provides an alternative other than the explicit schemes. In addition, it requires solving the elliptic equation only once during each time step, while the explicit scheme (2.3)–(2.5) requires twice, which outweigh its possible drawback in the slower computation of the variable coefficient elliptic equation, as will be shown in Section 2.6.3.

2.6.2 Scalability of MPI parallelization

The scalability means the ability to reduce the execution time as the number of cores increases. Scalability can be measured by speedup S_p, which is the ratio of the execution time of the sequential program to the execution time of the parallel

program over p processes.

We developed our codes based on the well-known PETSc [8]. PETSc contains data structures and routines for both scalable and parallel solutions of partial differential equations, and it supports MPI parallelism. The simulations were performed on the cluster Tianhe2-JK located at Beijing Computational Science Research Center. It includes 514 computational nodes, each of which is equipped with two Intel Xeon E5-2660 v3 CPUs (10 cores, 2.6 GHz) and 192 GB of memory. The nodes are connected by a TH high-speed network interface. More details can be found at `https://www.csrc.ac.cn/en/facility/cmpt/2015-05-07/8.html`.

Unless otherwise stated, we used the following setup for a double-headed streamer in a homogeneous field between two parallel planes (at atmospheric pressure $p = 760\,\mathrm{Torr}$, with applied voltage $\phi_0 = 52\,\mathrm{kV}$ between $1\,\mathrm{cm}$, which are typical values introduced in Section 1.2.2) in following testings and applications in this chapter,

$$n_0(\vec{x}) = 10^{-6} + \exp\left(-\left(\frac{z-0.5}{\sigma_z}\right)^2 - \left(\frac{(x-1)^2+(y-1)^2}{\sigma_r^2}\right)\right), \tag{2.36}$$

where $\sigma_z = 0.027$ and $\sigma_r = 0.021$.

The convergence criterion for the iterative algebraic elliptic solver $A\tilde{\phi} = \beta$ is given by a tolerance of the relative residual, i.e., the iteration continues until

$$\frac{\|A\tilde{\phi} - \beta\|_2}{\|\beta\|_2} < \varepsilon, \tag{2.37}$$

where the tolerance ε is set to $\varepsilon = 10^{-8}$ in all the simulations hereafter.

The scalability of our simulator, which is the second-order semi-implicit scheme (2.12)–(2.13) combined with multigrid preconditioned FGMRES elliptic solver, is tested. We consider a domain $\Omega = (0,1) \times (0,1) \times (0,1)$, with three different mesh sizes: $256 \times 256 \times 320$, $512 \times 512 \times 640$ and $1024 \times 1024 \times 1280$. The time step is chosen to be proportional to the mesh size, which is $\tau_n = \Delta z / v_{\mathrm{ch}}$ with v_{ch} being the maximum characteristic speed. Here we choose $v_{\mathrm{ch}} = 3\tilde{\mu}_e |E_z|$ and $E_z = 4$ to ensure stability. Using a fixed time step $\tau = \tau_n$, we execute the code for 50 time steps, and record the elapsed wall-clock time for the whole run. Additionally, to obtain a more

reliable result, we execute the same code five times and take the average at each mesh size.

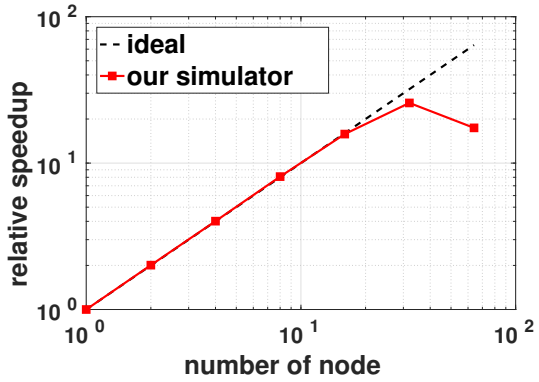
The code is executed over different numbers of nodes, using all 20 cores in each node. This mode (in which all available cores are used in each node) is called the "compact mode" in [137]. The average elapsed times are given in Table 2.4.

Table 2.4: Mean time (s) for 50 time steps on three different meshes, using second-order semi-implicit scheme with multigrid preconditioned FGMRES, with 20 cores in each node. Mesh 1, Mesh 2 and Mesh 3 refer to $256 \times 256 \times 320$, $512 \times 512 \times 640$ and $1024 \times 1024 \times 1280$, respectively.

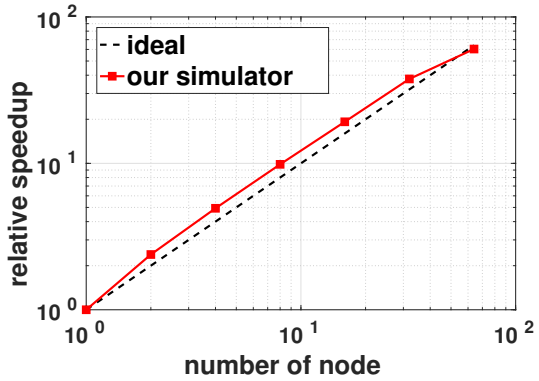
Number of nodes	1	2	4	8	16	32	64	128
Mesh 1	299.94	149.31	74.743	37.093	19.003	11.645	17.234	–
Mesh 2	3007.2	1261.3	609.70	305.39	156.41	79.722	49.806	–
Mesh 3	–	–	–	3753.5	1375.1	560.89	305.13	181.97

Note that the times shown in the tables are the average values over five runs, each with 50 time steps, instead of the average times for a single time step. The data are summarized in Figure 2.3 for clarity, where relative speedup denotes the speedup with respect to the execution time using the smallest number of nodes in Table 2.4.

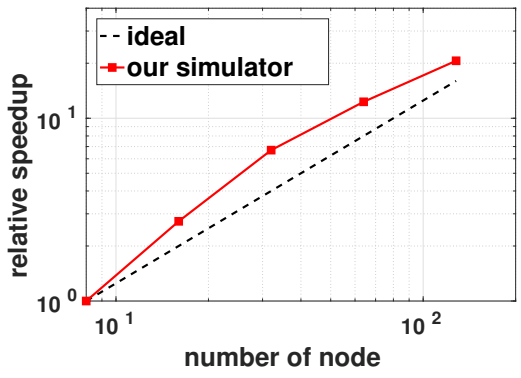
Generally the satisfactory scalability of our program can be seen in Figure 2.3. Figure 2.3(a) shows a nearly linear speedup when the number of nodes is small. When 32 or more nodes are used, the speedup obviously becomes sublinear. In particular, the time consumed for 64 nodes is even larger than that for 32 nodes. The reason is that the ratio of communication to computation becomes larger as the number of nodes increases. This tendency is visible in Figures 2.3(b) and 2.3(c) even though the larger computational work load postpones the significant drop of the parallel efficiency. It is interesting that in these two figures, we sometimes obtain a performance even better than the ideal case. One possible explanation for this phenomenon is that when the number of nodes is small, each node is heavily loaded,



(a) mesh size $256 \times 256 \times 320$



(b) mesh size $512 \times 512 \times 640$



(c) mesh size $1024 \times 1024 \times 1280$

Figure 2.3: Scalability of the second-order semi-implicit scheme using the multigrid preconditioned FGMRES solver over three meshes. Summarized from Table 2.4.

causing a lower cache hit ratio [68, Overview & Chapter 1]; while for each process, the amount of data required for communication is relatively large, causing more network latency. Such a phenomenon can also be observed clearly in Table 2.4.

2.6.3 Comparison of different algebraic elliptic solvers

In this subsection, we first study the performance of multigrid preconditioned FGMRES solver and then compare it with R&B SOR and other multigrid preconditioned Krylov subspace methods.

We again adopt the double-headed streamer in homogeneous field for testing purposes, using the same configuration as in Section 2.6.2 and three different mesh sizes $256 \times 256 \times 320$, $512 \times 512 \times 640$ and $1024 \times 1024 \times 1280$. As mentioned in Section 2.4.2, a zero initial guess and the FMG preconditioner are used in the first time step; subsequently, the V-cycle multigrid preconditioner is applied. We simulate the double-headed streamer until 2.5 ns (dimensionless $T \approx 4.9605 \times 10^{-2}$), using a fixed time step of $\Delta t_n = 2\,\text{ps}$ ($\tau_n \approx 3.9684 \times 10^{-5}$). Therefore, 1250 time steps are required to finish the simulation.

We execute our program on 640 cores, distributed among 32 nodes, with 20 cores on each node. We record the maximum wall-clock time consumed by the elliptic solver over all cores, including both the computation and communication in the solver as well as the assembly of the coefficient matrix and the right-hand side. The total times consumed by the elliptic solver are 320.76 s, 1813.2 s and 14001 s for the three aforementioned mesh sizes, respectively, and these do not exceed linear growth with the number of DOFs. Note that these times do not include the computation for quantities other than ϕ.

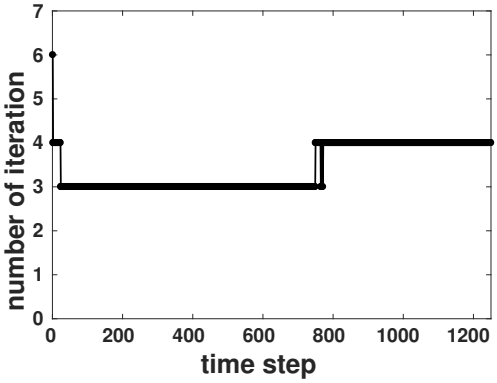
The number of iterations at each time step, for the multigrid preconditioned FGMRES solver in second-order semi-implicit scheme (2.12)–(2.13), is shown in Figure 2.4, with the tolerance of the residual is set to 10^{-8}. Except the first time step, the elliptic solver requires only 2 to 4 iterations, and the number of iterations does not increase as the mesh is refined. Figure 2.5 shows the rapid reduction of the

relative residual.

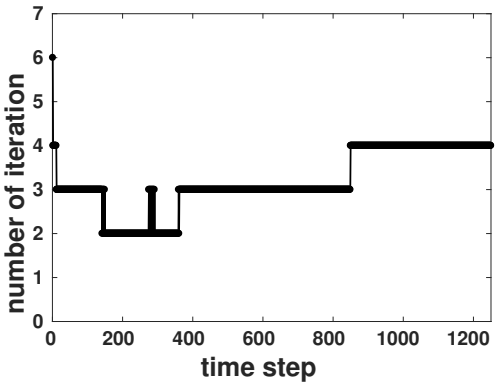
Next, we compare our FGMRES solver with other multigrid preconditioned Krylov subspace methods [141, Chapter 6–9], including Conjugate Gradient (CG), Conjugate Gradient Squared (CGS), Bi-CGSTAB (BiCGSTAB), GMRES, and Flexible Conjugate Gradients (FCG). For all these algebraic solvers, we use the same geometric multigrid preconditioner described in Section 2.4.2, with the same convergence criterion $\varepsilon = 10^{-8}$. From the previous test, we can see that the number of iteration steps does not vary significantly in the evolutionary process. Hence, for a quick test, we run the same simulation using the same parallel settings but for only 50 steps with a slightly larger time step ($\tau_n = 6.5104 \times 10^{-5}$). Table 2.5 shows the times consumed by the different Krylov subspace solvers; each time is the average of five runs. Manifestly, FGMRES works best with the multigrid preconditioner, consuming the least computation time in all cases.

Table 2.5: Time costs of different elliptic solvers using 640 cores over 50 time steps using the proposed second-order semi-implicit scheme.

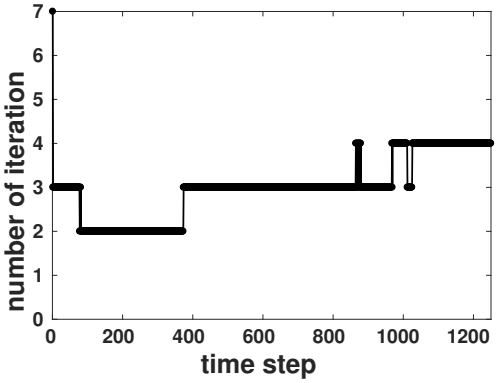
Mesh size: $256 \times 256 \times 320$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time (s)	10.589	12.588	12.584	12.415	14.084	14.044
Ratio (on FGMRES)	1	1.1888	1.1884	1.1724	1.3301	1.3263
Mesh size: $512 \times 512 \times 640$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time (s)	75.401	91.058	91.696	91.913	103.19	103.31
Ratio (on FGMRES)	1	1.2076	1.2161	1.2190	1.3685	1.3701
Mesh size: $1024 \times 1024 \times 1280$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time (s)	511.38	654.50	655.65	659.29	741.57	740.00
Ratio (on FGMRES)	1	1.2799	1.2821	1.2892	1.4501	1.4471



(a) mesh size $256 \times 256 \times 320$

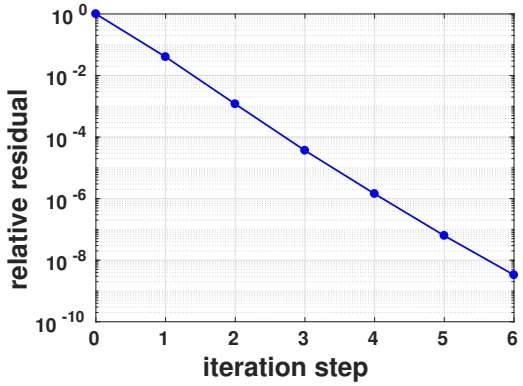


(b) mesh size $512 \times 512 \times 640$

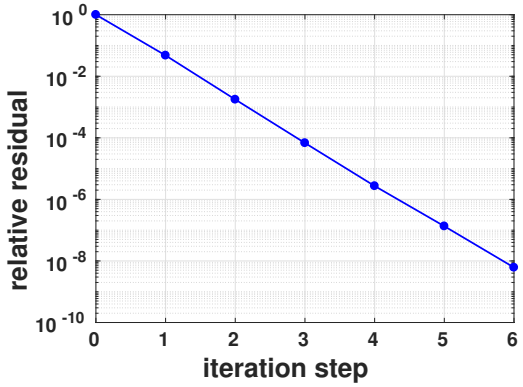


(c) mesh size $1024 \times 1024 \times 1280$

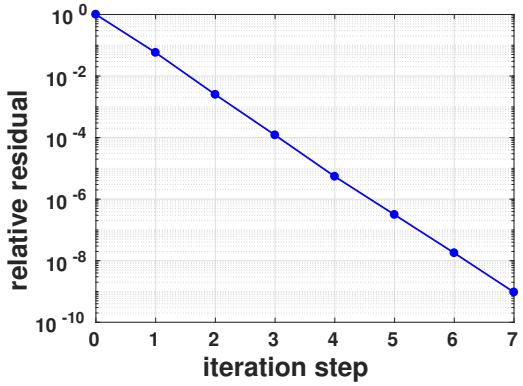
Figure 2.4: Iteration step for multigrid preconditioned FGMRES solver at each time step by the second-order semi-implicit scheme, on three different meshes.



(a) 1st step, mesh $256 \times 256 \times 320$



(b) 1st step, mesh $512 \times 512 \times 640$



(c) 1st step, mesh $1024 \times 1024 \times 1280$

Figure 2.5: Relative residual at each iteration for the multigrid preconditioned FGMRES solver in first step on three different meshes.

We also implement and test the second-order explicit scheme (2.3)–(2.5) using the same configurations. The results are shown in Table 2.6, and the FGMRES solver still performs outstandingly. Here we emphasize again that in the explicit scheme, we need to solve the constant coefficient Poisson equation that requires matrix assembly only at the initial step. Even so, for most of the Krylov subspace solvers, the times shown in Table 2.6 are larger than the times in Table 2.5. This result occurs because in each time step, our second-order semi-implicit method needs to solve the elliptic problem only once, while the explicit method needs twice, provided that our solver is efficient for both coefficient constant or varied Poisson equation. The only exception is the FGMRES method, in which the explicit scheme has better performance.

Table 2.6: Time costs of different elliptic solvers using 640 cores over 50 time steps by the second-order explicit scheme.

Mesh size: $256 \times 256 \times 320$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time [s]	10.306	13.550	13.724	13.941	17.039	17.014
Ratio (on FGMRES)	1	1.3148	1.3317	1.3527	1.6533	1.6509
Mesh size: $512 \times 512 \times 640$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time [s]	61.028	94.798	95.294	95.398	119.30	119.61
Ratio (on FGMRES)	1	1.5534	1.5615	1.5632	1.9548	1.9599
Mesh size: $1024 \times 1024 \times 1280$						
Method	FGMRES	GMRES	CG	FCG	CGS	BiCGSTAB
Mean time [s]	352.88	694.12	689.98	695.97	867.26	865.01
Ratio (on FGMRES)	1	1.9670	1.9553	1.9723	2.4577	2.4513

As Tables 2.5 and 2.6 show, the time consumption is roughly proportional to the number of DOFs, indicating the excellent performance of the multigrid preconditioner. In fact, the mean time for all the solvers scales sublinearly. One possible

reason is that the average number of iterative steps for a small grid size may be larger, as shown in Figure 2.4. In addition, the communication time may grow only sublinearly, especially when the amount of data being transferred is small.

Our results also show that the proposed algebraic elliptic solver outperforms the R&B SOR solver introduced in [137], which consumes approximately 55 seconds on 200 cores to converge to a relative residual lower than 10^{-6} on an 800^3 grid (see Figure 6(c) in [137]). For comparison purposes, we assume a linear speedup for R&B SOR; then, the same solver costs $55/(640/200) \approx 17.2$ seconds on 640 cores. In our numerical test, even for a much larger grid size $1024 \times 1024 \times 1280$ ($\approx 2.62 \times 800^3$), solving a single linear system to a tolerance of 10^{-8} requires only $352.88/(50 \times 2) = 3.52$ seconds. Here, we use the value from Table 2.6 because the constant coefficient Poisson equation was solved in [137]. In addition, Figure 6(b) and Figure 6(c) in [137] imply that the time complexity of the R&B SOR method is higher than $O(N)$(even higher than $O(N \log N)$) where N is the total number of DOFs. It should be remarked that the performance depends on the hardware and network, and this comparison is based on the data obtained on two independent high performance clusters. The cluster in [137] is built using Intel Xeon E5-2680 v2 processors (10 cores, 2.8 GHz) and that of Tianhe2-JK are Intel Xeon E5-2660 v3 CPUs (10 cores, 2.6 GHz).

2.7 Applications

In this section, we carry out two applications with different initial settings to compare the 3D results with the results obtained by the 2D moving mesh method, and to study the interactions of two streamers. The semi-implicit method (2.12)–(2.13) with adaptive time stepping (2.20) is used. When choosing the time step for our second-order semi-implicit scheme, we are unable to choose τ_n according to (2.20) directly because τ_n should be given before we solve $\phi^{n+1/2}$ in (2.17), and $\vec{E}^{n+1/2}$ can be computed only after obtaining $\phi^{n+1/2}$. Therefore, as an alternative,

we may choose a relatively small τ_0 at the first step, and then update time step by (2.20) with $\vec{E}^{n+1/2}$ is replaced as $\vec{E}^{n-1/2}$ and multiplying a safety factor 0.5 to it. We use simplified conditions without considering detailed models like the chemical reactions, because we aim towards code verification and proof of principle, which is by itself already very challenging for streamer simulations.

2.7.1 Double-headed streamer propagation

The first application is the double-headed streamer in a homogeneous field. A computational domain $\Omega = (0,1) \times (0,1) \times (0,1)$ is used; however, the numerical results are nearly indistinguishable from the results computed on a larger domain $(0,2) \times (0,2) \times (0,1)$. The initial charge $n_0(\vec{x})$ is a Gaussian located at the center of the domain (see (2.36)).

A very fine mesh with $2048 \times 2048 \times 2560$ cells is used in the simulation, comprising a total of more than 10.7 billion cells. For MPI parallelism, 1280 CPU cores (64 nodes) are used. The initial time step is set to 2 ps ($\tau_n \approx 3.9684 \times 10^{-5}$), and the adaptive time step is subsequently selected. The simulation requires 1558 steps to reach the final time 2.5 ns, resulting in an average time step of 1.6 ps.

Figure 2.6 shows the electric field along the z-axis at the center of the streamer channel ($x = y = 0.5$). The result in [14], which used moving mesh method in 2D simulation, is provided as a reference, and the two are in close agreement. It should be noted that [14] assumes streamer to be axisymmetric and therefore solve the 3D problem (1.6) in 2D. At $t = 2$ ns, the positive (cathode-directed) and the negative (anode-directed) streamer move approximately 0.19 and 0.26 cm, respectively, measured using the position of the highest electric field strength. At $t = 2.5$ ns, they move approximately 0.28 and 0.36 cm. Hence, the average velocities of negative and positive streamer from 2 to 2.5 ns are estimated approximately 2.0×10^8 and 1.95×10^8 cm/s, in other words, no obvious difference. Although the negative streamer propagates faster than the positive streamer at the beginning of the propagation, the difference in velocity becomes much smaller after 1.5 ns. The

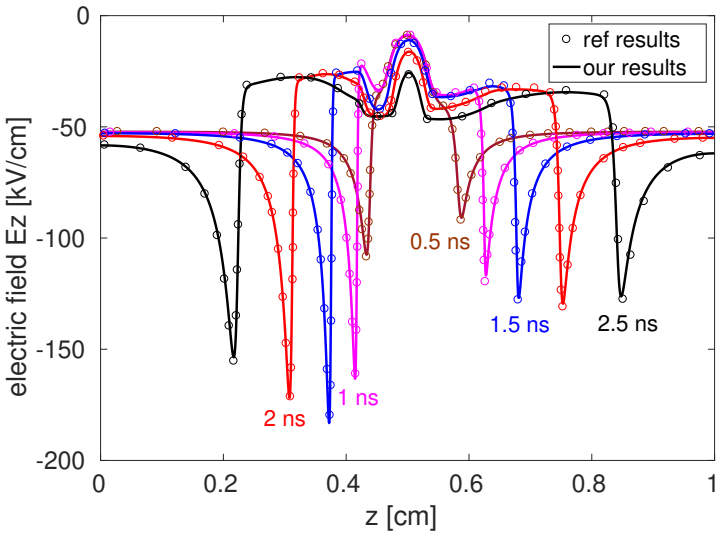


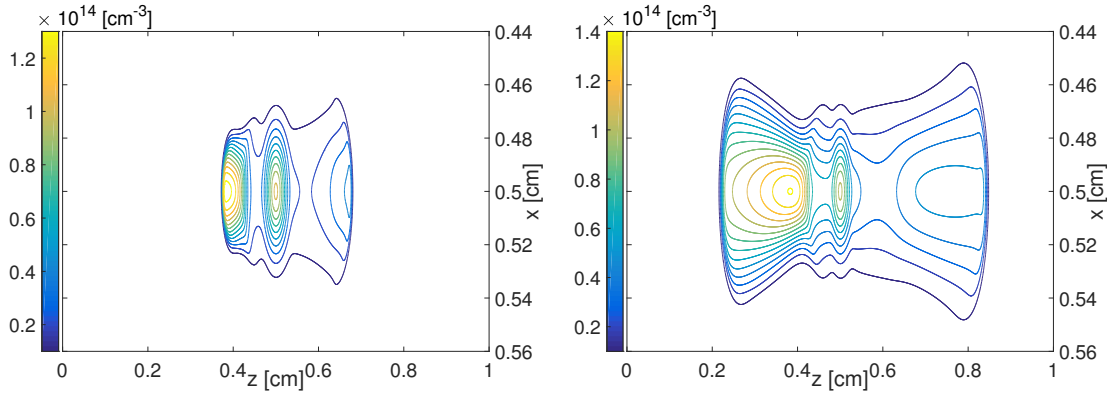
Figure 2.6: Electric field E_z along the line $x = 0.5$, $y = 0.5$ in a double-headed streamer (reference data taken from [14]).

experimentally measured velocities of streamers vary under different experimental conditions by an order of magnitude [178], but the typical measured velocities are similar to those obtained in this simulation.

The contours of the electron density and net charge densities on the plane $y = 0.5$ are shown in Figure 2.7 and Figure 2.8, respectively. The results in Figure 2.7 are very similar to the results obtained in [14]. Figure 2.8 clearly shows that at the front of the streamers' head, a thin layer of net charge that has the same polarity as the streamer exists. The thickness of this layer is approximately 0.2 mm to 0.3 mm, and the maximum net charge density is on the order of $1\mu\text{C/cm}^3$, which is approximately between 10^{12} and 10^{13} charged particles per cm^3.

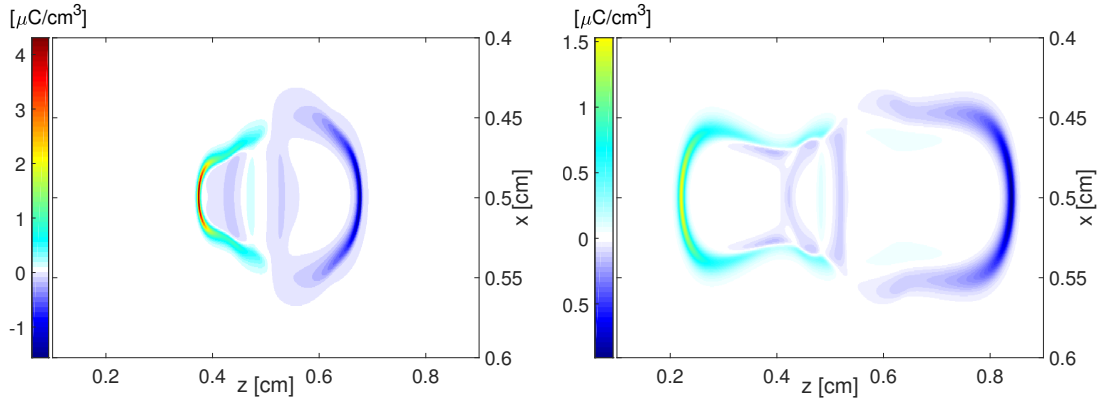
2.7.2 Interactions of two cathode-directed streamers

The second application considers the interactions of two cathode-directed streamers. A 3D simulation enables study of streamer discharges without assuming the axisymmetry. The settings of these simulations are the same as those described in Section 2.7.1 and 2.6.2, except that the initial conditions are set to be the sum of



(a) 1.5 ns (b) 2.5 ns

Figure 2.7: Electron density on the $y = 0.5$ plane in a double-headed streamer at 1.5 and 2.5 ns.



(a) 1.5 ns (b) 2.5 ns

Figure 2.8: Net charge density (in terms of charge per unit volume) on the $y = 0.5$ planes in a double-headed streamer at 1.5 and 2.5 ns. Here net charge density equals to the density of the charge carriers $(n_p - n_e)$ times the elementary charge $(1.602 \times 10^{-13} \mu\text{C})$.

two Gaussians:

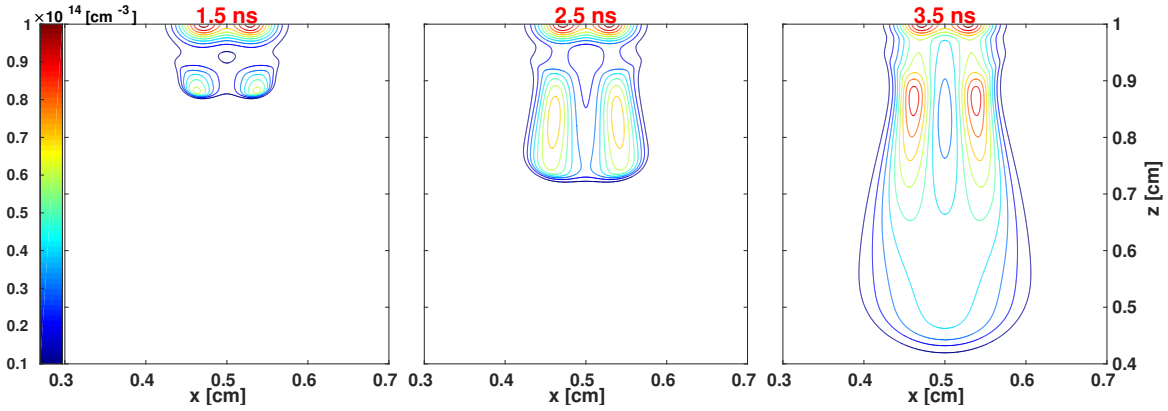
$$n_0(\vec{x}) = 10^{-6} + \exp\left(-\left(\frac{(x-0.5-x_0)^2+(y-0.5)^2+(z-1)^2}{\sigma^2}\right)\right) + \exp\left(-\left(\frac{(x-0.5+x_0)^2+(y-0.5)^2+(z-1)^2}{\sigma^2}\right)\right), \tag{2.38}$$

where $2x_0$ describes the distance between the centers of the two Gaussian-shaped seed charges, and $\sigma = 0.03$. We set x_0 to σ, 2σ and 3σ, to study the effects of different distances between the two streamers on the interactions.

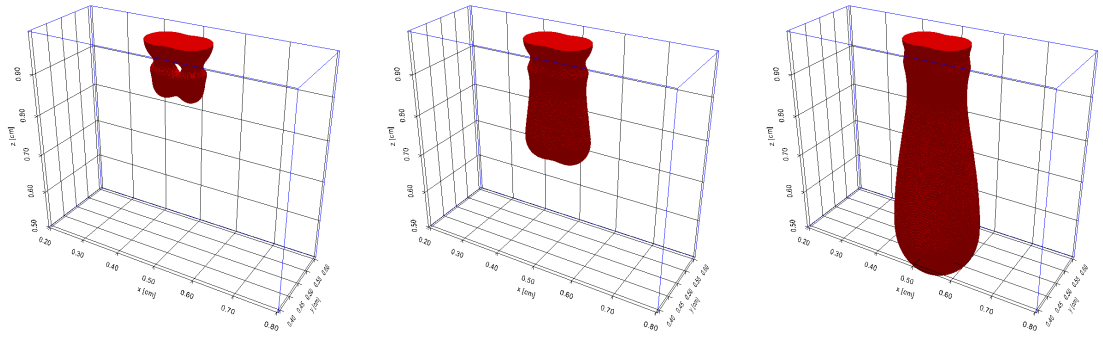
Figures 2.9–2.11 show the evolution of the electric field, net charge and electron density for different x_0 in (2.38) when time $t = 1.5$, 2.5 and 3.5 ns. When the two Gaussians are close (e.g., $x_0 = \sigma$), the two streamers clearly merge into one. When the distance between the two Gaussians is slightly larger (e.g., $x_0 = 2\sigma$), the strong repulsion resulting from the net charge layer at the fronts of the streamers causes the streamers to no longer propagate in the direction of the applied field; however, the distribution of positive ions has already merged by $t = 3.5$ ns, and the two streamers become closer as they propagate.

This may be partly explained as follows. The electric field in the centre of the two streamers (i.e., at $x = 0.5$) is along the direction of the applied field and perpendicular to the electrodes, while the electric fields on the left and right sides of this line have opposite directions, both pointing away from the streamers, which drives the electrons toward the streamers and leaves behind the positive ions in this area. However, this positive net charge greatly enhances the local electric field and attracts the seed electrons (which may be generated by nonlocal photoionization [104]) ahead of it to this area. Hence, the electron density gradually increases due to the collision ionization, which causes the possible merging of the two streamers.

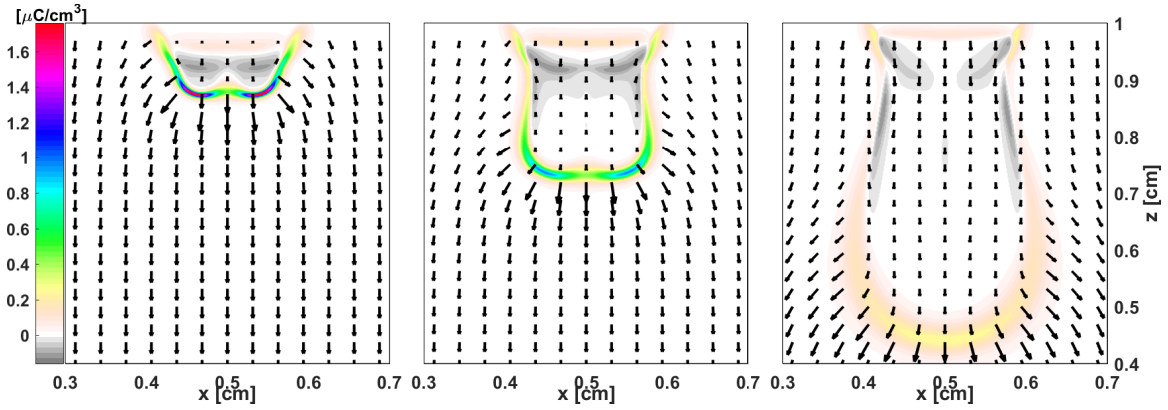
To more focus on the simulator, we here have used a background photoionization rate for simplicity [46]. However, the basic observations are consistent with those in [104] which indicates that two adjacent streamers can interact through electrostatic repulsion and through attraction due to nonlocal photoionization. We will study this using a more detailed photoionization model in next chapter.



(a) contours of electron density on $y = 0.5$ plane

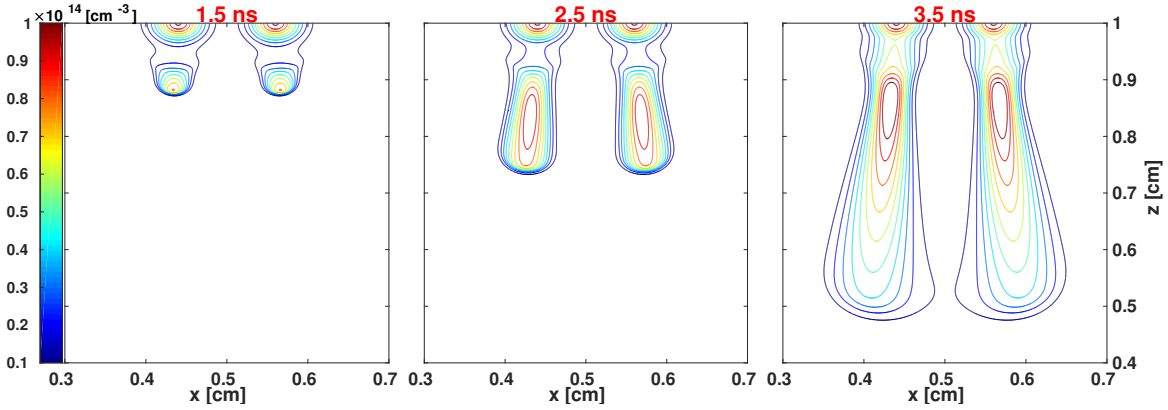


(b) 3D plots of electron density greater than 10^{13} cm^{-3}

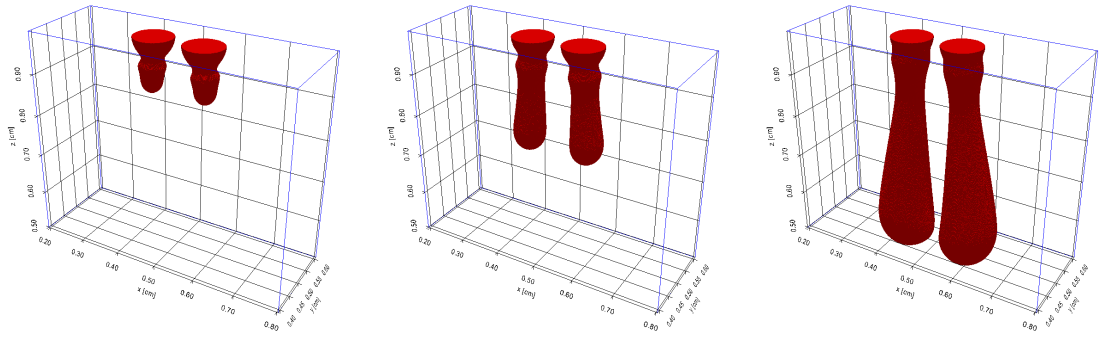


(c) net charge density and electric field (E_x, E_z) on $y = 0.5$

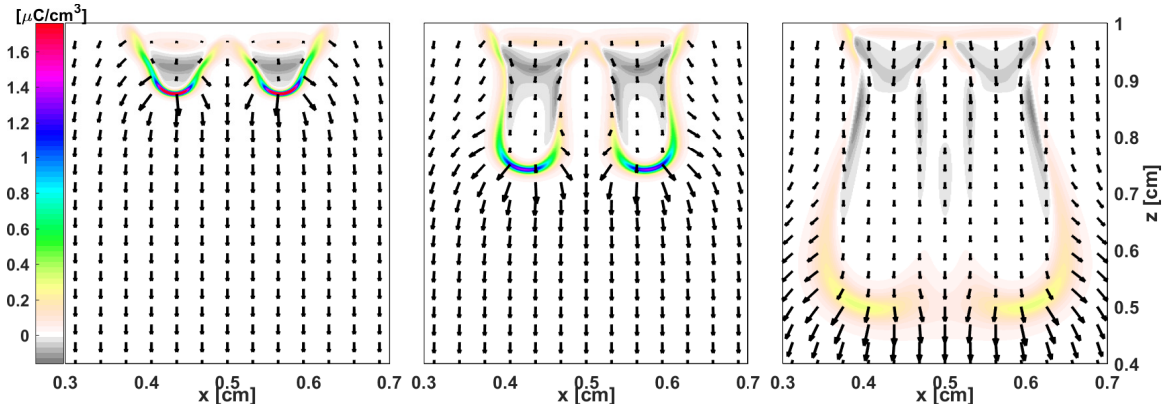
Figure 2.9: Electron density, net charge density and electric field (E_x, E_z) for the interaction between two streamers for $x_0 = \sigma$.



(a) contours of electron density on $y = 0.5$ plane

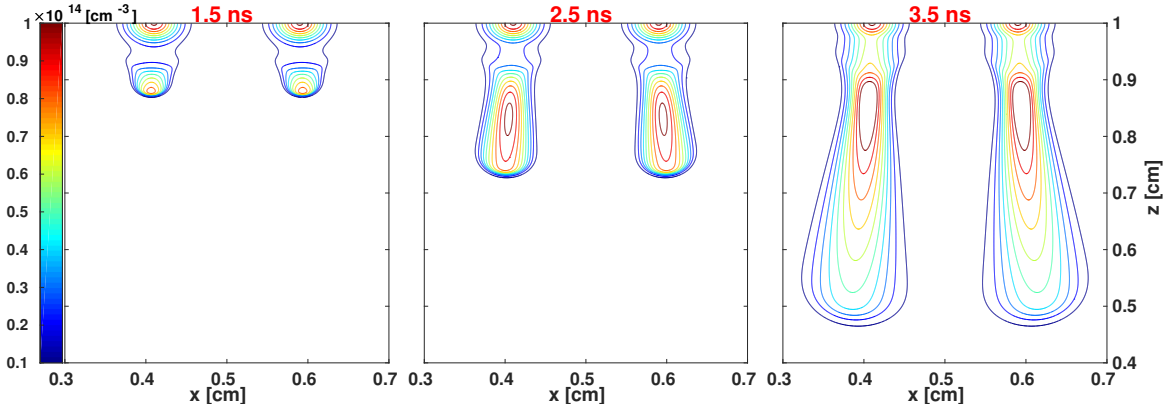


(b) 3D plots of electron density greater than 10^{13} cm^{-3}

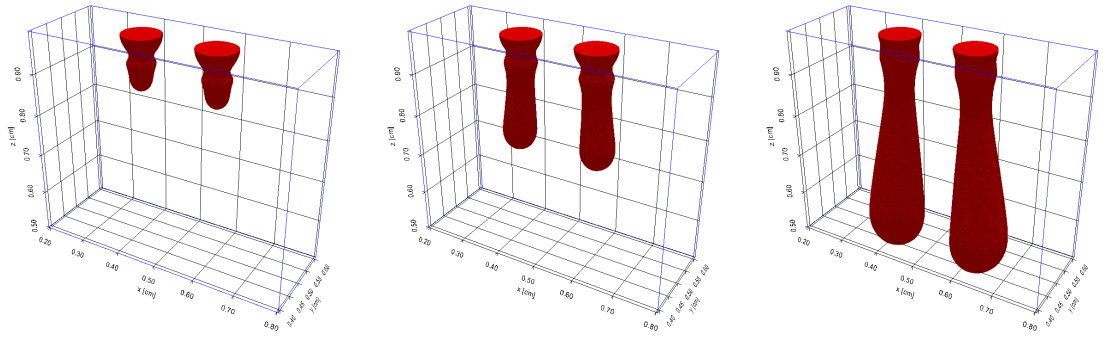


(c) net charge density and electric field (E_x, E_z) on $y = 0.5$

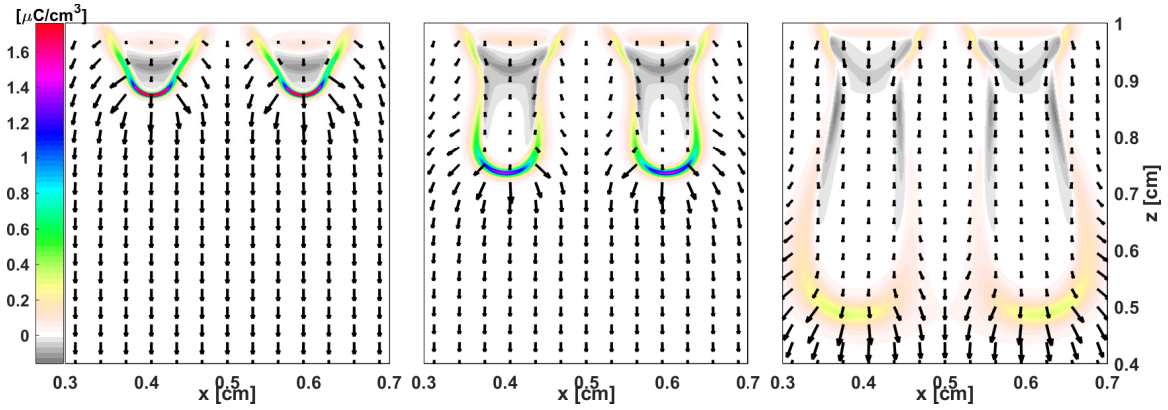
Figure 2.10: Electron density, net charge density and electric field (E_x, E_z) for the interaction between two streamers for $x_0 = 2\sigma$.



(a) contours of electron density on $y = 0.5$ plane



(b) 3D plots of electron density greater than 10^{13} cm^{-3}



(c) net charge density and electric field (E_x, E_z) on $y = 0.5$

Figure 2.11: Electron density, net charge density and electric field (E_x, E_z) for the interaction between two streamers for $x_0 = 3\sigma$.

Chapter 3

Accurate and Efficient Calculation of Photoionization

Photoionization plays an important role in streamer discharge, especially for positive streamer. In this chapter, we focus on the accurate and efficient calculation of photoionization based on the classical integral model derived by Zheleznyak *et al.* [179]. Firstly, the integral model and previous approximations on it are briefly reviewed. Secondly and most importantly, the fast multipole method is described for the evaluation of the integral. Thirdly, numerical testing reports the performance of this evaluation in comparison with other approximations. Finally, we shows numerical applications for interacting streamers with the inclusion of photoionization.

3.1 Existing approaches

To make the contents self-contained, we briefly review commonly used approaches for photoionization calculations. Due to the occurrence of a large amount of physical parameters, we would like to use the fluid model without nondimenionalization (1.3) in this chapter.

3.1.1 Classical integral photoionization

The widely used photoionization model derived by Zheleznyak *et al.* [179] describes the photoionization rate S_{ph} in (1.3) by

$$S_{\text{ph}}(\vec{x}) = \iiint_{V'} \frac{I(\vec{y})g(|\vec{x}-\vec{y}|)}{4\pi|\vec{x}-\vec{y}|^2}\mathrm{d}\vec{y}, \qquad \forall \vec{x} \in V, \tag{3.1}$$

where V' is the source chamber in which the photons are emitted, and V is the collector chamber where the photons are absorbed, $I(\vec{y})$ is proportional to the intensity of the source radiation:

$$I(\vec{y}) = \xi \frac{p_q}{p+p_q}\frac{\omega}{\bar{\alpha}} S_i(\vec{y}), \tag{3.2}$$

where ξ is the photoionization efficiency, p_q is the quenching pressure, p is the gas pressure, ω and $\bar{\alpha}$ are the excitation coefficient of emitting states without quenching processes and the effective Townsend ionization coefficient, respectively, with $\frac{\omega}{\bar{\alpha}}$ being a coefficient to be determined by experiments, and S_i is the effective ionization rate given in (1.4). The function $g(r) = g(|\vec{x}-\vec{y}|)$ in (3.1) is given by

$$\frac{g(r)}{p_{O_2}} = \frac{\exp(-\chi_{\min} p_{O_2} r) - \exp(-\chi_{\max} p_{O_2} r)}{p_{O_2} r \ln(\chi_{\max}/\chi_{\min})}, \tag{3.3}$$

where $r = |\vec{x}-\vec{y}|$, p_{O_2} is the partial pressure of oxygen, $\chi_{\max} = 2\,\text{cm}^{-1}\,\text{Torr}^{-1}$ and $\chi_{\min} = 0.035\,\text{cm}^{-1}\,\text{Torr}^{-1}$ are the maximum and minimum absorption coefficients of O_2 in wavelength 980-1025 Å, respectively, as indicated in [179]. Note that when we define $g(r)$ in (3.3), we follow [18, 130, 179] to write $g(r)/p_{O_2}$ on the left-hand side so that the right-hand side is dependent directly on the product $p_{O_2} r$. Interested readers may refer to [179] and [130] for more details.

Clearly, (3.1) is a convolution in three dimensions. A naive numerical implementation of (3.1) requires a whole domain quadrature for every point $\vec{x} \in V$, which requires a time complexity $O(N^2)$ with N being the total number of degrees of freedom. One idea to reduce the computational cost is to use a coarse grid in the weak field at the price of possibly losing some accuracy, as [80] did in the three dimensional cases with cylindrical symmetry.

3.1.2 Helmholtz PDE approximation

Instead of a straightforward computation of the integral, the efficiency can be significantly enhanced by converting it into a problem of differential equations at the expense of losing some accuracy. One important and pioneer work was done in [105], which approximates the photoionization kernel as the sum of the fundamental solutions of a number of partial differential equations. The function $g(r)$ defined by (3.3) is approximated as follows:

$$\frac{g(r)}{p_{O_2}} \approx p_{O_2} r \sum_{j=1}^{N_E} C_j \exp(-\lambda_j p_{O_2} r), \tag{3.4}$$

where λ_j and C_j $(1 \leq j \leq N_E)$ are constants that can be fit numerically [18, 105]. Consequently, it suffices to take the linear combination of $S_{\mathrm{ph},j}$ to approximate the integral (3.1)

$$S_{\mathrm{ph}}(\vec{x}) \approx \sum_{j=1}^{N_E} C_j S_{\mathrm{ph},j}(\vec{x}), \tag{3.5}$$

where $S_{\mathrm{ph},j}(\vec{x})$ is the solution of the following modified Helmholtz equation

$$(-\Delta + (\lambda_j p_{O_2})^2) S_{\mathrm{ph},j}(\vec{x}) = (p_{O_2})^2 I(\vec{x}). \tag{3.6}$$

This conversion makes use of the Green's function $G(r) = \frac{\exp(-\lambda_j p_{O_2} r)}{4\pi r}$ of the modified Helmholtz equation (3.6) with proper boundary condition. The resulting elliptic equation (3.6) can be solved efficiently by numerous fast elliptic solvers like multigrid-preconditioned FGMRES method mentioned in Chapter 2.4.

$N_E = 2$ was used in [105], and the constants λ_j and C_j were chosen to fit the low-pressure experimental data from [135] (the misprint of these constants is corrected in [49]). $N_E = 3$ was suggested in [18] for a better fitting for the range $1 < p_{O_2} r < 150\,\mathrm{Torr{\cdot}cm}$, and the constants λ_j and C_j are chosen to fit the function in Eq. (3.4) since it agrees well with both the low-pressure experimental data from [135] and the experimental data in atmospheric air from [3], as indicated in [125]. While zero boundary conditions were used in [105], it is suggested in [18] that the boundary

Table 3.1: Coefficients of three-exponential ($N_E = 3$) approximation in (3.4) [18].

j	C_j ($\mathrm{cm}^{-2}\,\mathrm{Torr}^{-2}$)	λ_j ($\mathrm{cm}^{-1}\,\mathrm{Torr}^{-1}$)
1	1.986×10^{-4}	0.0553
2	0.0051	0.1460
3	0.4886	0.89

condition for (3.6) can be provided by computing the integral (3.1). In this thesis, we take the three-term exponential approximation and adopt the coefficients in [18] listed in Table 3.1.

3.1.3 Three-group radiative transfer approximation

Another type of differential equations that can facilitate the computation of the photoionization rate is the radiative transfer equation. In [18, 23, 144], the following multi-group approximation of the steady-state radiative transfer equation is chosen to describe the intensity of radiation Ψ_j for the j-th group of spectral frequency:

$$\vec{\omega} \cdot \nabla \Psi_j(\vec{x}, \vec{\omega}) + \kappa_j \Psi_j(\vec{x}, \vec{\omega}) = \frac{n_u(\vec{x})}{4\pi c\, \tau_u}, \quad j = 0, 1, \cdots, N_\nu, \tag{3.7}$$

where $\vec{\omega} \in S^2$ is the solid angle defined on the unit sphere, κ_j is the absorption coefficient, n_u is the density of the species with the excited state u, c is the speed of light and τ_u is the radiative relaxation time for the state u. Here the scattering and the change in frequency of the photons during collisions with molecules have been neglected [23, 144]. For photoionization in air, $\kappa_j = \lambda_j\, p_{O_2}$, and for simplicity, only one excited state is considered

$$\frac{n_u(\vec{x})}{\tau_u} = \frac{I(\vec{x})}{\xi}, \tag{3.8}$$

Table 3.2: Coefficients of three-group ($N_\nu = 3$) approximation in (3.10) [18].

j	A_j ($\mathrm{cm^{-1}\,Torr^{-1}}$)	λ_j ($\mathrm{cm^{-1}\,Torr^{-1}}$)
1	0.0067	0.0447
2	0.0346	0.1121
3	0.3059	0.5994

with λ_j to be determined by data fitting [18, 144, 179]. The photoionization rate is then proportional to the weighted sum of the integral of Ψ_j over $\vec{\omega} \in S^2$:

$$\begin{aligned} S_{\mathrm{ph}}(\vec{x}) &= \sum_{j=1}^{N_\nu} A_j \, \xi \, p_{O_2} c \int_{S^2} \Psi_j(\vec{x}, \vec{\omega}) \mathrm{d}\vec{\omega}, \\ &= \sum_{j=1}^{N_\nu} A_j \, \xi \, p_{O_2} c \iiint_V \frac{n_u(\vec{y})}{c\,\tau_u} \frac{\exp(-\lambda_j p_{O_2} |\vec{x} - \vec{y}|)}{4\pi |\vec{x} - \vec{y}|^2} \mathrm{d}\vec{y}, \end{aligned} \tag{3.9}$$

where A_j are also parameters which can be fit according to the experimental data, and we have applied the analytical solution of the radiative transfer equation to express the right-hand side. To determine the parameters, it is noticed that (3.9) is identical to (3.1) if

$$\sum_{j=1}^{N_\nu} A_j p_{O_2} \exp(-\lambda_j p_{O_2} r) = g(r), \qquad r = |\vec{x} - \vec{y}|, \tag{3.10}$$

where $g(r)$ is given in (3.3), and the coefficient A_j and λ_j ($1 \le j \le N_\nu$) are determined by fitting the left hand side of (3.10) with $g(r)$ in the range $0.1 < p_{O_2} r < 150\,\mathrm{Torr{\cdot}cm}$ [18]. The results for three-group ($N_\nu = 3$) approximation are shown in Table 3.2.

Instead of computing the integral in (3.9), a more efficient way to get the intensity function Ψ_j is to solve (3.7) as an differential equation. For example, in [23], a direct solver of (3.7) is employed for two-dimensional axisymmetric discharges using the finite volume method for both space and angular variables.

However, the radiative transfer equation (3.7) is still a five-dimensional partial differential equation. Further reduction of dimensionality can be realized by the

improved Eddington or SP_3 approximation [18, 144, 182]. In [83], the simplified P_N (SP_N) approximations of optically thick radiative heat transfer equations are theoretically derived by asymptotic analysis. SP_N approximations are introduced in [144] to obtain a fast numerical simulation for the photoionization source term mainly with monochromatic (one-group) approximation. The SP_N approximations for photoionization are further improved in [18], and extended to multi-group approximation, including the three-group SP_3 method which approximates the isotropic part of the solution by [18, 144]

$$\int_{S^2} \Psi_j(\vec{x}, \vec{\omega}) \mathrm{d}\vec{\omega} = \frac{\gamma_2 \phi_{j,1}(\vec{x}) - \gamma_1 \phi_{j,2}(\vec{x})}{\gamma_2 - \gamma_1}, \tag{3.11}$$

where $\gamma_n = \frac{5}{7}\left[1 + (-1)^n 3\sqrt{\frac{6}{5}}\right]$ with $n = 1, 2$, and $\phi_{j,1}(\vec{x})$ and $\phi_{j,2}(\vec{x})$ are solutions of the following two Helmholtz equations

$$\left(-\Delta + \frac{(\lambda_j p_{O_2})^2}{\mu_1^2}\right) \phi_{j,1}(\vec{x}) = \frac{\lambda_j p_{O_2}}{\mu_1^2} \frac{n_u(\vec{x})}{c\,\tau_u}, \tag{3.12}$$

$$\left(-\Delta + \frac{(\lambda_j p_{O_2})^2}{\mu_2^2}\right) \phi_{j,2}(\vec{x}) = \frac{\lambda_j p_{O_2}}{\mu_2^2} \frac{n_u(\vec{x})}{c\,\tau_u}, \tag{3.13}$$

with the coefficients $\mu_n = \sqrt{\frac{3}{7} + (-1)^n \frac{2}{7}\sqrt{\frac{6}{5}}}$ $(n = 1, 2)$. The equations (3.12)-(3.13) need to be equipped with proper boundary conditions (BCs). In [18], the BCs for (3.12)-(3.13) are obtained directly from the integral model (3.1), which requires numerical integrations over the whole domain for all the grid points on the boundary. Later in [98], the same authors proposed the following more efficient BCs based on [83] for a boundary surface without reflection and emission:

$$\nabla \phi_{j,1} \cdot \vec{n} + \alpha_1(\lambda_j p_{O_2}) \phi_{j,1} = -\beta_2(\lambda_j p_{O_2}) \phi_{j,2}, \tag{3.14}$$

$$\nabla \phi_{j,2} \cdot \vec{n} + \alpha_2(\lambda_j p_{O_2}) \phi_{j,2} = -\beta_1(\lambda_j p_{O_2}) \phi_{j,1}, \tag{3.15}$$

where $\vec{n}$ is the outward unit normal vector, $\alpha_n = \frac{5}{96}\left(34 + (-1)^{n-1} 11\sqrt{\frac{6}{5}}\right)$ and $\beta_n = \frac{5}{96}\left(2 + (-1)^n \sqrt{\frac{6}{5}}\right)$ $(n = 1, 2)$. A comparison of these two BCs can be found in [24, Chapter III.6].

3.2 Fast multipole method for the evaluation of integral

As can be seen from the Sections 3.1.2 and 3.1.3, different methods based on differential equations have been proposed to approximate the integral (3.1) or (3.9), leading to much higher numerical efficiency. However, these methods may suffer numerical issues, i.e. the approximation errors might be significant in some cases. On the other hand, despite the high computational cost [49], the results calculated from the integral form are free of further approximations, therefore, these results are often used as reference solutions [18, 24, 144]. Moreover, the integral form can be easily extended to stochastic versions [25, 153]. The importance of the integral form inspires us to tackle the original integration problem (3.1) directly using fast algorithms. The exponential decay of the kernel with respect to the distance (see (3.3)) reminds us to adopt the fast and accurate fast multipole method [172, 173], which utilizes the low-rank structure of far-away interactions to gain significant speed-up. As a result, the description in this section is mainly based on [172].

3.2.1 General idea

The fast multipole method used in this thesis [96] is established based on the fast evaluation of the numerical quadrature of (3.1). For convenience, we discretize S_{ph} and n_e on the same mesh. In general, the integral (3.1) can be discretized as

$$S_{\mathrm{ph}}(\vec{x}_i) = \sum_{j=1}^{N_{\mathrm{pt}}} G(\vec{x}_i, \vec{y}_j) I(\vec{y}_j), \qquad i = 1, \cdots, N_{\mathrm{pt}}, \tag{3.16}$$

where $G(\cdot, \cdot)$ is the discrete kernel function calculated from the corresponding function in (3.1) and the numerical quadrature weights. In this thesis, we apply the midpoint quadrature rule on each grid cell unless $\vec{x}_i$ and $\vec{y}_j$ are in the same grid where the second-order Gauss-Legendre quadrature rule is alternatively applied. We remark that this is not essential and other numerical quadratures can also be used.

The kernel-independent adaptive fast multipole method [172] does not require the implementation of multipole expansions [30, 59] of the kernel function. Based on a hierarchical tree, it uses a continuous equivalent density on a surface enclosing a box to represent the potential generated by sources inside the box. Given a set of N_{pt} points in three dimensions, a hierarchical octree is constructed adaptively such that each leaf cube of the tree contains no more than m points, where m is a selected constant. This octree can be built from a sufficiently large root cube to contain all N_{pt} points, and then subdivided to equal-sized sub-cubes recursively if the current cube contains more than m points. For illustrative purpose, an example of the hierarchical tree in two dimensions (2D), i.e. quadtree, is shown in Figure 3.1.

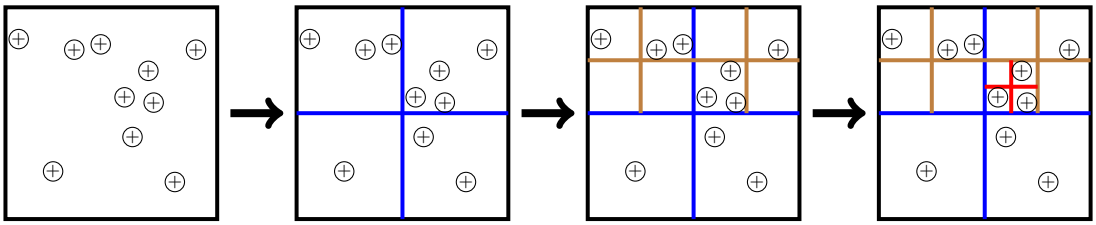


Figure 3.1: An example of a hierarchical tree in 2D, with $N_{\text{pt}} = 10$, $m = 2$. The arrows show the construction procedure, and the circles with "+" denote the N_{pt} points.

To sketch the idea, we consider the simple case where the source points are uniformly distributed. This corresponds to the case when the uniform mesh is applied in the discretization of $I(\vec{y})$ in (3.2). In this case, for each target point $\vec{x}_i$ in a cube or box B, fast multipole method splits the summation (3.16) into two parts, namely, near interactions and far interactions:

$$S_{\text{ph}}(\vec{x}_i) = \sum_{\vec{y}_j \in \mathcal{N}(B)} G(\vec{x}_i, \vec{y}_j) I(\vec{y}_j) + \sum_{\vec{y}_j \in \mathcal{F}(B)} G(\vec{x}_i, \vec{y}_j) I(\vec{y}_j), \tag{3.17}$$

where $\mathcal{N}(B)$ and $\mathcal{F}(B)$ are the near range and far range of B, respectively. For $\vec{y}_j \in \mathcal{N}(B)$, the interactions with all $\vec{x}_i \in B$ are calculated directly. For the points $\vec{y}_j \in \mathcal{F}(B)$, the interactions can be approximated with controlled accuracy due to

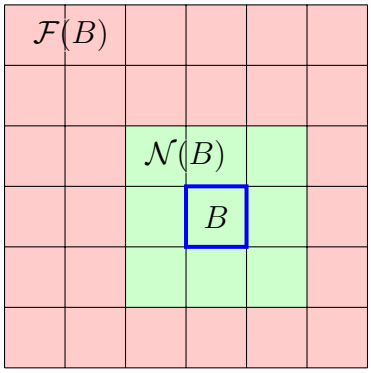


Figure 3.2: Cross section of near range $\mathcal{N}(B)$ and far range $\mathcal{F}(B)$ of a box B in 3D. The blue thick side is the boundary of B, green part is $\mathcal{N}(B)$ and red part is $\mathcal{F}(B)$.

the low-rankness of $G(\vec{x}_i, \vec{y}_j)$. If a box is centered at $\vec{c}$ with side length $2r$, then $\mathcal{N}(B)$ is defined as a box centered at $\vec{c}$ with side length $6r$, and $\mathcal{F}(B)$ is the domain outside $\mathcal{N}(B)$ (See Figure 3.2).

In (3.17), the summation for points $\vec{y}_j \in \mathcal{F}(B)$ can be approximated using the hierarchy tree. The idea is composed of two parts: 1) represent the potential generated from source points inside any box B by some equivalent source points enclosing B; 2) represent the potential generated from source points in $\mathcal{F}(B)$ by other equivalent source points enclosing B, which gives an approximation to the summation for points $\vec{y}_j \in \mathcal{F}(B)$ in (3.17). The first part is implemented by post-order traversal of the hierarchical tree. If B is a leaf box, the potential generated from the source points inside B is represented by several equivalent points surrounding the box, as is called the multipole expansion to be defined in (3.18). If B is not a leaf box, its multipole expansion can be accumulated from the multipole expansion of all its children boxes by "M2M translation" to be defined in (3.20). With the help of the equivalent source points in the first part, we can approximate the potential in B from original source points in $\mathcal{F}(B)$ by a small number of equivalent source points in $\mathcal{F}(B)$ calculated from the first part. This is the idea of the second part, and we similarly represent the potential generated from source points in $\mathcal{F}(B)$ by some equivalent source points surrounding B, as is called the local expansion to be defined

in (3.19). The second part is implemented by pre-order traversal of the hierarchical tree. If a non-root box B is embedded in its parent box $\mathcal{P}(B)$, its local expansion is calculated from: the accumulation of the local expansion of $\mathcal{P}(B)$, which is called "L2L translation" to be defined in (3.22); and the multipole expansion of the boxes in $\mathcal{N}(\mathcal{P}(B))$ but not adjacent to B, as it is implemented by the operation called "M2L translation" to be defined (3.21).

Next we will show more details about the kernel-independent FMM in the next two subsections: firstly introduce multipole expansion and local expansion in the FMM, and then show three translations among them: M2M (multipole to multipole), M2L (multipole to local) and L2L (local to local). For simplicity, we would like to neglect the vector symbol on $\vec{x}$ and $\vec{y}$ when introducing FMM.

3.2.2 Multipole and local expansions

In this subsection, multipole expansion and local expansion are introduced.

Multipole expansion Multipole expansion of a box B is used to represent the potential in $\mathcal{F}(B)$, generated by the source inside B. Two surfaces of the cube are introduced for the approximation, upward equivalent surface $y^{B,u}$ and upward check surface $x^{B,u}$. The equivalent surface $y^{B,u}$ should be taken to enclose B, and check surface $x^{B,u}$ encloses equivalent surface $y^{B,u}$. Moreover, both $y^{B,u}$ and $x^{B,u}$ should locate inside $\mathcal{N}(B)$. See these two box surfaces in Figure 3.3.

An upward density function $\phi^{B,u}(y)$, or the density $\phi_k^{B,u} = \phi^{B,u}(y_k^{B,u})$ on several upward equivalent source points $y_k^{B,u} \in y^{B,u}$, is introduced to represent the potential in $\mathcal{F}(B)$, generated by the source inside B. If the upward check potential $q^{B,u}(x)$ at the check surface $x^{B,u}$, evaluated from the source in B, is equal to the potential $q^{B,u}(x)$ evaluated from the equivalent source $\phi_k^{B,u}$, then these source density points $\phi_k^{B,u}$ can be used to represent the potential outside the check surface $x^{B,u}$ including $\mathcal{F}(B)$. This is because of the uniqueness of the Dirichlet boundary value problem (similar to the method of image charges in electrostatics). The equality is written

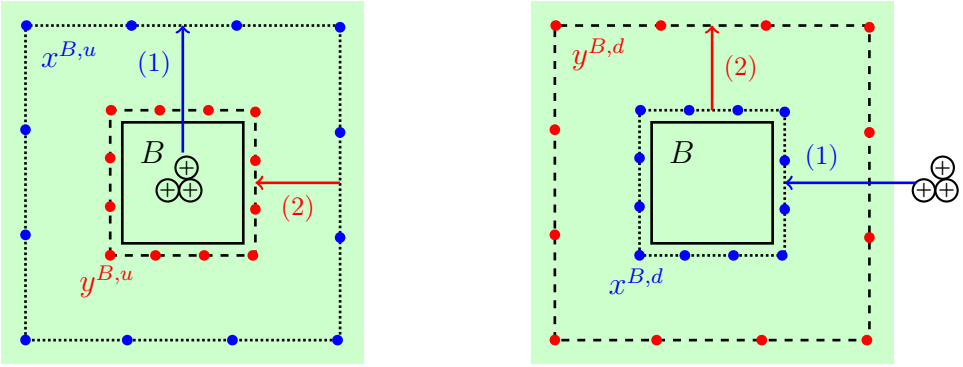


Figure 3.3: Cross section of equivalent surfaces and check surfaces in multipole expansion (left subfigure) and local expansion (right subfigure) of box B. Dashed lines with red dots denote equivalent surfaces, where red dots can be viewed as equivalent sources. Dotted lines with blue dots denote check surface, where blue dots can be viewed as check points. Green shadow is the near range of B. Circles with "+" denote source points. Step (1) in blue arrow is the evaluation of potential on check surface, and step (2) in red arrow is the calculation of equivalent density on equivalent surface.

as

$$\sum_{k\in Id(y^{B,u})} G(x_j^{B,u}, y_k^{B,u})\phi_k^{B,u} = q^{B,u}(x_j^{B,u}) = \sum_{i\in I_s^B} G(x_j^{B,u}, y_i)I(y_i),\ \forall j \in Id(x^{B,u}), \tag{3.18}$$

where I_s^B is the index set of the source points inside B, $Id(y^{B,u})$ is the index set of discrete source points on $y^{B,u}$ and $Id(x^{B,u})$ is the index set of discrete check points on $x^{B,u}$. A prescribed number m_0 is used to denote the number of discrete equivalent source points at each side of $y^{B,u}$, and this number is identical to the number of check points at each side of $x^{B,u}$.

Local expansion Local expansion is used to represent the potential inside a box B, generated by the source in $\mathcal{F}(B)$. Similar to the multipole expansion, a downward equivalent surface $y^{B,d}$ with downward equivalent density $\phi^{B,d}$ on it, is introduced. At the same time, downward check surface $x^{B,d}$ with downward check potential $q^{B,d}$

is used to check the equality of potential generated by the source in $\mathcal{F}(B)$ and the one generated by $\phi^{B,d}$. Different from the multipole expansion, $y^{B,d}$ should enclose $x^{B,d}$, since in the local expansion we want to approximate the potential inside B. Again both $y^{B,d}$ and $x^{B,d}$ should locate between B and $\mathcal{F}(B)$. Two surfaces are shown in Figure 3.3 as an example, with evaluation procedure.

The downward equivalent density satisfies:

$$\sum_{k\in Id(y^{B,d})} G(x_j^{B,d}, y_k^{B,d})\phi_k^{B,d} = q^{B,d}(x_j^{B,d}) = \sum_{i\in I_s^{\mathcal{F}(B)}} G(x_j^{B,d}, y_i)I(y_i),\ \forall j \in Id(x^{B,d}), \tag{3.19}$$

where $I_s^{\mathcal{F}(B)}$ is the index set of the source points in $\mathcal{F}(B)$, $Id(y^{B,d})$ is the index set of source points on $y^{B,d}$ and $Id(x^{B,d})$ is the index set of check points on $x^{B,d}$. Again a prescribed finite number (related to m_0) of index is chosen in $Id(\cdot)$.

3.2.3 Translations among different expansions

After introducing the multipole and local expansions in the previous subsection, we shows three translations among them in this subsection, which are M2M, M2L and L2L. Then the algorithm of FMM is summarized.

M2M translation M2M translation translates the upward equivalent density $\phi^{B,u}$ of a box, to the upward equivalent density $\phi^{\mathcal{P}(B),u}$ of its parent box $\mathcal{P}(B)$. The idea is similar to (3.18), with an upward check surface of $\mathcal{P}(B)$ as $x^{\mathcal{P}(B),u}$, and the corresponding upward check potential $q^{\mathcal{P}(B),u}$. The equality is given as

$$\begin{aligned} q^{\mathcal{P}(B),u}(x_j^{\mathcal{P}(B),u}) &= \sum_{k\in Id(y^{\mathcal{P}(B),u})} G(x_j^{\mathcal{P}(B),u}, y_k^{\mathcal{P}(B),u})\phi_k^{\mathcal{P}(B),u} \\ &= \sum_{i\in Id(y^{B,u})} G(x_j^{\mathcal{P}(B),u}, y_i^{B,u})\phi_i^{B,u}, \qquad \forall j \in Id(x^{\mathcal{P}(B),u}). \end{aligned} \tag{3.20}$$

In the implementation, we first add the potential from the upward equivalent density of all children boxes to the check surface of the parent box, which is marked as blue arrow in the left-most subfigure of Figure 3.4. After accumulation from all children boxes to $q^{\mathcal{P}(B),u}$, we evaluate the upward equivalent density $\phi^{\mathcal{P}(B),u}$ which

is marked as red arrow in the same subfigure. This implementation, which is adding potential to the check surface and then calculating the equivalent density from check potential, is also applied to the calculation of downward equivalent density. Therefore, we also indicate the implementation by blue and red arrows in other subfigures related to M2L and L2L translations in Figure 3.4.

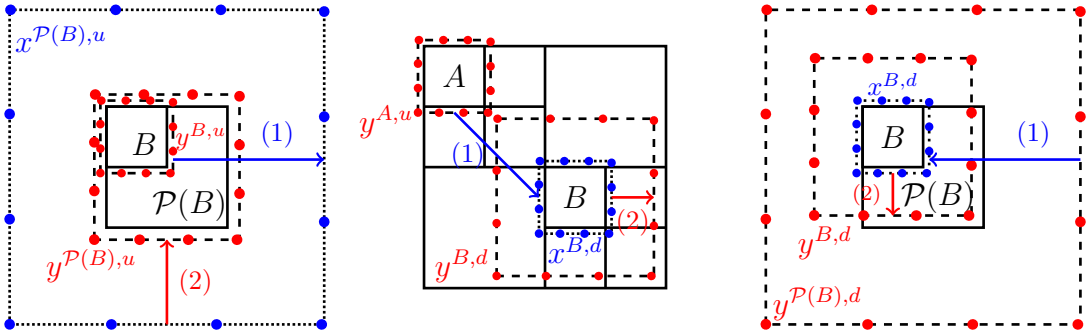


Figure 3.4: Cross section of M2M translation (left subfigure), M2L translation (middle subfigure) and L2L translation (right subfigure). Dashed lines with red dots denote equivalent surfaces. Dotted lines with blue dots denote check surface. Step (1) in blue arrow is the evaluation of potential on check surface, and step (2) in red arrow is the calculation of equivalent density on equivalent surface. $\mathcal{P}(B)$ denotes parent box of B.

M2L translation Two boxes A and B are well-separated if $A \subset \mathcal{F}(B)$ and $B \subset \mathcal{F}(A)$. If two boxes A and B are in same size and well-separated, M2L translation can be used to translate the multipole expansion of A to local expansion of B. In other words, M2L translation calculates the downward equivalent density of B from the upward equivalent density of A, which accumulates the potential in B from the source in A. See this procedure in Figure 3.4, which satisfies

$$\sum_{k\in Id(y^{B,d})} G(x_j^{B,d}, y_k^{B,d})\phi_k^{B,d} = q^{B,d}(x_j^{B,d}) = \sum_{i\in Id(y^{A,u})} G(x_j^{B,d}, y_i^{A,u})\phi_i^{A,u},\ \forall j \in Id(x^{B,d}). \tag{3.21}$$

Since these two boxes A and B are in same size, fast Fourier transform can be used to speed up the calculation in (3.21), as indicated in [172].

L2L translation For a box B, $\mathcal{F}(\mathcal{P}(B)) \subset \mathcal{F}(B)$, therefore L2L translation is used to calculation the local expansion of B from the local expansion of $\mathcal{P}(B)$. This specifies the potential in B from the source in $\mathcal{F}(\mathcal{P}(B))$. Similar as (3.20), the equation is

$$\begin{aligned} q^{B,d}(x_j^{B,d}) &= \sum_{k \in Id(y^{\mathcal{P}(B),d})} G(x_j^{B,d}, y_k^{\mathcal{P}(B),d}) \phi_k^{\mathcal{P}(B),d} \\ &= \sum_{i \in Id(y^{B,d})} G(x_j^{B,d}, y_i^{B,d}) \phi_i^{B,d}, \qquad \forall j \in Id(x^{B,d}). \end{aligned} \tag{3.22}$$

Outline of the algorithm The outline steps of the kernel-independent FMM is presented as follows:

1. Tree construction: to construct a hierarchical tree in pre-order traversal, such that each leaf box contains no more than m source points.

2. Upward pass: to calculate the multipole expansion for leaf boxes, and use M2M translation for multipole expansion of all non-leaf boxes in a post-order traversal of the hierarchical tree.

3. Downward pass: for non-root boxes, use local expansion, M2L and L2L translations to accumulate the potential from far range in a pre-order traversal of the tree.

4. Target potential: for each leaf box in pre-order traversal of the tree, sum up the near interactions with the potential calculated in the last step, get the target potential.

We would like to remark that we tested the backward stable pseudo-inverse trick indicated in [107] for (3.1), and find few difference with results given by the original pseudo-inverse in [172] for our problems. Therefore, we implement the

kernel-independent FMM by the original pseudo-inverse with prescribed number $m_0 = 6$ (number of equivalent source or check points at each side of the enclosing box) in this thesis. We find that $m_0 = 6$ gives a good balance between accuracy and computational cost. Readers may consider increasing m_0 to obtain a more accurate result, or decreasing m_0 for a faster computation.

It should be emphasized that in order to capture the multiscale structure of streamers, a non-uniform mesh may be adopted in the simulation like in [14, 51, 109], or what will be mentioned in Chapter 4. The aforementioned fast multipole framework still works for non-uniform and unstructured meshes. For a more detailed algorithm on non-uniform mesh, we would like to refer to [172].

3.3 Numerical comparisons

In this section, we compare the performance, in terms of accuracy and efficiency, of different methods for the evaluation of the photoionization S_{ph} defined in (3.1) with (3.2). We take $V = V' = \Omega = (0, x_d) \times (0, y_d) \times (0, z_d)$ cm^3 and denote its center as $\vec{x}_0 = (x_0, y_0, z_0)^T = (x_d/2, y_d/2, z_d/2)^T$ cm. The box V is partitioned uniformly by $n_x \times n_y \times n_z$ cells, with n_x, n_y and n_z the number of cells along x, y, z directions, respectively.

For simplicity, different numerical methods to be compared are summarized in Table 3.3. The numerical simulations were performed on the Tianhe2-JK cluster located at Beijing Computational Science Research Center. More details can be found at `https://www.csrc.ac.cn/en/facility/cmpt/2015-05-07/8.html`. In our computations via the MPI parallelism, excepted stated otherwise, we always use 4 nodes with 20 cores in each node in the simulation.

Table 3.3: Notations of several methods introduced in this chapter.

Notation of method	Brief description
Classical Int	Direct calculation on (3.1), with (3.2) and (3.3).
Helmholtz zero BC	Three-term summation on (3.5), by solving (3.6) with zero boundary condition.
Helmholtz Int BC	Three-term summation on (3.5), by solving (3.6) with integral boundary condition from (3.1).
SP_3 Larsen BC	Three-group summation on (3.9), by solving (3.12)–(3.13) with boundary conditions (3.14)–(3.15).
SP_3 Int BC	Three-group summation on (3.9), by solving (3.12)–(3.13) with integral boundary condition (3.1).
FMM classical Int	Fast multipole method based on (3.1), with (3.2) and (3.3).

The accuracy of different numerical methods is quantified by the following relative errors:

$$\begin{aligned} \mathcal{E}_V &:= \frac{\|S_{\rm ph}^{\rm num}(\vec{x}) - S_{\rm ph}^{\rm ref}(\vec{x})\|_2}{\|S_{\rm ph}^{\rm ref}(\vec{x})\|_2} \times 100\%, \\ \mathcal{E}_\delta(\vec{x}_0) &:= \frac{1}{N_{\rm tot}} \sum_{|\vec{x}-\vec{x}_0|\le\delta} \frac{|S_{\rm ph}^{\rm num}(\vec{x}) - S_{\rm ph}^{\rm ref}(\vec{x})|}{S_{\rm ph}^{\rm ref}(\vec{x})} \times 100\%, \end{aligned} \tag{3.23}$$

where $\|\cdot\|_2$ is the standard discrete l^2-norm on V, $\vec{x}_0 \in V$, $\delta > 0$ is a constant to be fixed later, $S_{\rm ph}^{\rm ref}(\vec{x})$ is the reference result calculated by the (discrete) Classical Int method, $S_{\rm ph}^{\rm num}(\vec{x})$ is the numerical approximation by a numerical method, and $N_{\rm tot}$ is the number of grid points located within a δ radius of $\vec{x}_0$. In fact, here $\mathcal{E}_V$ and $\mathcal{E}_\delta(\vec{x}_0)$ can be regarded as the global relative error over the whole domain V and the local relative error over a ball centered at $\vec{x}_0$ with radius δ, respectively.

3.3.1 Gaussian source with different sizes of domain

The first example is to compute the photoionization rate $S_{\rm ph}(\vec{x})$ in (3.1) generated from a single Gaussian emission source, which is taken from [18]. The Gaussian

ionization source $S_i(\vec{x})$ in (3.2) is given as

$$S_i(\vec{x}) = 1.53\times 10^{25} \exp\left(-\left((x-x_0)^2+(y-y_0)^2+(z-z_0)^2\right)/\sigma^2\right) \mathrm{cm}^{-3}\,\mathrm{s}^{-1}, \quad (3.24)$$

where $\sigma > 0$ is a constant to be fixed later. The other physics parameters in (3.1)-(3.3) are chosen as [18, 144]: $p_q = 30\,\mathrm{Torr}$, $p = 760\,\mathrm{Torr}$, $\xi = 0.1$, $\omega/\alpha = 0.6$, $p_{O_2} = 150\,\mathrm{Torr}$. We take $\delta = 5\sigma$ in (3.23).

We take a relatively small grid size $n_x = n_y = 320$ and $n_z = 160$ because direct computation of the classical integral (3.1) is too time-consuming even if parallel computing is utilized.

Similar to [18], we demonstrate the influence of different ranges of $p_{O_2} r$ by considering two different sizes of the domain V:

1. $x_d = y_d = 0.4\,\mathrm{cm}$, $z_d = 0.2\,\mathrm{cm}$, $\sigma = 0.01\,\mathrm{cm}$;
2. $x_d = y_d = 0.04\,\mathrm{cm}$, $z_d = 0.02\,\mathrm{cm}$, $\sigma = 0.001\,\mathrm{cm}$;
3. $x_d = y_d = 4\,\mathrm{cm}$, $z_d = 2\,\mathrm{cm}$, $\sigma = 0.1\,\mathrm{cm}$.

The numerical results are shown in Figures 3.5–3.7, and the relative errors are then shown in Tables 3.4–3.6.

As it is clearly shown in Figure 3.5, Figure 3.6 and Figure 3.7, the FMM classical Int method always gives the most accurate results, especially for the smaller domain. In all the figures, the lines for "FMM classical Int" almost coincide with the lines for "Classical Int". The deviations of the solutions of the other four methods from the reference results are clearly observable, especially in the central area where the peak locates. Near the boundaries, the methods based on modified Helmholtz equations and SP_3 equations are accurate only when the boundary values are computed from direct integration. Tables 3.4–3.6 also show the superiority of the FMM method in terms of accuracy. Its relative error is one or two orders of magnitude less than other methods.

Regarding the efficiency, it should be noted that both Helmholtz Int BC method and SP_3 Int BC method take the boundary values from the Classical Int method, and

Table 3.4: Time usage and relative error of methods indicated in Table 3.3, for the case of single Gaussian source $x_d = y_d = 0.4\,\text{cm}$, $z_d = 0.2\,\text{cm}$, $\sigma = 0.01\,\text{cm}$.

Method	Time usage (s)	$\mathcal{E}_V$	$\mathcal{E}_\delta(\vec{x}_0)$
Classical Int	184248	—	—
FMM classical Int	27.1897	0.21%	1.30%
Helmholtz zero BC	3.66044	25.33%	16.37%
Helmholtz Int BC	3.76133+4606.20[a]	25.32%	15.54%
SP_3 Larsen BC	12.9268	12.05%	8.49%
SP_3 Int BC	7.30268+4606.20[a]	12.05%	8.53%

[a]Time usage to compute the boundary values, which is estimated from Classical Int method, with multiplication to a factor $2(n_x \times n_y + n_x \times n_z + n_y \times n_z)/(n_x \times n_y \times n_z)$.

Table 3.5: Time usage and relative error of methods indicated in Table 3.3, for the case of one Gaussian source $x_d = y_d = 0.04\,\text{cm}$, $z_d = 0.02\,\text{cm}$, $\sigma = 0.001\,\text{cm}$.

Method	Time usage (s)	$\mathcal{E}_V$	$\mathcal{E}_\delta(\vec{x}_0)$
Classical Int	183761	—	—
FMM classical Int	27.6406	0.22%	0.53%
Helmholtz zero BC	3.87327	83.26%	48.03%
Helmholtz Int BC	4.09442+4594.03[a]	82.96%	44.82%
SP_3 Larsen BC	17.3571	68.92%	16.67%
SP_3 Int BC	7.88867+4594.03[a]	68.88%	16.53%

[a]Time usage to compute the boundary values, which is estimated from Classical Int method, with multiplication to a factor $2(n_x \times n_y + n_x \times n_z + n_y \times n_z)/(n_x \times n_y \times n_z)$.

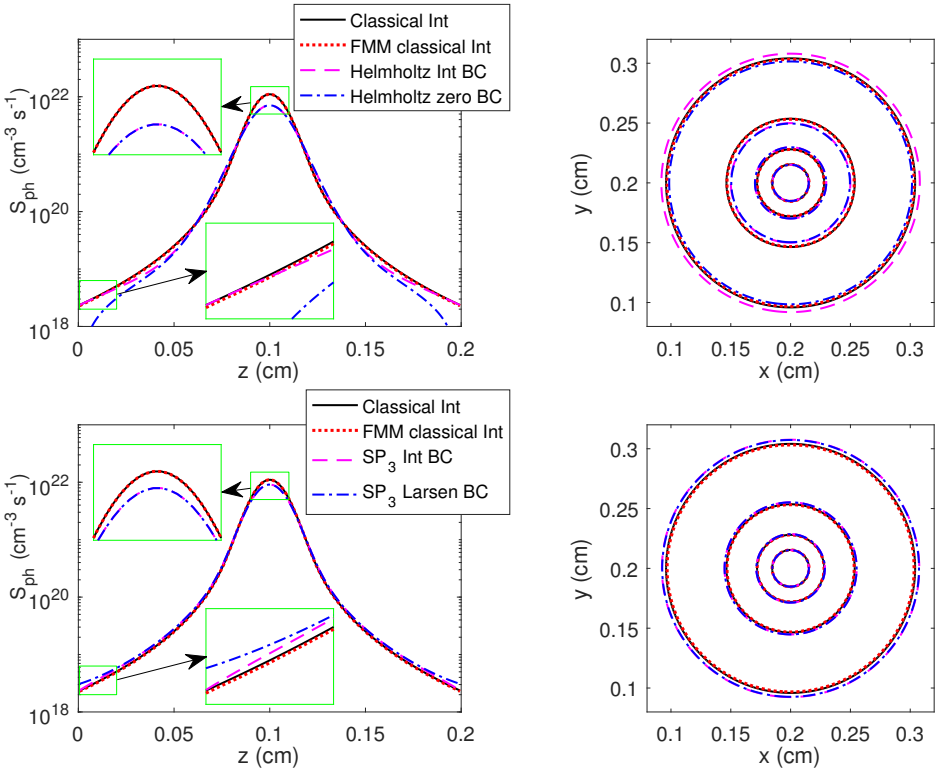


Figure 3.5: Photoionization rate S_{ph} calculated from one Gaussian source in (3.24). $x_d = y_d = 0.4\,\text{cm}$, $z_d = 0.2\,\text{cm}$, $\sigma = 0.01\,\text{cm}$. The figures in the left column are S_{ph} along line $x = y = 0.2\,\text{cm}$, while the figures in the right column are contours of S_{ph} on the plane $z = 0.1\,\text{cm}$, with the values of the contour lines being 2×10^{18}, 2×10^{19}, 2×10^{20}, $2 \times 10^{21}\,\text{cm}^{-3}\,\text{s}^{-1}$. The line color and format in the right-hand side subfigures are same as the one in the left-hand side in same row.

the time to compute the boundary conditions is also included in Tables 3.4–3.6 for a fair comparison. Both tables show that the FMM classical Int method is significantly faster than the Classical Int method. In fact, the integration only for the boundary nodes is already much more expensive than the FMM classical Int method. For the three efficient methods, including FMM classical Int, Helmholtz zero BC, and SP_3 Larsen BC, their computational times have similar magnitudes, and the speed-accuracy trade-off can be observed, meaning that higher computational cost yields better accuracy. Nevertheless, the remarkably lower numerical error and the mildly

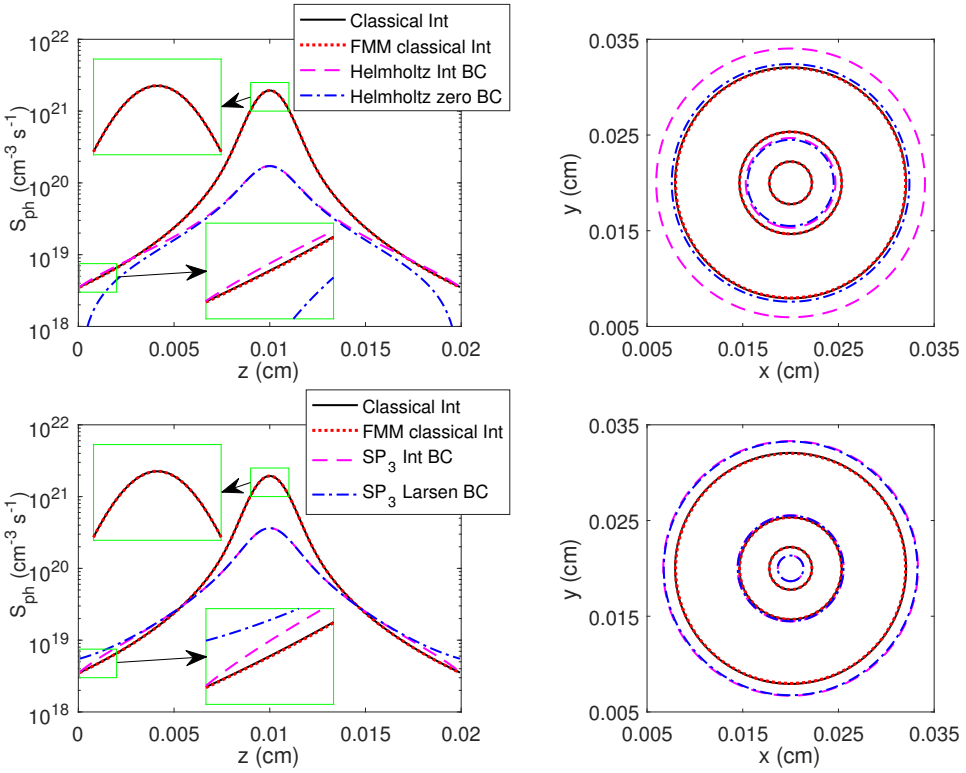


Figure 3.6: Photoionization rate S_{ph} calculated from one Gaussian source in (3.24). $x_d = y_d = 0.04\,\text{cm}$, $z_d = 0.02\,\text{cm}$, $\sigma = 0.001\,\text{cm}$. The figures in the left column are S_{ph} along line $x = y = 0.02\,\text{cm}$, while the figures in the right column are contours of S_{ph} on the plane $z = 0.01\,\text{cm}$, with contour values 2×10^{18}, 2×10^{19}, $2\times10^{20}\,\text{cm}^{-3}\,\text{s}^{-1}$. The line color and format in the right-hand side subfigures are same as the one in the left-hand side in same row.

higher computational cost of the FMM classical Int method indicate its outstanding competitiveness among all the approaches for computing the photoionization rates. Additionally, the time usages of FMM classical Int method are stable for different problem settings, while that of SP_3 Larsen BC method varies significantly (see Tables 3.4–3.6).

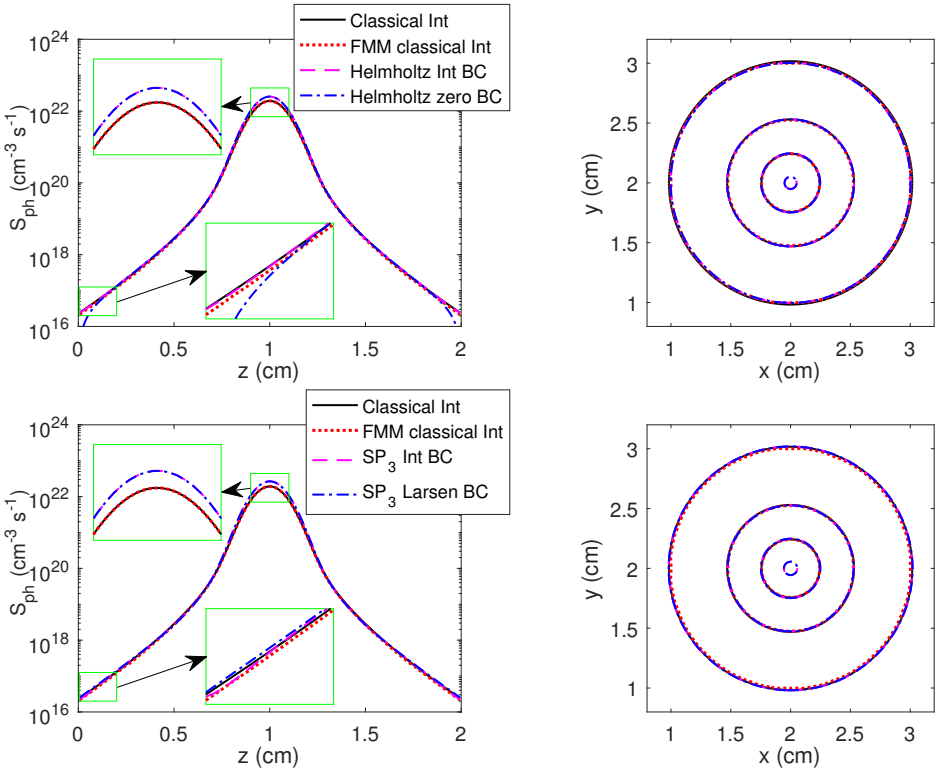


Figure 3.7: Photoionization rate S_{ph} calculated from one Gaussian source in (3.24). $x_d = y_d = 4\,\text{cm}$, $z_d = 2\,\text{cm}$, $\sigma = 0.1\,\text{cm}$. The figures in the left column are S_{ph} along line $x = y = 2\,\text{cm}$, while the figures in the right column are contours of S_{ph} on the plane $z = 1\,\text{cm}$, with contour values 2×10^{18}, 2×10^{19}, $2 \times 10^{20}\,\text{cm}^{-3}\,\text{s}^{-1}$. The line color and format in the right-hand size subfigures are same as the one in the left-hand size in same row.

3.3.2 Gaussian source with different pressures

The second example is to compute the photoionization rate $S_{\text{ph}}(\vec{x})$ in (3.1) generated from a single Gaussian radiation source, which is taken from [99, 100], in order to test the effect of the partial pressure of oxygen p_{O_2} in the kernel g given in (3.3). The Gaussian source of radiation I in (3.2) is taken as [23]

$$I(\vec{x}) = 4\pi\xi c \exp\left[-\frac{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}{\sigma^2}\right] \text{cm}^{-3}\,\text{s}^{-1}, \tag{3.25}$$

where c is the speed of light. Similar to [23], we take $\xi = 0.1$, $\sigma = 0.01\,\text{cm}$, $c = 2.99792458 \times 10^{10}\,\text{cm}\cdot\text{s}^{-1}$, $\delta = 5\sigma$ and $V = [0, 0.25] \times [0, 0.25] \times [0, 1.4]\,\text{cm}^3$. We

Table 3.6: Time usage and relative error of methods indicated in Table 3.3, for the case of one Gaussian source $x_d = y_d = 4\,\text{cm}$, $z_d = 2\,\text{cm}$, $\sigma = 0.1\,\text{cm}$.

Method	Time usage (s)	$\mathcal{E}_V$	$\mathcal{E}_\delta(\vec{x}_0)$
Classical Int	276089	—	—
FMM classical Int	27.6504	0.11%	2.64%
Helmholtz zero BC	3.44298	29.35%	5.06%
Helmholtz Int BC	3.60366+6902.23[a]	29.35%	5.06%
SP_3 Larsen BC	8.84413	36.23%	8.40%
SP_3 Int BC	6.91597+6902.23[a]	36.23%	8.40%

[a]Time usage to compute the boundary values, which is estimated from Classical Int method, with multiplication to a factor $2(n_x \times n_y + n_x \times n_z + n_y \times n_z)/(n_x \times n_y \times n_z)$.

fix the ratio of the partial pressure of oxygen and the air pressure $p_{O_2}/p = 0.2$ in this example. Finally, the box V is uniformly partitioned by $256 \times 256 \times 320$ cells.

In this example, when the partial pressure of oxygen p_{O_2} in (3.3) is lower, the photoionization rate decays slower as the distance from the emission source increases. Therefore, we would like to test the performance of different methods under different pressures p_{O_2}. The robustness of the methods under different pressures is of great significance in practical applications such as sprite discharges.

For comparison purpose, two different pressures are considered: (i) $p_{O_2} = 160\,\text{Torr}$; (ii) $p_{O_2} = 10\,\text{Torr}$. The photoionization rate along the central vertical line is plotted in Figures 3.8 and 3.9, and the time usage and the numerical error are shown in Tables 3.7 and 3.8.

Figure 3.8 shows that in the high-pressure case, all methods give similar results despite obvious mismatch of the peak values. As the pressure decreases, the discrepancy between different methods becomes more obvious, as shown in Figure 3.9. The curves given by the Helmholtz and SP_3 methods (with both boundary conditions) deviate significantly from the curves of Classical Int method in the low-pressure

Table 3.7: Time usage and relative error of methods indicated in Table 3.3, for the case of one Gaussian in (3.25) with $p_{O_2} = 160\,\text{Torr}$.

Method	Time usage (s)	$\mathcal{E}_V$	$\mathcal{E}_\delta(\vec{x}_0)$
Classical Int	292196	—	—
FMM classical Int	24.7371	0.15%	1.24%
Helmholtz zero BC	5.59405	31.93%	16.49%
Helmholtz Int BC	5.60994+6391.79[a]	31.93%	15.95%
SP_3 Larsen BC	17.6286	21.31%	8.74%
SP_3 Int BC	11.2337+6391.79[a]	21.31%	8.73%

[a] Estimated from the time usage of Classical Int method, with multiplication to a factor $2(n_x \times n_y + n_x \times n_z + n_y \times n_z)/(n_x \times n_y \times n_z)$.

Table 3.8: Time usage and relative error of methods indicated in Table 3.3, for the case of one Gaussian in (3.25) with $p_{O_2} = 10\,\text{Torr}$.

Method	Time usage (s)	$\mathcal{E}_V$	$\mathcal{E}_\delta(\vec{x}_0)$
Classical Int	292426	—	—
FMM classical Int	24.8619	0.19%	0.44%
Helmholtz zero BC	6.17337	88.73%	65.87%
Helmholtz Int BC	6.01811+6396.82[a]	88.29%	63.03%
SP_3 Larsen BC	23.5598	77.74%	35.11%
SP_3 Int BC	12.2226+6396.82[a]	77.71%	35.16%

[a] Estimated from the time usage of Classical Int method, with multiplication to a factor $2(n_x \times n_y + n_x \times n_z + n_y \times n_z)/(n_x \times n_y \times n_z)$.

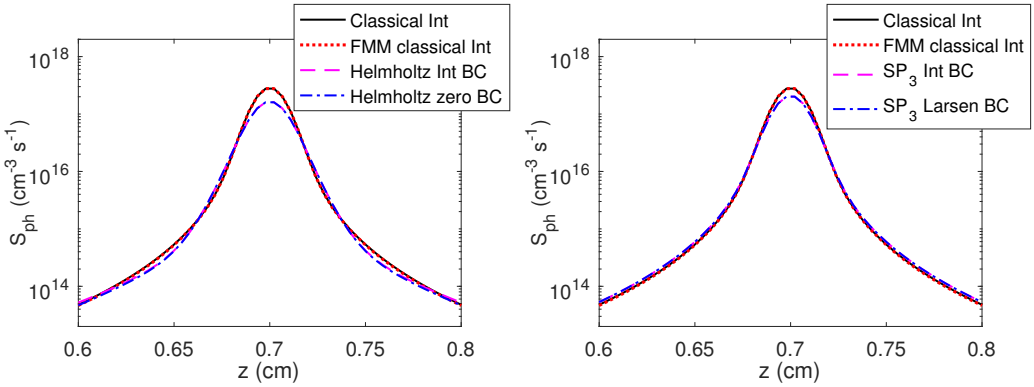


Figure 3.8: Photoionization rate S_{ph} along line $x = y = 0.125\,\text{cm}$, calculated from one Gaussian in (3.25) with $p_{O_2} = 160\,\text{Torr}$.

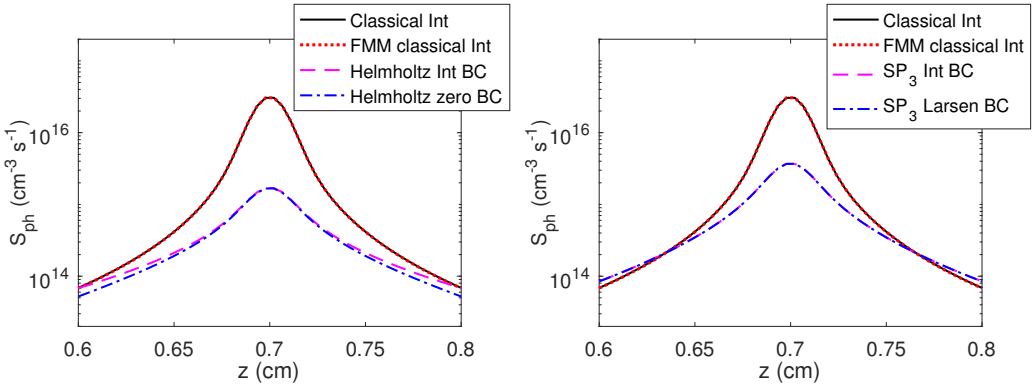


Figure 3.9: Photoionization rate S_{ph} along line $x = y = 0.125\,\text{cm}$, calculated from one Gaussian in (3.25) with $p_{O_2} = 10\,\text{Torr}$.

case. On the contrary, the results of the FMM classical Int method and the reference Classical Int method are in good agreement regardless of the air pressure. The values of the errors provided in Tables 3.7 and 3.8 again show the advantage of the FMM classical Int method in accuracy. In fact, for the case of low air pressure, the time used by the FMM classical Int is quite close to the method of SP_3 Larsen BC.

In order to see the relationship between the error and computational time with respect to different pressures, we compute more numerical examples under the same settings with different partial pressures of oxygen ranging from 10 Torr to 160 Torr. Results for the three most efficient method, i.e., FMM classical Int, Helmholtz zero

BC and SP_3 Larsen BC methods, are plotted in Figure 3.10. The FMM classical Int method always provides the most accurate results for all pressures, and the error is basically stable as the pressure varies. For the other two methods, the error increases as pressure decreases. Moreover, the global relative error $\mathcal{E}_V$ and the local relative error $\mathcal{E}_\delta(\vec{x}_0)$ near the center $\vec{x}_0$ of the box V of the FMM classical Int method are, in general, at least two order of magnitudes less than those of the other two methods. The time usage of the FMM classical Int method is also independent of pressure while the computation times of the other two methods increase slightly as the pressure becomes lower. As the pressure decreases, the time cost between the FMM classical Int method and the SP_3 Larsen BC method trends to be the same.

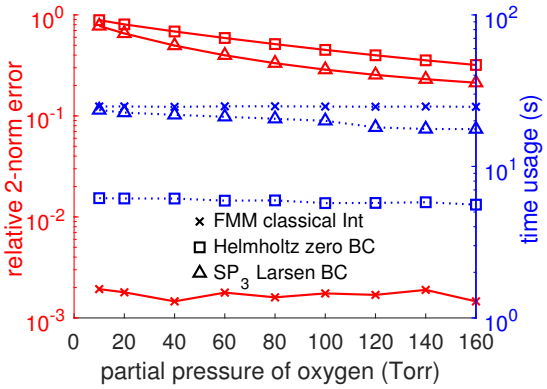


Figure 3.10: Error (red color, solid line) and time usages (blue color, dotted line) of FMM classical Int, Helmholtz zero BC and SP_3 Larsen BC methods with different air pressures.

3.4 Simulation of streamer discharge with photoionization

To further compare their performances of different methods for treating the photoionization $S_{\text{ph}}(\vec{x})$ in (3.1), we study the dynamics of streamers with photoionization, where S_{ph} appears as the source term of the transport of charged particles.

The streamer discharge between two parallel plates are used for comparision. The computational domain V is set to be a three-dimensional axis-aligned hyper-rectangle similar as Section 3.3, which is $V = V' = \Omega = (0, x_d) \times (0, y_d) \times (0, z_d)$ cm^3 with center $\vec{x}_0 = (x_0, y_0, z_0)^T = (x_d/2, y_d/2, z_d/2)^T$ cm. For the Poisson equation, the Dirichlet boundary conditions are applied on the two faces perpendicular to the z axis, i.e., $\phi = \phi_0$ on the upper face and $\phi = 0$ on the bottom face; and the homogeneous Neumann boundary conditions are applied on the other four faces. Homogeneous Neumann boundary conditions are assigned on all boundaries for n_e and the inflow boundaries for n_p. The initial value is set as

$$N_e(\vec{x}, t = 0) = N_p(\vec{x}, t = 0) = \tilde{N}_0(\vec{x}). \tag{3.26}$$

Coefficients in (1.3) are taken as the typical values in Section 1.2.2. The other physics parameters in (3.1)–(3.3) are chosen as [18, 144]: $p_q = 30$ Torr, $\xi = 0.1$, $\omega/\alpha = 0.6$.

The numerical method for discretizing (1.6) follows previous work in Chapter 2. For spatial discretization, the second-order MUSCL method with Koren limiter is applied to drift terms, and the central difference scheme is chosen for diffusion terms. The second-order explicit method (2.3)–(2.5) is adopted for time integration of (1.6) [182]. The multigrid-preconditioned FGMRES is used as the efficient algebraic elliptic solver to solve the Poisson equation in (1.3) iteratively. The iteration terminates when the relative residual is less than 10^{-8}. The other elliptic equations (3.6), (3.12) and (3.13) are solved by the same algebraic elliptic solver, with weaker stopping condition that the relative residual is less than 10^{-6}.

In the following numerical experiments, we will first study the effect on the inclusion of photoionization. After that, we compare the FMM method with other two efficient approximations in calculating the photoionization. Finally, the scalability of FMM for parallel computing is studied.

3.4.1 The effect of photoionization

To study the effect of photoionization, we simulate the interaction of two double-headed streamers using the fluid model (1.3): (i) with photoionization as a source term in transport equations; (ii) without photoionzation S_{ph}. The photoionization in this subsection is calculated by the SP_3 Larsen BC method introduced in Section 3.1.3, which is (3.12)–(3.13) with boundary conditions (3.14)–(3.15).

The initial value $\tilde{N}_0(\vec{x})$ in (3.26) is taken as a summation of two Gaussian plasmas [182]:

$$\begin{aligned}\tilde{N}_0(\vec{x}) = 10^{14}\Big(&\exp\left(-\left((x-0.47)^2+(y-0.5)^2+(z-0.41)^2\right)/(0.03)^2\right)\\ &+\exp\left(-\left((x-0.53)^2+(y-0.25)^2+(z-0.59)^2\right)/(0.03)^2\right)\Big)\text{cm}^{-3}.\end{aligned}$$

The simulation domain is chosen as $x_d = y_d = z_d = 1\,\text{cm}$, and it is partitioned by an uniform mesh of 1280^3 cells. Due to the usage of explicit temporal discretization, the time step Δt should be chosen as the minimum of CFL constraint (2.20) (with $C_\gamma = 1$) and the dielectric relaxation constraint (2.34). For safety, we times a constant ratio 0.5 to (2.20) and other ratio 0.6 to (2.34), and then take the minimum of these two numbers as the time step. The partial pressure of oxygen is taken as $p_{O_2} = 150\,\text{Torr}$.

Figures 3.11 and 3.12 show the electron density, net charge and electric field at $t = 0.5\,\text{ns}$ and $1.5\,\text{ns}$. We firstly focus on the top row, which are the results in the initial stage of the interaction at $t = 0.5\,\text{ns}$. These results exhibits little difference, indicating the photoionization is not prominent when two streamers start to interact. This little difference could be attributed to the strong impact ionization at the heads of two streamers with different polarities. As can be seen in the top row of Figure 3.11, the outward propagation of two streamers are slow compared with the interaction at the centers of two subfigures. The strong interaction with different polarities (observed in Figure 3.12) at the center of simulation region induces a strong electric field, which contributes to the strong ionization. This strong

ionization dominates the creation of electron-ion pairs and therefore the effect of photoionization is not so evident during initial stage of interaction. Nevertheless, the photoionization enhances the interaction, which can be seen in Figure 3.11 from the higher density at the central part in the top-left subfigure compared with the top-right one.

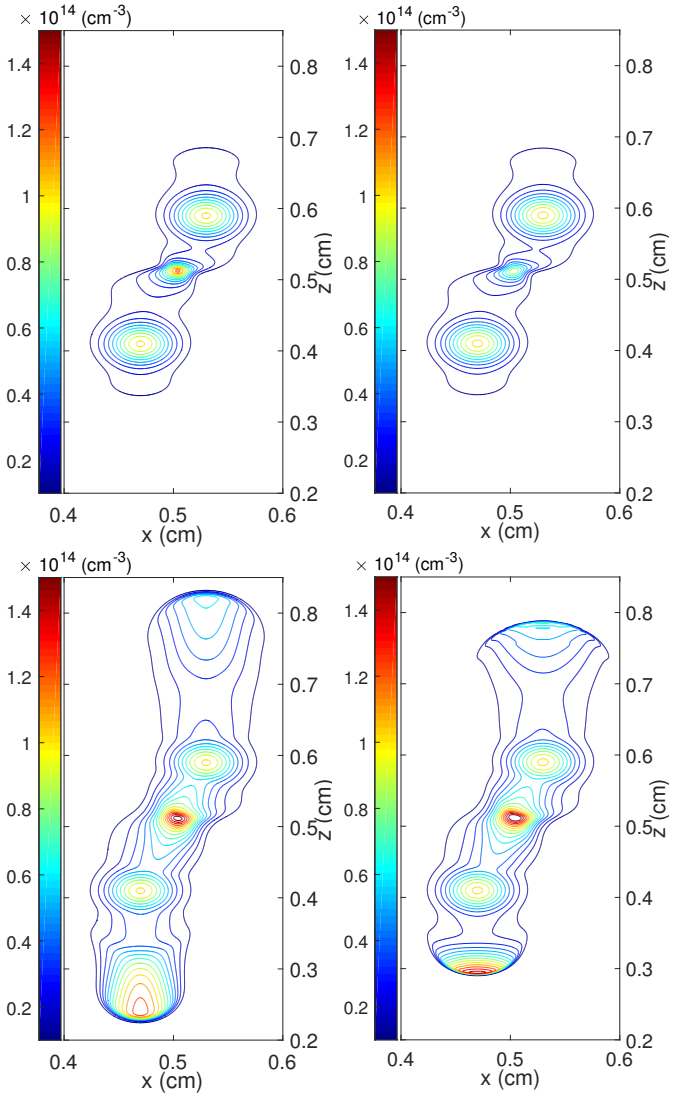


Figure 3.11: The contour of electron density on the $y = 0.5$ plane at $t = 0.5\,\text{ns}$ (top row) and $t = 1.5\,\text{ns}$ (bottom row). The subfigures in the left column are simulation results with photoionzation, while the subfigures in the right column are results without photoionization.

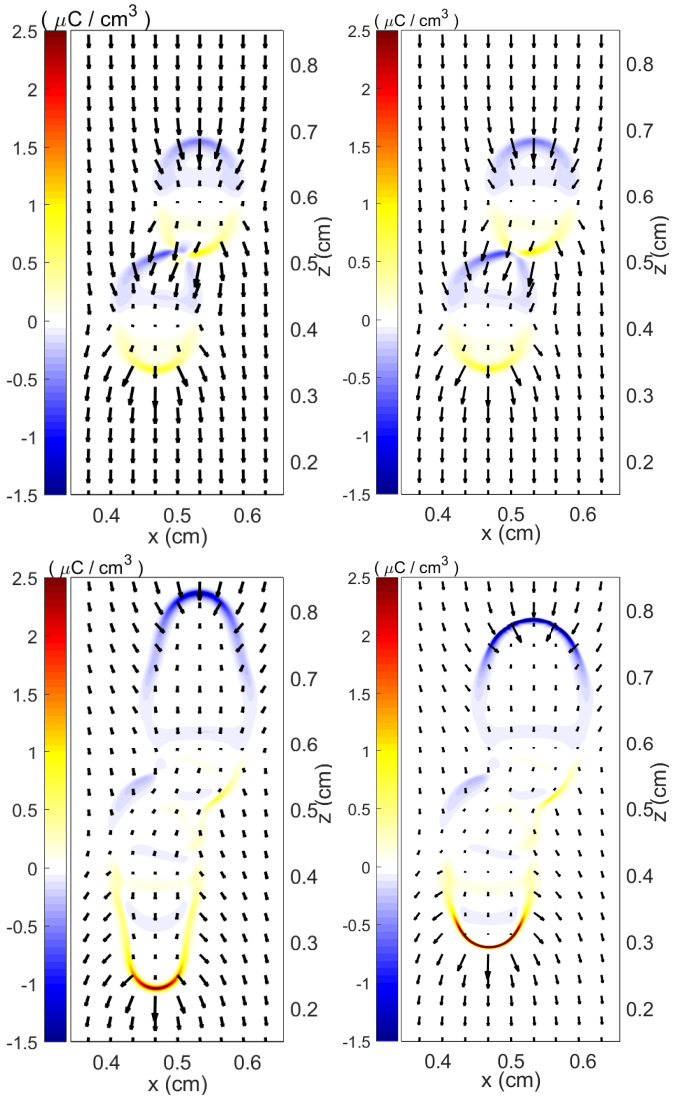


Figure 3.12: The net charge density and the electric field (E_x, E_z) on the $y = 0.5$ plane at $t = 0.5\,\text{ns}$ (top row) and $t = 1.5\,\text{ns}$ (bottom row). The subfigures in the left column are simulation results with photoionzation, while the subfigures in the right column are results without photoionization.

Next we compared the results for interactions at $t = 1.5\,\text{ns}$ when two streamers has already merged. This can be observed in the bottom rows in Figures 3.11 and 3.12. On the one hand, it can be seen clearly that the streamers propagate slower without considering the photoionization. This could be explained by the absence of seed electrons produced by photoionization in front of the heads of streamers. Without these photoionized seed electrons, the avalanche ahead of streamers is slow and lead to slower propagation. On the other hand, the structure of streamer heads exhibits obvious difference. The case considering the photoionization shows a smoother profile, which indicates that the photoionization has a smoothing effect to the charge distribution.

3.4.2 Comparison among different methods

In this subsection, we consider the interaction of two double-headed streamers and compare the numerical results of the three most efficient methods: FMM classical Int method, Helmholtz zero BC method and SP_3 Larsen BC method.

The initial value $\tilde{N}_0$ in (3.26) is taken as

$$\begin{aligned}\tilde{N}_0(\vec{x}) = 10^{14}\Big(&\exp\left(-\left((x-0.22)^2+(y-0.25)^2+(z-0.41)^2\right)/(0.03)^2\right)\\ &+\exp\left(-\left((x-0.28)^2+(y-0.25)^2+(z-0.59)^2\right)/(0.03)^2\right)\Big)\text{cm}^{-3}.\end{aligned}$$

The computational domain is fixed as $V = [0, 0.5] \times [0, 0.5] \times [0, 1]\,\text{cm}^3$, which is partitioned by a uniform grid of $512 \times 512 \times 1280$ cells. The time step is chosen as $\Delta t = 2.5 \times 10^{-3}\,\text{ns}$, where $1\,\text{ns} = 1 \times 10^{-9}\,\text{s}$. In order to see the interaction with respect to different p_{O_2}, we pick two values as $p_{O_2} = 1\,\text{Torr}$ and $p_{O_2} = 150\,\text{Torr}$, respectively, in our simulations.

We first compare the three methods by observing the electron density. The contours of the electron densities at 1.5 ns are shown in Figure 3.13, where the curves of different methods are plotted as different line styles and colours. Generally, the results of the three different methods are in good agreement in most part of the domain for both partial pressures of oxygen, while some differences can be

observed at the heads of streamers. The differences are particularly obvious at the head of positive streamer, which is zoomed in the same figure. The generally good agreement can be attributed to a stronger influence of the impact ionization comparing to the photoionization in the region with higher electric field, while the pronounced difference at the head of positive streamer may be due to the fact that photoionization plays a more important role in the propagation of positive streamers compared with the negative ones.

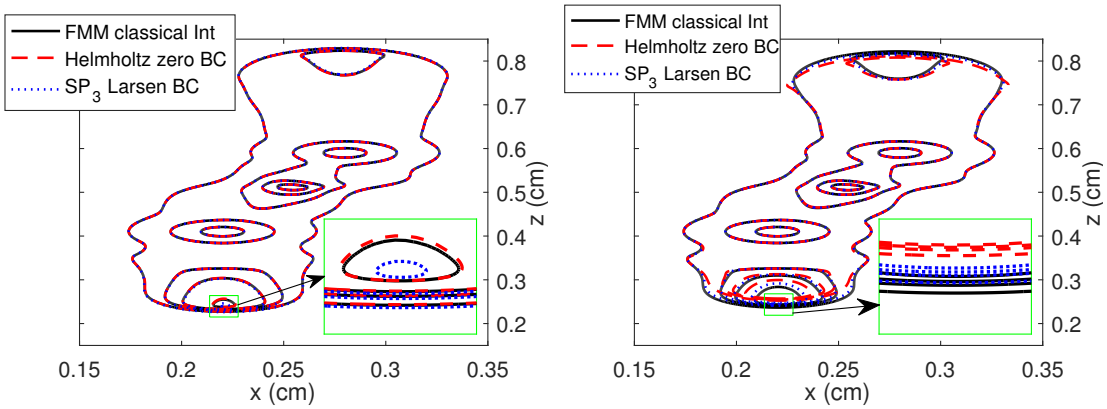


Figure 3.13: Contours of different electron density values at $n_e = 1 \times 10^{13}$, 5×10^{13}, 9×10^{13}, $1.3 \times 10^{14}\,\mathrm{cm}^{-3}$, on plane $y = 0.25\,\mathrm{cm}$ at $1.5\,\mathrm{ns}$ for $p_{O_2} = 150\,\mathrm{Torr}$ (left) and $p_{O_2} = 1\,\mathrm{Torr}$ (right).

Additionally, Figure 3.13 also displays the difference among three methods with respect to different p_{O_2}. As expected from Section 3.3.2, the difference between three methods are smaller in higher partial pressure of oxygen (150 Torr), and more observable when p_{O_2} is lower (1 Torr). This implies the validity of using the Helmholtz zero BC method and SP_3 Larsen BC for the photoionization in higher p_{O_2} and also the necessity of using the FMM classical Int method in lower p_{O_2}.

Besides observing the electron density at a fixed time 1.5 ns in Figure 3.13, the third component E_z of the electric field $\vec{E} = (E_x, E, y, E_z)^T$ along the line $x = y = 0.25\,\mathrm{cm}$ at 0.5 ns, 1.0 ns and 1.5 ns is also shown in Figure 3.14. As expected from Figure 3.13, the differences of the three methods are generally small, while the difference are easier to be observed near the heads of streamers (the leftmost and

rightmost minimum points). The difference is larger when p_{O_2} is small as 1 Torr, and the deviation increases over time, which is consistent with the results in [23]. One can see that, in the result of the FMM classical int method, the head of the streamer propagates slightly faster than the other two, which is possibly due to the underestimation of photoionization using the Helmholtz zero BC method and SP_3 Larsen BC method. As a summary, these results indicate that the accurate approximation of the photoionization could be significant in simulations with long-time propagation of streamers, especially when the partial pressure of oxygen is low.

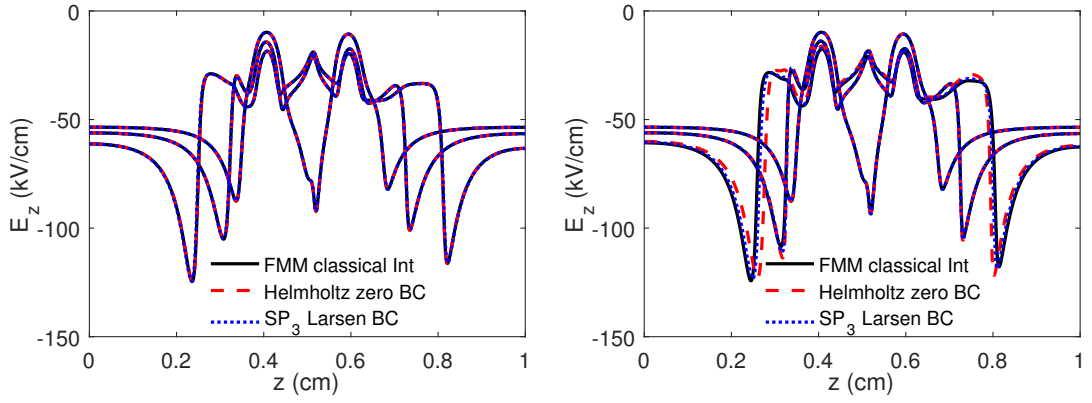


Figure 3.14: The third component E_z of the electric field $\vec{E}$ along line $x = y = 0.25\,\text{cm}$, at 0.5, 1.0 and 1.5 ns for $p_{O_2} = 150\,\text{Torr}$ (left) and $p_{O_2} = 1\,\text{Torr}$ (right).

3.4.3 Scalability of MPI parallelization

As demonstrated previously, one advantage of the FMM method is the scalability in parallel computing with distributed memory, which means the ability to reduce the execution time as the number of processes increases. In this subsection, we study the scalability of the FMM classical Int method, which is quantified by the relative speed-up, defined by the ratio of the execution time using the smallest number of cores over the execution time of the parallel program.

In this test, the governing equation is again (1.3), and the initial value $\tilde{N}_0$ in

Table 3.9: Time usage (s) using different nodes over two meshes. 20 cores are used in each node.

Number of nodes	Mesh size: $256 \times 256 \times 160$	Mesh size: $512 \times 512 \times 320$
1	3843.02	31198.5
2	2030.53	16329.4
4	1057.19	7998.79
8	569.106	4093.52
16	304.626	2116.02
32	157.651	1140.97
64	90.8405	621.827

(3.26) is set as one Gaussian,

$$\tilde{N}_0(\vec{x}) = 10^{14} \exp\left(-\left((x-0.2)^2 + (y-0.2)^2 + (z-0.1)^2\right)/(0.03)^2\right) \mathrm{cm}^{-3}.$$

All physics parameters are similar as those in previous subsection except stated otherwise. We set the computational domain as $[0, 0.4] \times [0, 0.4] \times [0, 0.2]\,\mathrm{cm}^3$, and adopt two uniform meshes with $256 \times 256 \times 160$ and $512 \times 512 \times 320$ grid cells. The simulation is run until 5×10^{-2} ns with a fixed time step 1×10^{-3} ns. It should be noted that S_{ph} is evaluated twice in each time step, and therefore the FMM classical Int method is applied 100 times in one simulation.

The time usage for the FMM classical Int method in whole simulation (100 evaluations) using different numbers of CPU cores is given in Table 3.9 and plotted in Figure 3.15, where a satisfactory scalability can be observed.

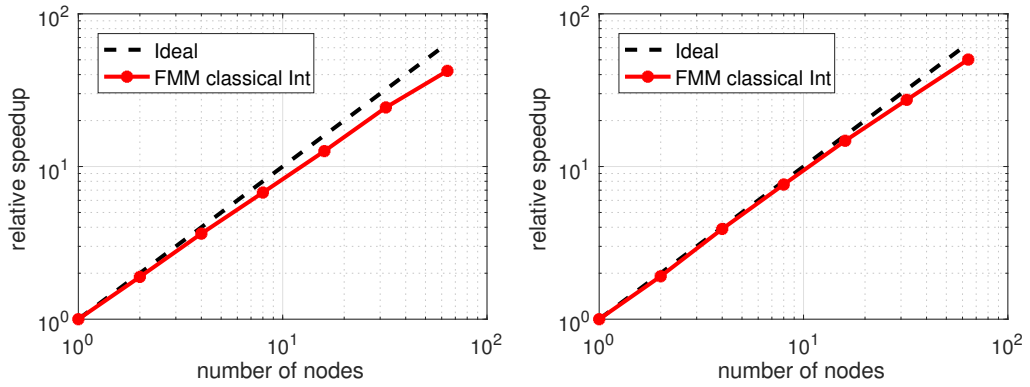


Figure 3.15: Relative speedups over two meshes: $256 \times 256 \times 160$ (left) and $512 \times 512 \times 320$ (right). 20 cores are used in each node.

Chapter 4

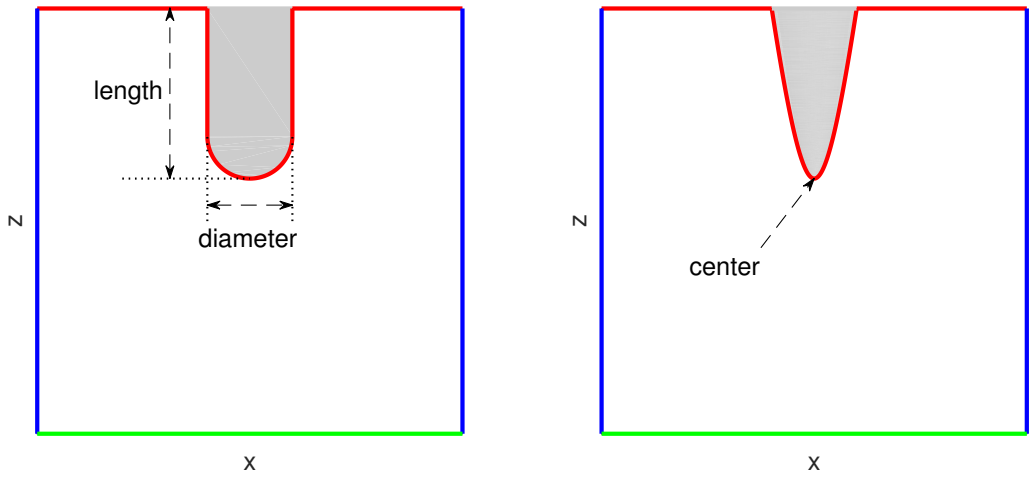
Simulation for Streamer Discharge inside General Electrodes

In addition to two flat-plate discharges, streamer discharge can be observed or triggered in electrodes of other shapes. This chapter intends to simulate streamer in general electrodes and further improve the efficiency with adaptive meshes. Firstly, the problems setting and corresponding boundary conditions are briefly demonstrated. Secondly, we illustrate the embedded boundary method to handle the general boundaries and discuss its convergence and stability. Thirdly, an adaptive mesh refinement method, together with the discussion of some refinement indicators, is introduced for the fluid equations. Finally, the generalized simulator is applied to study the effects of different shapes of anode and the interaction of streamers.

4.1 Problem setting and boundary conditions

To model the streamer discharge in the experiments with a pin or needle electrode (see experimental flowchart in Figure 1.3), we consider a typical domain Ω composed of a pin or needle anode on the upper domain with a flat-plate cathode on the bottom domain. Therefore, the domain Ω can be regarded as a cubic domain subtracts a pin anode. It should be noted that this is a typical shape in the experiment

[20,40,126,131], which we focus on in this chapter. However, the methods introduced in this chapter are not limited to this typical shape. Two examples of Ω are depicted in Figure 4.1, including a round-rod pin and a hyperbolic pin.



(a) round-rod pin (b) hyperbolic pin

Figure 4.1: Cross sections of two typical examples of simulation domain Ω. The shadow part is outside simulation.

As can be seen in Figure 4.1, the considered simulation domain Ω is a cubic domain $(x_0, x_1) \times (y_0, y_1) \times (z_0, z_1)$ without the shadow pin anode in the upper domain. Similar as Section 2.1, Dirichlet boundary conditions are applied on the upper and lower plate electrodes, i.e., $\phi|_{z=z_1} = \phi|_{(x,y,z)\in\text{curve electrode}} = \phi_0$ in the upper surface (where the cross section is red curve in Figure 4.1) and constant applied potential ϕ_0 is chosen, and $\phi|_{z=z_0} = 0$ in the lower flat plane (green segment in Figure 4.1). Homogeneous Neumann boundary conditions, which are $\frac{\partial\phi}{\partial x}|_{x=x_0,x_1} = 0$ and $\frac{\partial\phi}{\partial y}|_{y=y_0,y_1} = 0$, are applied on other four flat sides (blue segments in Figure 4.1). Initial conditions for n_e and n_p is taken as (2.1), where the same simplified assumption $n_{e,0}(\vec{x}) = n_{p,0}(\vec{x}) = n_0(\vec{x})$ with $n_0(\vec{x})$ a given function is taken. Homogeneous Neumann boundary conditions are applied at all the boundaries for n_e, and at all inflow boundaries for n_p.

To focus on the main mechanism of the fluid equation (1.6) in general electrodes, this chapter does not consider the non-local photoionization in the fluid equation. As a result, we assume $\tilde{S}_{\text{ph}} = 0$ in (1.6). However, the methods introduced in Chapter 3 including the fast multipole method can be directly applied to the calculation of photoionization in general electrodes.

The two simulation domains in Figure 4.1 are characterized by the upper pin anodes as follows:

Round-rod pin This round-rod pin is constructed by the union of a cylinder and a hemisphere in 3D. The length in Figure 4.1(a) denotes summation of the height of the cylinder and the radius of the hemisphere, and the diameter notation in the same figure is the diameter of the cylinder as well as the diameter of the hemisphere. As a result, this shape of pin is characterized by two parameters: length and diameter. In 2D, the round-rod pin is similarly constructed by the union of a rectangle and a semicircle, which can also be characterized by the same two parameters in Figure 4.1(a).

Hyperbolic pin This hyperbolic pin is depicted by a hyperbola, which gives its name "hyperbolic". The hyperbola has following expression

$$\frac{z^2}{z_c^2} - \frac{(x-x_c)^2 + (y-y_c)^2}{p|z_c|} = 1, \tag{4.1}$$

where $\vec{x}_c = (x_c, y_c, z_c)$ is the center of the pin (i.e., the tip of anode in Figure 4.1(b), which is also denoted as center in the same figure), and $p > 0$ is the semi-latus rectum. Therefore, the hyperbolic pin is characterized by a vector $\vec{x}_c$ and a positive scalar parameter p. In 2D with x, z axis, the constructing hyperbola is same as (4.1) with plugging $y = y_c$. Therefore, the 2D case is characterized by a vector (x_c, z_c) and the scala p.

4.2 Embedded boundary (EB) method for general domains

As can be seen from Figure 4.1, the boundary of simulation domain is not fully aligned with the coordinates and contains curves. Embedded boundary (EB) method offers one way to handle these curves on the boundary. It can be regarded as a modification of the structured Cartesian grid to embed the curve boundary.

Grid generation should be considered specifically for non-aligned simulation domain. Generally speaking, there are two typical ways to directly discretize general simulation domains with curve boundaries. One way is generating the grid using unstructured meshes [57,158]. When generating the grid, the boundary of domain is taken into consideration. Therefore, the grid points conform to the shape of boundary and spatial discretization should be considered on the unstructured meshes. Figure 4.2 shows an example of this unstructured meshes, which is generated for a 2D domain of Figure 4.1(a) by FreeFem++ [64].

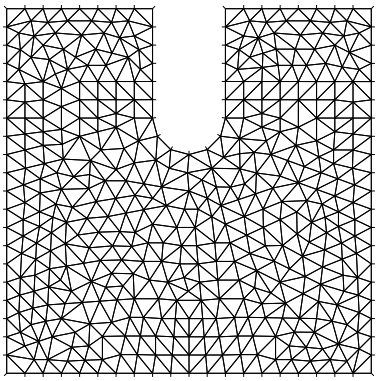
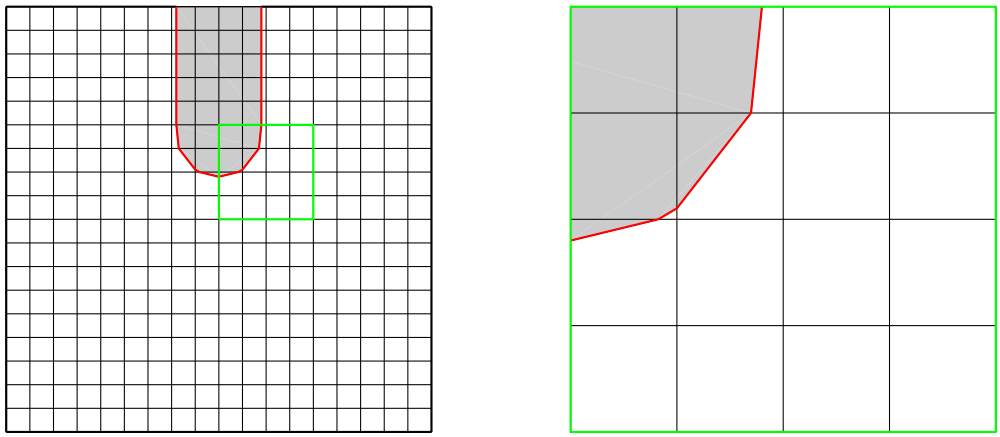


Figure 4.2: Unstructured grid of 2D domain from the cross section in Figure 4.1(a).

The other way of direct discretization is cutting a background structured grid by the boundary curve [32, 45, 117, 136], which is called immersed or embedded boundary method. The background Cartesian grid is firstly generated regardless of the boundary and then cut by it. Therefore, the grid do not conform to the boundary. Figure 4.3 illustrates one implementation of this grid, which is taken in

this thesis. Some background cells are cut by straight segments, endpoints of which are intersecting points of the curve boundary and the background Cartesian grid.



(a) discretized round-rod pin (b) zoom in the green square in the left subfigure

Figure 4.3: Structured grid with embedded boundary of 2D domain from the cross section in Figure 4.1(a).

Compared with the mentioned conforming unstructured grid, the EB method maintains the main structure of the Cartesian grid. As a result, the scheme introduced in Section 2 can be re-used for most cells in the grid without introducing other complicated discretization operators. In addition, the mesh generation is fairly simple, while the other unstructured grid should consider the boundary during generation. In some cases [47,139], the unstructured grid generation should take more effort to make sure the grid is of good quality everywhere. Furthermore, the background Cartesian grid in embedded boundary method paves the way for the geometric multigrid methods introduced in Section 2.4.

To take the advantages of EB method, we adopt it for spatial discretization in this thesis. However, although the grid generation is greatly simplified, it is not straightforward to impose boundary conditions in EB. As a result, the EB method requires additional modification in the vicinity of boundary. On the other hand, the

simple grid generation will bring some cells with poor quality (e.g., the second cell on the leftmost column in Figure 4.3(b)), which to be considered and modified if we use a similar scheme in Section 2. These modifications will be discussed in the following two subsections, which includes the scheme for transport equations and elliptic equations separately.

We simply comment here that there are some other ways for indirect grid generation [54, 93, 95, 157]. For example in [157], an additional computational region is introduced together with a mapping from the computational region to the physical region. This fixed computational domain simplifies the treatment of physical boundary regardless its shape and movement, however, the mapping need to be constructed and the original equations should be modified, which might be more complicated in terms of containing more terms or changing to variable coefficients.

We focus on the spatial discretization containing EB in this section. Nevertheless, all the temporal discretizations in Section 2.2 can be applied with the EB method, including the proposed second order semi-implicit schemes (2.12)–(2.13). To handle the occurring variable coefficient elliptic equations in semi-implicit schemes, the spatial discretization is considered for the equation with variable coefficient in Section 4.2.2.

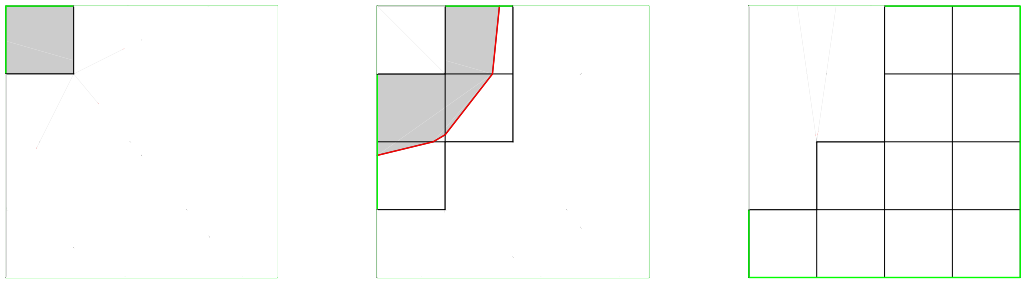
Before discussing schemes using EB, we would like to introduce three types of cell in the grid. Following the notations in [118], we classify cells occur in the EB method into three types:

1. regular cells: those cells inside simulation domain and not intersect with embedded boundary;

2. irregular cells: those cells which intersect with embedded boundary and partially inside simulation domain;

3. covered cells: those cells fully outside simulation domain.

The decomposition of cells in Figure 4.3(b) is shown in Figure 4.4 for illustration of the three types of cells.



(a) covered cell (b) irregular cells (c) regular cells

Figure 4.4: Decomposition of three types of cells in Figure 4.3(b).

4.2.1 EB for transport equations

In this subsection, we focus on the spatial discretization of transport equations, which are first two equations in (1.6). The idea mainly comes from [31, 35, 38, 118], while we change the temporal and spatial discretization to be consistent with what we introduced in Section 2.2 and 2.3. For easy explanation, we will introduce the idea in 2D. Nevertheless, the extension to 3D will be briefly discussed, which is generally straightforward due to the usage of structured Cartesian grid.

The discretization for covered cells and regular cells is fairly simple. For covered cells, they are located fully outside the simulation domain. Therefore, we do not put state variables on these cells and no discretization is required for covered cells. For regular cells, they are exactly identical to the Cartesian cells introduced in Section 2.3. As a result, similar discretization is applied on them, which are the second-order MUSCL scheme with the monotonized central (MC) limiter [163]. The details can also be found in the following part when we discuss the scheme on the irregular cells, where the regular cells can be regarded as a special case of the irregular ones.

The scheme on irregular cells is a composition of conservative update, preliminary update and redistribution. The conservative update is an ordinary finite volume method for spatial discretization, and the preliminary update and redistribution [31] are included to maintain the CFL constraint 2.20 regardless of the shape of irregular

cells.

In order to illustrate the conservative update, discretized variables should be introduced together with some geometry information of the irregular cells. Discretized state variables $(n_e)^n_{i,j}$ and $(n_p)^n_{i,j}$ are defined at the center of full i,j-th Cartesian cells, even if the cells are partially covered. It should be noted that i,j is ordered from the background Cartesian grid regardless of the embedded boundary. The length (area) of each boundary segment (face) of the irregular cells, the area (volume) of partial cell and the normal vector of the EB segment (face) is required in 2D (3D). For simplicity, we assume $\Delta x = \Delta y = h$. We represent length of top segment of i,j-th cell as $l_{i,j+1/2}h$, where $0 \leq l_{i,j} \leq 1$ denotes the ratio of length. Similar notations are applied for $l_{i\pm1/2,j}$ and $l_{i,j-1/2}$. The length of segment of EB in i,j-th cell is denoted as $l^{EB}_{i,j}h$, and its normal vector is denoted as $\vec{n}_{i,j}$, which is pointed to the simulation domain. The area of the i,j-th cell is $\Lambda_{i,j}h^2$, where $\Lambda_{i,j}$ denotes the ratio of area. These notations are illustrated in Figure 4.5, where we pick one irregular cell from Figure 4.4(b).

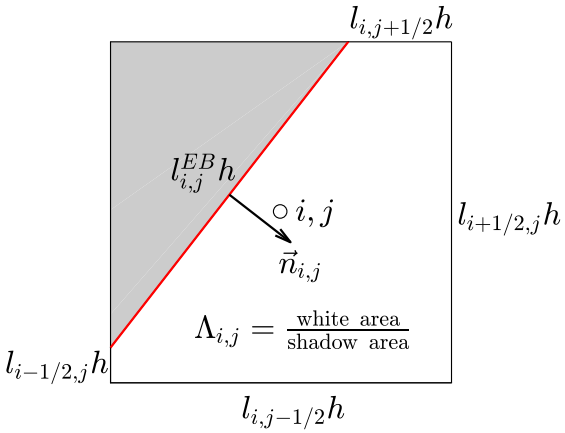


Figure 4.5: Geometry notations of an irregular cell.

With the information in Figure 4.5 constructed, the conservative update of first

equation in (2.2) in 2D is given as

$$\frac{(n_e)_{i,j}^{C,n+1}-(n_e)_{i,j}^{n}}{\tau_n}-\frac{1}{\Lambda_{i,j}h}\Big(l_{i+1/2,j}F_{i+1/2,j}^{n}-l_{i-1/2,j}F_{i-1/2,j}^{n}+l_{i,j+1/2}F_{i,j+1/2}^{n}-l_{i,j-1/2}F_{i,j-1/2}^{n}$$
$$-l_{i,j}^{EB}F_{i,j}^{n}\Big)=\tilde{\alpha}(|\vec{E}_{i,j}^{n}|)\tilde{\mu}_e|\vec{E}_{i,j}^{n}|(n_e)_{i,j}^{n}, \tag{4.2}$$

where $F_{i,j+1/2}^{n}$ is the numerical flux of $\tilde{\mu}_e E_y^n n_e^n+\tilde{D}_{e,y}\partial_y n_e^n$ at the center of partial boundary segment at $(i,j+1/2)$ and $F_{i,j}^{n}$ is the numerical flux $(\tilde{\mu}_e\vec{E}^n n_e^n+\tilde{D}_e\nabla n_e^n)\cdot\vec{n}_{i,j}$ located at the center of the segment of embedded boundary in (i,j) cell. Similarly, $F_{i,j-1/2}^{n}$ and $F_{i\pm1/2,j}^{n}$ are numerical fluxes at the centers of partial boundary segments at $(i,j-1/2)$ and $(i\pm1/2,j)$, respectively.

The scheme for the second equation in (2.2), which is the transport equation for n_p, is similar as (4.2). It has same form as (4.2), but replacing n_e by n_p on the left-hand side of equality and changing the numerical flux to $-\tilde{\mu}_p\vec{E}^n n_p^n$. Together, this explicit update of n_e and n_p can be similarly applied to other temporal schemes in Section 2.2, with proper modification to the superscripts of n_e, n_p and $\vec{E}$ according to different schemes. Furthermore, the conservative update (4.2) can be extended to 3D by adding two more fluxes on z direction, letting the fluxes defined on the centroids of cell boundaries and changing the ratio l and Λ to ratio of area and volume, respectively.

It should be noted that fluxes F^n in (4.2) is defined on the centers of partial cell boundary, while state variable $(n_e)_{i,j}^{n}$ is located at the center of regular cells. We will show more details of the construction of fluxes F^n at the end of this subsection, after introducing nonconservative fluxes $\tilde{F}^n$ in preliminary update.

The construction of nonconservative fluxes $\tilde{F}^n$ do not need geometry information in Figure 4.5. In fact, $\tilde{F}^n$ is defined at the center cell boundaries of the regular background Cartesian cell. The different locations of $\tilde{F}^n$ and F^n are depicted in Figure 4.6. $\tilde{F}^n$ is constructed following similar procedure of the MUSCL scheme in regular cells: first, we evaluate the slopes of each direction in every cells (with MC limiter); second, we get the values of state variable at two sides of every face of the irregular cells; third, we get the flux $\tilde{F}^n$ by solving the Riemann problem (which is

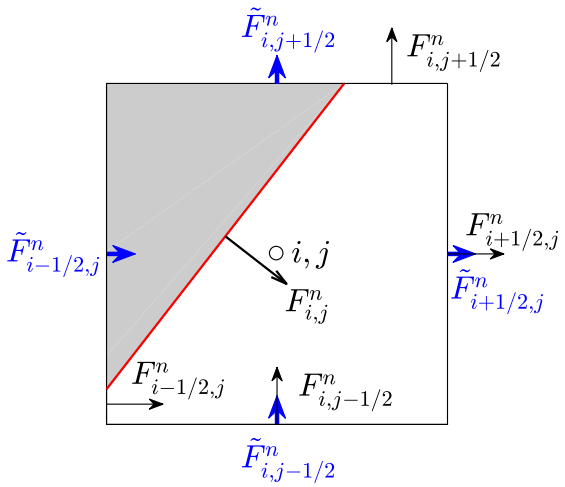


Figure 4.6: Fluxes $\tilde{F}^n$ and F^n.

taking upwind flux in our problem) for two-side values. There are two special cases where only one-side value can be achieved. In the first case when the cell boundary is aligned with the boundary of domain, we set the other value required in the Riemann problem identical to this one-side value due to the homogeneous Neumann boundary condition. In the second case when the cell boundary is fully outside the simulation domain, we take the idea from [118], which extrapolates using the slope and state values from one adjacent cell to provide the other value required in the Riemann problem. For example, if face $(i-1/2,j)$ in Figure 4.6 is fully covered, then the two values required in Riemann problem $(n_e)^L_{i-1/2,j}$ and $(n_e)^R_{i-1/2,j}$ are given as

$$(n_e)^R_{i-1/2,j} = (n_e)^n_{i,j} - \frac{\Delta x}{2}(\sigma_x)^n_{i,j}, \qquad (n_e)^L_{i-1/2,j} = (n_e)^n_{i+1,j} - \frac{3}{2}((n_e)^n_{i+2,j} - (n_e)^n_{i+1,j}),$$

where $(\sigma_x)^n_{i,j}$ is the slope in x direction at cell (i,j).

After the construction of nonconservative fluxes $\tilde{F}^n$, the nonconservative update

is represented at all irregular and regular cells as:

$$\frac{(n_e)_{i,j}^{NC,n+1}-(n_e)_{i,j}^{n}}{\tau_n}-\frac{1}{h}\left(\tilde{F}_{i+1/2,j}^{n}-\tilde{F}_{i-1/2,j}^{n}+\tilde{F}_{i,j+1/2}^{n}-\tilde{F}_{i,j-1/2}^{n}\right) = \tilde{\alpha}(|\vec{E}_{i,j}^{n}|)\tilde{\mu}_e|\vec{E}_{i,j}^{n}|(n_e)_{i,j}^{n}, \tag{4.3}$$

which is regardless of the geometry information in Figure 4.5. Compared with the conservative update (4.2), the nonconservative update (4.3) is stable with the CFL-type constraint (2.20), since it does not consider the geometry information in the irregular cells. However, (4.3) is not a conservative update, and it is only first-order accurate in irregular cells [118].

To achieve a more accurate results but with CFL-type constraint independent of $\Lambda_{i,j}$, a linear combination between $(n_e)_{i,j}^{C,n+1}$ and $(n_e)_{i,j}^{NC,n+1}$ is constructed to form a preliminary state $(n_e)_{i,j}^{P,n+1}$:

$$(n_e)_{i,j}^{P,n+1}=\eta_{i,j}(n_e)_{i,j}^{C,n+1}+(1-\eta_{i,j})(n_e)_{i,j}^{NC,n+1}, \tag{4.4}$$

where $0\le\eta_{i,j}\le 1$. Following necessary condition $\eta_{i,j}/\Lambda_{i,j}=O(1)$ for stability shown in [118], we take $\eta_{i,j}=\Lambda_{i,j}$.

The preliminary (4.4) has a stability constraint regardless of Λ, however, it is not globally conservative. To ensure the conservation, the following conservative error in cell (i,j)

$$\Lambda_{i,j}h^2\left((n_e)_{i,j}^{C,n+1}-(n_e)_{i,j}^{P,n+1}\right) \tag{4.5}$$

should be redistributed into a neighborhood of regular and irregular cells adjacent to cell (i,j) [31, 118]. This neighborhood includes (i,j) itself, and the weight of each cell $(\hat{i},\hat{j})$ in the neighborhood of (i,j) is taken as its volume $\Lambda_{\hat{i},\hat{j}}$. By this locally volume-weighted redistribution, the conservative error (4.5) is added back and therefore global conservation is constructed.

We have almost finished the explanation of scheme including EB for the transport equations. As a summary, there is no state variable defined on covered cells. Evolution on regular cells takes conservative update (4.2) or (4.3) since these two

equations are identical for regular ones. State variables on irregular cells is firstly evaluated by preliminary update (4.4), which requires both conservative update (4.2) and nonconservative update (4.3). Then the conservative error in irregular cells (4.5) is redistributed to a neighborhood of the cells.

The construction of each fluxes F^n in (4.2) finishes the clarification of scheme using EB method. On the one hand, in order to construct the flux $F^n_{i,j}$ located at the EB boundary, we first utilize state value $(n_e)^n_{i,j}$, slopes $(\sigma_x)^n_{i,j}$ and $(\sigma_y)^n_{i,j}$ to get a state value at the center of EB segment. Then by the homogeneous Neumann boundary condition and Riemann solver, we get the flux values $F^n_{i,j}$. It should be noted that the velocity for this flux is evaluated from $-\tilde{\mu}_e \vec{E}^n_{i,j} \cdot \vec{n}_{i,j}$. On the other hand, other four fluxes in (4.2) are constructed from linear interpolation of nonconservative fluxes $\tilde{F}$. One example can be found in Figure 4.7, where $F^n_{i,j+1/2}$ is interpolated from

$$\begin{aligned} F^n_{i,j+1/2} &= (1-\bar{l})\tilde{F}^n_{i,j+1/2} + \bar{l}\tilde{F}^n_{i+1,j+1/2} \\ &= \frac{1+l_{i,j+1/2}}{2}\tilde{F}^n_{i,j+1/2} + \frac{1-l_{i,j+1/2}}{2}\tilde{F}^n_{i+1,j+1/2}, \end{aligned} \tag{4.6}$$

where $l_{i,j+1/2}$ comes from geometry information of cell (i,j) (see Figure 4.5).

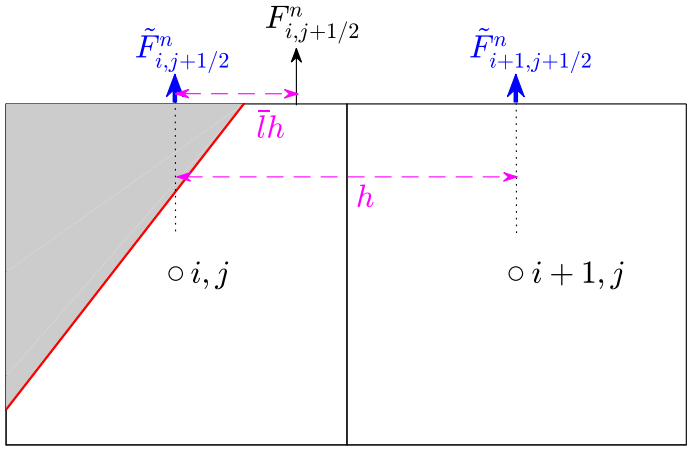


Figure 4.7: Linear interpolation of fluxes $\tilde{F}^n_{i,j+1/2}$ and $\tilde{F}^n_{i+1,j+1/2}$ to get $F^n_{i,j+1/2}$.

4.2.2 EB for elliptic equations

The idea of embedded boundary method with interpolation in Section 4.2.1 can also be applied to elliptic equations for potential ϕ. After solving the discrete potential ϕ, the discrete electric field $\vec{E}$ can be viewed as the negative slope of ϕ, where the method and coding for slope calculation in Section 4.2.1 could be reused. Therefore in this subsection, we focus on spatial discretization of variable coefficient elliptic equation for potential ϕ on background Cartesian grid with embedded boundary. The illustration in this subsection is mainly based on EB method proposed in [72, 143]. For simplicity, the idea will be introduced in 2D.

To avoid lengthy expression of the variable coefficient and the right-hand side in (2.9) or (2.14), we introduce $\kappa(x, y)$ and $\rho(x, y)$ to denote the coefficient and right-hand side, respectively, and consider discretization on the following elliptic equation:

$$-\nabla \cdot (\kappa(x, y)\nabla\phi) = \rho(x, y), \tag{4.7}$$

where ϕ satisfies Dirichlet or Neumann boundary condition following the problem setting in Section 4.1. Here the superscript of ϕ is not shown explicitly due to the elliptic equations in the fluid model could be solved on different time stages (e.g. n, $n + 1/2$ or $n + 1$), depending on the temporal scheme in Section 2.2. Similarly, the superscripts of κ and ρ are also omitted.

The general idea of EB method for elliptic equations is similar as the method for transport equations in Section 4.2.1. The discretization is different for three types of cells in Figure 4.4. For covered cells, we do not need to set discrete $\phi^n_{i,j}$ on them. For regular cells, the classical second-order central difference scheme is adopted, which has a similar form as (2.17). Ghost point method is taken if the regular cells are adjacent to boundaries aligned with coordinates. For irregular cells, the discrete state variable $\phi^n_{i,j}$ is defined at the center of regular background grid. Finite volume method is applied on each irregular cell, resulting in a flux-difference or discrete divergence form. Then interpolation is introduced to make the flux located at the

center of partial cell boundary.

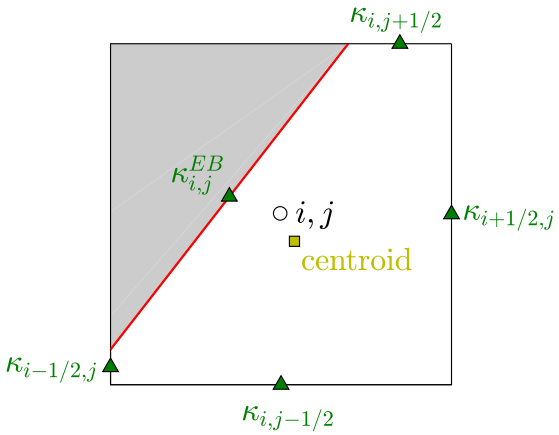


Figure 4.8: Locations of κ and the centroid of irregular cell.

We take the same irregular cell in Figure 4.5 to illustrate the scheme for elliptic equation (4.7). For simplicity and consistence to Figure 4.5, we assume $\Delta x = \Delta y = h$. By integrating (4.7) both sides over the irregular cell (integration is taken only for the white part in Figure 4.5) and dividing $\Lambda_{i,j}h^2$, the following scheme can be derived:

$$-\frac{1}{\Lambda_{i,j}h}\Big(l_{i+1/2,j}\kappa_{i+1/2,j}F_{i+1/2,j} - l_{i-1/2,j}\kappa_{i-1/2,j}F_{i-1/2,j} + l_{i,j+1/2}\kappa_{i,j+1/2}F_{i,j+1/2} - l_{i,j-1/2}\kappa_{i,j-1/2}F_{i,j-1/2} - l^{EB}_{i,j}\kappa^{EB}_{i,j}F_{i,j}\Big) = \bar{\rho}_{i,j}, \quad (4.8)$$

where $\kappa_{i,j+1/2}$ is the value of $\kappa(x,y)$ evaluated at the center of the partial segment on edge $(i, j+1/2)$, and similar notations are introduced for $\kappa_{i,j-1/2}$ and $\kappa_{i\pm1/2,j}$; $F_{l,m}$ are fluxes through each segment of boundary of the irregular cell, for $l = i \pm 1/2$ and $m = j \pm 1/2$; $F_{i,j}$ is the flux through the embedded segment; $\bar{\rho}_{i,j}$ is the volumed average of $\rho(x,y)$ over the irregular cells, or the value of $\rho(x,y)$ evaluated at the centroid of the irregular cell. It should be noted that fluxes F in (4.8) approximate the normal derivatives on the cell boundaries. The locations of discrete κ and the

centroid of irregular cell are depicted in Figure 4.8. In 3D, these κ are evaluated at the centroids of each boundary face.

Next the detailed construction of fluxes F in (4.8) is demonstrated. The idea is similar as the construction in transport equations in previous subsection. As shown in Figure 4.6, we introduce fluxes $\tilde{F}$ in the first step as

$$\tilde{F}_{i,j+1/2} = \frac{\phi_{i,j+1} - \phi_{i,j}}{h}, \qquad \tilde{F}_{i+1/2,j} = \frac{\phi_{i+1,j} - \phi_{i,j}}{h},$$

and then linearly interpolate from $\tilde{F}$ at the midpoints of regular background Cartesian edges to F at the midpoint of irregular edge in the second step. The linear interpolation follows same form of (4.6), which is also demonstrated in Figure 4.7.

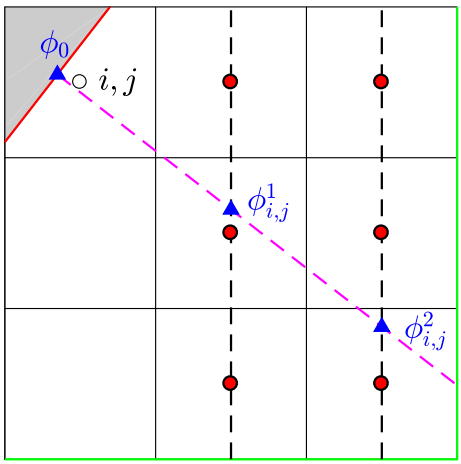


Figure 4.9: The construction of EB fluxes on the red edge. A quadratic polynomial is constructed from three values ϕ_0, $\phi^1_{i,j}$ and $\phi^2_{i,j}$ on the blue triangles. The first one takes value from Dirichlet boundary condition. The other two take values from quadratic interpolation by discrete ϕ on three red circles along y direction. The direction is chosen perpendicular to the largest-magnitude component of $\eta_{i,j}$. All triangles locate on the line starting at the midpoint of the red EB edge with direction $\eta_{i,j}$, which is plotted as pink dash line.

The construction of fluxes F in (4.8) is clear except $F_{i,j}$, which is located at

the EB edge. Although Dirichlet boundary condition is applied to the embedded boundary in problem setting, we simply comment here that the value of $F_{i,j}$ is provided if Neumann boundary condition is applied. Compared with the Neumann condition, the implementation of Dirichlet condition is a little complicated. We adopt the quadratic interpolation in [72], which is constructed by three points along the normal direction $\eta_{i,j}$ of the EB edge. One point is picked as the midpoint of the EB edge in cell (i,j), and the value $\phi = \phi_0$ at this point is provided by the boundary condition. The other two points $\phi^1_{i,j}$ and $\phi^2_{i,j}$ are selected as the first two intersection points of: the line along direction $\eta_{i,j}$; and two parallel grid lines which are perpendicular to the direction of the largest-magnitude component of $\eta_{i,j}$ but not contains $\phi_{i,j}$. The two parallel lines are plotted as black dash lines in Figure 4.9, and the three points for quadratic interpolation are plotted as blue triangles in the same figure. In Figure 4.9, if we denote the distance between the triangle of ϕ_0 and $\phi^1_{i,j}$ as $d^1_{i,j}$ and denote the distance between two triangles ϕ_0 and $\phi^2_{i,j}$ as $d^2_{i,j}$, then the flux $F_{i,j}$ in (4.8) is given as

$$F_{i,j} = \frac{1}{d^2_{i,j} - d^1_{i,j}} \left(\frac{d^2_{i,j}}{d^1_{i,j}} \left(\phi_0 - \phi^1_{i,j} \right) - \frac{d^1_{i,j}}{d^2_{i,j}} \left(\phi_0 - \phi^2_{i,j} \right) \right),$$

which is the derivative at the point of ϕ_0 of the quadratic interpolating polynomial crossing ϕ_0, $\phi^1_{i,j}$ and $\phi^2_{i,j}$. Furthermore, the two values $\phi^1_{i,j}$ and $\phi^2_{i,j}$ are evaluated by the quadratic interpolation from discrete ϕ, which is located at three red circles at the parallel black dash line containing $\phi^{1,2}_{i,j}$ in Figure 4.9.

We give a summary of the scheme for the elliptic equation 4.7 using EB method. The scheme takes a flux-differencing form as shown in (4.8), which is applied to all regular cells and irregular cells. In regular cells, $l^{EB}_{i,j} = 0$ and the scheme (4.8) is identical to the classical second-order central difference scheme. In irregular cells, the fluxes in all (partial) edges aligned to the coordinates are constructed by linear interpolation from the fluxes defined at the midpoints of background regular cells. If Dirichlet boundary condition is applied on EB boundary, the flux $F_{i,j}$ at the EB edge is constructed as the derivative of a quadratic polynomial. This quadratic

polynomial lies along with the direction of normal vector $\eta_{i,j}$, where three points are depicted in Figure 4.9.

The multigrid preconditioned elliptic solver introduced in Section 2.4 can be used to solve the algebraic equation from (4.8). For the ease of programming, we take multigrid as a solver in this chapter. As mentioned in (2.30), this solver can also be viewed as a Richardson iteration with multigrid preconditioner. Due to the inclusion of EB method, the multigrid solver should take minor adjustments on smoothing and restriction in Figure 2.1.

The smoothing procedure is partitioned to two steps [72]. In the first step, Jacobi iteration is performed only for the irregular cells, with fixed values in the regular cells. In the second step, red-black Gauss-Seidel relaxation is taken for regular cells, with fixed values in the irregular cells.

The restriction is based on coarsening the grid. Figure 4.10 shows the procedure of coarsening from four adjacent cells in finer layer to one cell in the coarser layer. We comment here that the top-left finer cell of these four cells is a covered cell, which contains no discrete state variable ϕ in finer layer. The EB edge in coarser cell is taken as the segment from intersection points of coarser background Cartesian grid and the boundary, which can be seen in the right subfigure of Figure 4.10.

Although the construction of coarser EB edge do not conserve the volume of original four finer cells, we follow [72] and define the volume of coarser cell as the sum of volumes of four finer cells. If some finer cells are covered cells, the volume of them is zero. To be consistent with this definition of volume, the restriction in Figure 2.1 is taken as the volume-weighted average of the residuals in the finer cells.

With the modification of smoothing and restriction, the V-cycle multigrid method is applicable on the algebraic elliptic equation with EB method. It should be noted that the prolongation is identical to the previous one introduced in Figure 2.1, which is the constant interpolation.

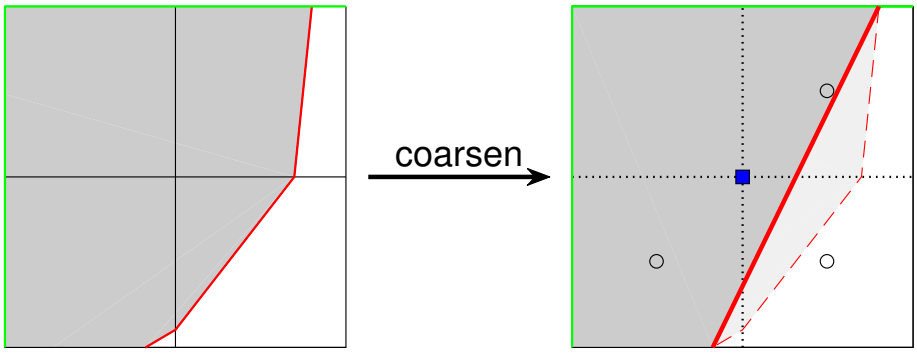


Figure 4.10: Illustration of coarsening procedure. Four finer cells combine to one coarser cell centered at blue square. The coarser value is taken as the volume-weighted average of all three non-covered finer value centered at circles. The red solid line in the right subfigure shows the EB edge of this coarser cell.

4.2.3 Convergence and stability

In this subsection, we would like to adopt the dimensionless model problem in 2D, to show the inclusion of embedded boundary method still keeps second-order convergence in spatial discretization, and not affect the stability with respect to the dielectric relaxation constraint and small volume ratio Λ of irregular cells. The following 2D model problem has similar form as (1.6) but without photoionization

$$\begin{cases} \dfrac{\partial n_e}{\partial t} - \nabla \cdot \left(\tilde{\mu}_e \vec{E} n_e\right) - \tilde{D}_e \Delta n_e = S \exp(-K/|\vec{E}|)\tilde{\mu}_e |\vec{E}| n_e, \\ \dfrac{\partial n_p}{\partial t} + \nabla \cdot \left(\tilde{\mu}_p \vec{E} n_p\right) = S \exp(-K/|\vec{E}|)\tilde{\mu}_e |\vec{E}| n_e, \qquad x \in \Omega, \quad t > 0, \\ -\lambda \Delta \phi = n_p - n_e, \qquad \vec{E} = -\nabla \phi, \end{cases} \tag{4.9}$$

where domain Ω is taken as the $(0,1) \times (0,1)$ subtracting a round-rod pin with length 0.15 and diameter 0.1; other parameters are taken as: $\tilde{\mu}_e = 1$, $\tilde{\mu}_p = 0.09$, $\tilde{D}_e = \text{Diag}(1 \times 10^{-4}, 1 \times 10^{-4})$, $S = 1000$ and $K = 4$. The dimensionless applied voltage in the boundary condition is taken as $\phi_0 = 1$, i.e., $\phi|_{y=1} = \phi|_{(x,y) \in \text{curve electrode}} = 1$ and

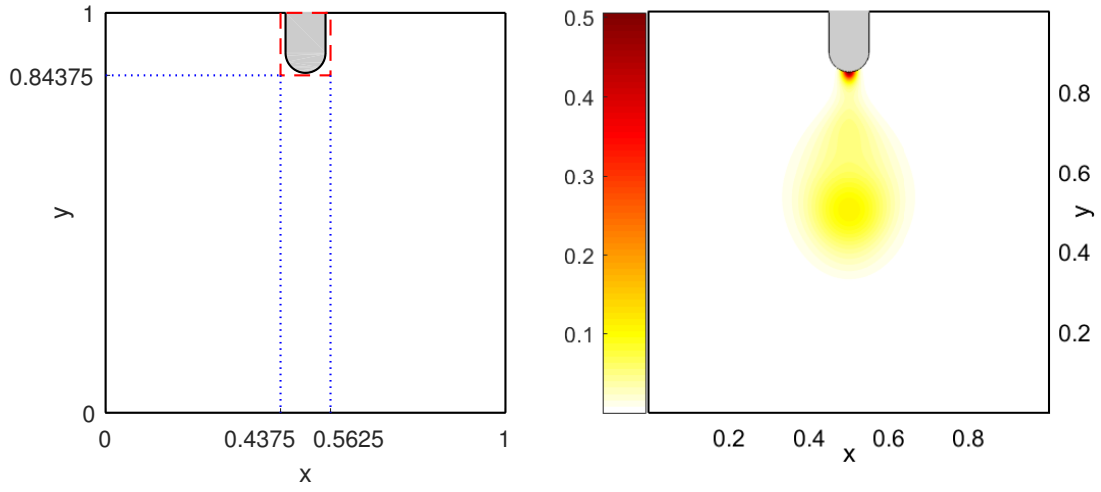
$\phi|_{y=0} = 0$. Homogeneous boundary condition is applied to other two boundaries as $\frac{\partial \phi}{\partial x}|_{x=0,1} = 0$. Homogeneous Neumann boundary conditions are applied at all the boundaries for n_e, and at all inflow boundaries for n_p. The parameter λ and initial condition will be given later. Constant time steps $\tau_n = \tau$ are used, and the computation is performed until $T = 0.03125$. To concentrate on the embedded boundary method, uniform mesh is adopted, and we take $\Delta x = \Delta y = h$. The criterion for elliptic equation is taken as (2.37) with $\varepsilon = 10^{-8}$ hereafter in this chapter.

Study of convergence In this testing example, we set $\lambda = 10^{-3}$ in (4.9). The initial value is $n_e(x, y, t = 0) = n_p(x, y, t = 0) = 10^{-6} + 0.1 \exp(-100[(x - 0.5)^2 + (y - 0.5)^2])$. For all the calculations in this example, we fix the ratio of the time step to the grid size at $\tau/h = 0.125$. The finite volume method with unlimited linear reconstruction is applied for spatial discretization. The "exact solution" for $(n_e)_{\text{ref}}$ and $(n_p)_{\text{ref}}$ are calculated by second-order explicit scheme (2.3)–(2.5) with $\tau = 0.03125/2^9$ which is sufficiently small. The error is calculated on a smaller domain, to prevent the case when the coarser cell contains some covered cells (e.g. Figure 4.10). The simulation domain, the domain for error estimate and the reference solution $(n_e)_{\text{ref}}$ are depicted in Figure 4.11. The numeric results are given in Table 4.1.

Table 4.1: L^2-norm error of the second-order semi-implicit scheme (2.12)–(2.13) in a 2D pin-plane problem.

Δt	$0.03125/2^4$	$0.03125/2^5$	$0.03125/2^6$	$0.03125/2^7$
$\|n_e - (n_e)_{\text{ref}}\|$	5.0027×10^{-3}	1.5912×10^{-3}	3.5364×10^{-4}	6.6669×10^{-5}
order	–	1.6526	2.1698	2.4072
$\|n_p - (n_p)_{\text{ref}}\|$	5.9607×10^{-3}	1.9943×10^{-3}	4.7795×10^{-4}	9.9153×10^{-5}
order	–	1.5796	2.0610	2.2691

The results in Tables 4.1 demonstrate that the EB method with semi-implicit



(a) simulation domain and the domain for error estimate

(b) reference solution $(n_e)_{\text{ref}}$ at $t = T$

Figure 4.11: Simulation domain, the domain for error estimate and reference solution in the problem for study of convergence. Simulation domain is the white region surrounded by black solid line, while the domain for error estimate is the the simulation outside red dash rectangle.

scheme (2.12)–(2.13) is indeed second order accurate, though the order has larger oscillation compared with the 1D example without EB in Section 2.6.1.

Study of stability In this example, we study the stability of EB method in two aspects. First, we check if the CFL-type stability condition (2.20) holds if the area of some irregular cells is very small. Second, we check the inclusion of EB method will not break the property of relaxing the dielectric relaxation constraint of the proposed second-order semi-implicit scheme (2.12)–(2.13).

We perform the calculation with $\lambda = 5 \times 10^{-5}$ in (4.9). The initial value is $n_e(x, y, t = 0) = n_p(x, y, t = 0) = 10^{-6} + \exp(-1000[(x - 0.5)^2 + (y - 0.85)^2])$. MC limiter is applied in spatial discretization.

Similar as Section 2.6.1, we introduce τ_{diel} and τ_{CFL} to denote the constraint of time step for the dielectric relaxation and CFL-type stability, respectively. In 2D, τ_{diel} is identical to (2.34), and CFL-type constraint is similar as (2.20) and (2.35), which is

$$\tau \le \tau_{\text{CFL}} = \frac{h}{C_\gamma \tilde{\mu}_e (|E_x|_{\max} + |E_y|_{\max}) + 2(\tilde{D}_{e,x} + \tilde{D}_{e,y})/h}. \tag{4.10}$$

To get a good estimation of τ_{diel} and τ_{CFL} for this test problem, we first perform the simulation on a very fine mesh $h = 1/2048$ with $\tau = 1/32768$, using second-order explicit scheme (2.3)–(2.5), and record the maximum values of $|E_x|$, $|E_y|$ and $(\tilde{\mu}_p n_p + \tilde{\mu}_e n_e)$. Then, we plug the maximum value of $(\tilde{\mu}_p n_p + \tilde{\mu}_e n_e)$ to get an estimation of τ_{diel} defined in (2.34), and the result is $\tau_{\text{diel}} = 4.3326 \times 10^{-5}$. To estimate τ_{CFL}, we insert the estimated value of maximum of $|E_x|$ and $|E_y|$ into (4.10), with C_γ set to be 2 and h chosen as a relatively larger cell size $h = 1/256$, which yields $\tau_{\text{CFL}} = 4.2658 \times 10^{-4}$. Thus we have $\tau_{\text{CFL}} \approx 9.8\tau_{\text{diel}}$, meaning that a much better efficiency can be achieved if we can break the dielectric relaxation time constraint. The numerical tests presented below will be carried out on the uniform background Cartesian grid with $h = 1/256$.

For this fixed testing uniform grid, we found the smallest ratio of area over all irregular cells is $\Lambda_{i,j} \approx 8.3724 \times 10^{-3}$. One of the cells with smallest ratio is depicted

in Figure 4.12.

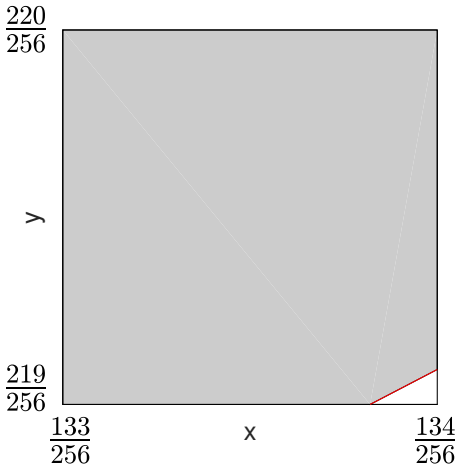


Figure 4.12: One irregular cell with smallest ratio of area $\Lambda_{i,j}$. Mesh size: $h = 1/256$.

Four different time steps, i.e., $0.1\tau_{\text{diel}}$, τ_{diel}, $3\tau_{\text{diel}}$ and $9\tau_{\text{diel}}$, are used to test the stability. All these four time steps are less than τ_{CFL}, and stability condition (2.35) is always satisfied for all simulations before numerical blow-up occurs. We consider a simulation to be unstable if $n_e > 10$ is detected in this example. According to our experiments, this condition always leads to a quick numerical blow-up of the solution. Similar as Section 2.6.1, we implemented four temporal discretizations for comparisons: the proposed semi-implicit scheme (2.12)–(2.13), the first-order explicit scheme (2.2), the first-order semi-implicit scheme (2.7), and second-order explicit scheme (2.3)–(2.5). Moreover, in order to see the effect of preliminary update (4.3) and redistribution (4.5) for cells with small ratio of area, we also implement the classical conservative update (4.2) over all irregular cells for comparison. The results of stability testing are summarized in Table 4.2.

The results in first several rows in Table 4.2 shows the stability of the dielectric relaxation constraint over four schemes, and they are similar as the results in Table 2.3. These results clearly show that the two semi-implicit methods remain stable when the time step exceeds τ_{diel} and reaches $9\tau_{\text{diel}}$. In contrast, the two explicit

Table 4.2: Stability of different temporal discretizations on a 2D pin-plane problem.

Temporal scheme (with preliminary update)	$\tau = 0.1\tau_{\text{diel}}$	$\tau = \tau_{\text{diel}}$	$\tau = 3\tau_{\text{diel}}$	$\tau = 9\tau_{\text{diel}}$
2^{nd} order semi-implicit (2.12)–(2.13)	stable	stable	stable	stable
1^{st} order semi-implicit (2.7)	stable	stable	stable	stable
2^{nd} order explicit (2.3)–(2.5)	stable	stable	unstable	unstable
1^{st} order explicit (2.2)	stable	stable	unstable	unstable
Temporal scheme (conservative update)	$\tau = 0.1\tau_{\text{diel}}$	$\tau = \tau_{\text{diel}}$	$\tau = 3\tau_{\text{diel}}$	$\tau = 9\tau_{\text{diel}}$
2^{nd} order semi-implicit (2.12)–(2.13)	unstable	unstable	unstable	unstable
1^{st} order semi-implicit (2.7)	unstable	unstable	unstable	unstable
2^{nd} order explicit (2.3)–(2.5)	unstable	unstable	unstable	unstable
1^{st} order explicit (2.2)	unstable	unstable	unstable	unstable

methods exhibit instability. As indicated, the stability condition (2.35) is still fulfilled for all the time steps and schemes. Therefore the instability is caused by the violation of the dielectric relaxation time constraint, and the semi-implicit schemes truly allow larger time steps on this occasion with the inclusion of the EB method.

On the other hand, comparison between top rows and bottom rows in Table 4.2 also shows the conservative update itself is unstable for the CFL-type constraint (4.10). It could exhibit instability with the EB method, in the case when some cut cells are small compared with the background regular cells. The small cut cells could bring a small factor in the CFL-type stability constraint and require a very small time step (e.g. less than $0.1\tau_{\text{diel}}$ in Table 4.2) for a stable update. As a result, the inclusion of preliminary update allows a larger and more robust time step to different shapes of boundary and mesh size in the EB method.

4.3 Adaptive mesh refinement (AMR) method

As can be seen in the previous figures of streamer discharge (e.g, Figures 4.11, 3.11, and 2.10), the width of streamer is small compared with the typical size of

simulation domain. This implies only a few cells in the grid of simulation domain are "important". Furthermore, in order to capture the important thin layer ahead of streamer propagation, a relative fine mesh should be taken. Therefore, if an uniform mesh is adopted during the simulation of streamer, the time cost is large even when parallelism has been taken, and the accurate simulation on those "not important" cells might be not necessary. To avoid some unnecessary calculation and make our simulator more efficient, we would like to consider adaptive mesh refinement (AMR) during simulation.

The adaptive skill has been adopted in the fluid model in [10,14,48,109,116,132, 154] and achieved great success. The multiscale structure of streamer discharge is captured with less number of fine mesh, which reduces the total DOFs and allows a faster simulation. Among the mentioned literature, the majority uses a multi-layer Cartesian adaptive mesh generation [48,109,132,154]. On the other hand, in [10,116], local adaptive method is applied to unstructured mesh; in [14], the moving mesh method is adopted, which could be viewed as achieving best accuracy with fixed computational cost. As can be seen in Section 4.2, we take a background Cartesian grid with EB to handle the curve boundary of domain. Due to the existence of this Cartesian grid, we would like to implement the adaptive mesh based on a multi-layer Cartesian grid for the ease of programming, where the AMR method used in this thesis mainly comes from [11, 13, 112].

Generally speaking, AMR adopts a hierarchy of several layers of Cartesian grids. Starting from a coarse Cartesian uniform mesh on the first layer, AMR finds those cells with unsatisfactory refinement in the first layer and refines them by subdividing these cells into r^d equal-sized sub-cells, where r is the ratio of division in each dimension and d is the dimension of problem. The refined sub-cells from the first layer form the second layer of the hierarchy. This refining procedure could continue if the some cells in the second layer are still not satisfactory. Then we continue to refine and subdivide those under-refined cells in the second layer and get the third layer. This procedure continues until all leaf cells are properly refined or the

hierarchy reaches maximum number of layers. An example of the AMR grid is demonstrated in Figure 4.13.

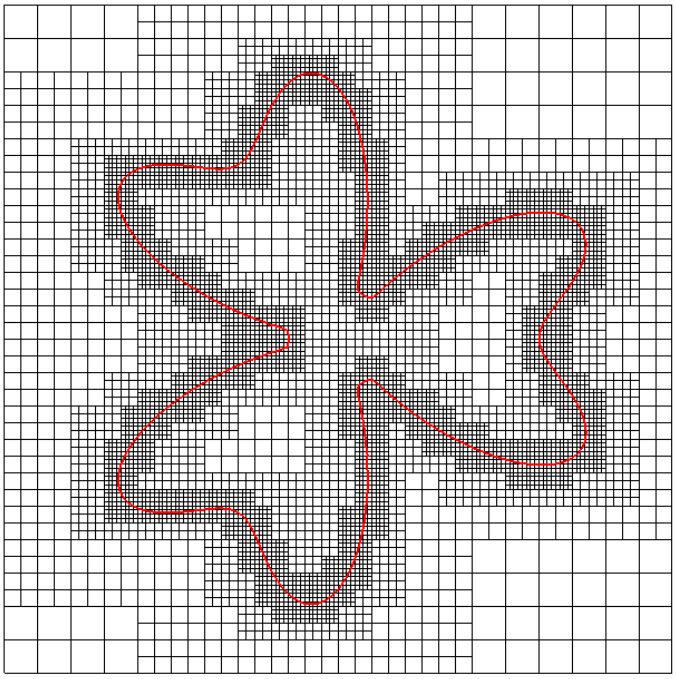


Figure 4.13: An example of the AMR grid with four layers. The mesh is required to be refined near the red curve.

Following the block structured approach in [11, 13], the hierarchy of an AMR grid can be described by four notations: level index l, level grid G_l, k-th patch at level l as $G_{l,k}$ and (i,j)-th individual cell $I^l_{i,j}$. We assume level index $1 \leq l \leq l_{\max}$, where $l = 1$ is the level of coarsest grid and $l_{\max}$ is the finest level. Moreover, we assume the coarsest grid is an uniform Cartesian mesh which covers the simulation domain Ω. At each level l, the level grid G_l is the union set of all the cells $I^l_{i,j}$ in this level. For simplicity, we assume identical and uniform mesh size in all direction at level l as $h_l = \Delta x_l = \Delta y_l$, where Δx_l and Δy_l are the mesh size in x, y direction in level l, respectively. Therefore, all the cells $I^l_{i,j}$ in level l have same mesh size h_l, and the index (i,j) of this cell is characterized globally at each level. If we consider the background domain as $[x_0, x_1] \times [y_0, y_1]$, then $I^l_{i,j} = [x_0 + ih_l, x_0 + (i+1)h_l] \times [y_0 + jh_l, y_0 + (j+1)h_l]$. We further introduce the ratio $r_{l,l+1}$ between two adjacent level

l and $l+1$, which is an integer larger than 1 (typical equal to 2) and $h_l = r_{l,l+1}h_{l+1}$. The remaining notation $G_{l,k}$ is introduced from the block structured approach, which serves as a bridge between $I^l_{i,j}$ and G_l. At each level l, the level grid G_l is represented as the union set of all patches $G_{l,k}$, which is $G_l = \cup_k G_{l,k}$. Each patch $G_{l,k}$ is a rectangle in 2D, and two characteristic corners of this patch is $(x_0+i_{k,1}h_l, y_0+j_{k,1}h_l)$ and $(x_0 + i_{k,2}h_l, y_0 + j_{k,2}h_l)$. Hence, $G_{l,k} = \{I^l_{i,j} | i_{k,1} \leq i < i_{k,2}, j_{k,1} < j_{k,2}\}$. As a result, the AMR with block structured approach does not refine the under-refined cells individually. Instead, it tries to covered the under-refined cells by finer patches.

The inclusion of patch $G_{l,k}$ between G_l and $I^l_{i,j}$ bring more computation due to the fact that some cells in one patch might do not need refined. However, the patch structure has better efficiency in data communication compared with individual cells [112], which is beneficial for parallel computing. In fact, a efficiency number $0 < r_{\text{ef}} \leq 1$ is introduced to restrict the ratio of cells need to be refined over the total number of cells in one patch. If r_{ef} is small, there could be less number of large finer patches to covered all the cells need to be refined, but bring more unnecessary refinement. On the other hand, if r_{ef} is large, there is less unnecessary refinement, but the number of patches could be large. To illustrate the usage of patch, we reuse the example in Figure 4.13 and dye different adjacent patches with different colors in the second and third layer of grid. The color patches are shown in Figure 4.14.

There are two requirements in the generation of grid hierarchy to ensure "properly nested" [11, 112]. First, one fine patch should starts and ends at the corner of cells in the adjacent coarser level grid. It should be commented that this requirement does not mean the fine patch is contained in one coarser patch, which can be seen from the top-left shadow patch in the right subfigure in Figure 4.14. This shadow patch cross two patches in the next coarser layer (green color, light-blue color in the left subfigure). Second, there must be one cell in level G_{l+1} separating a cell in G_l and a cell in G_{l+2}. The requirement makes sure the scheme and communication can be constructed between two adjacent levels, instead of jumping from one level to next two coarser or finer level.

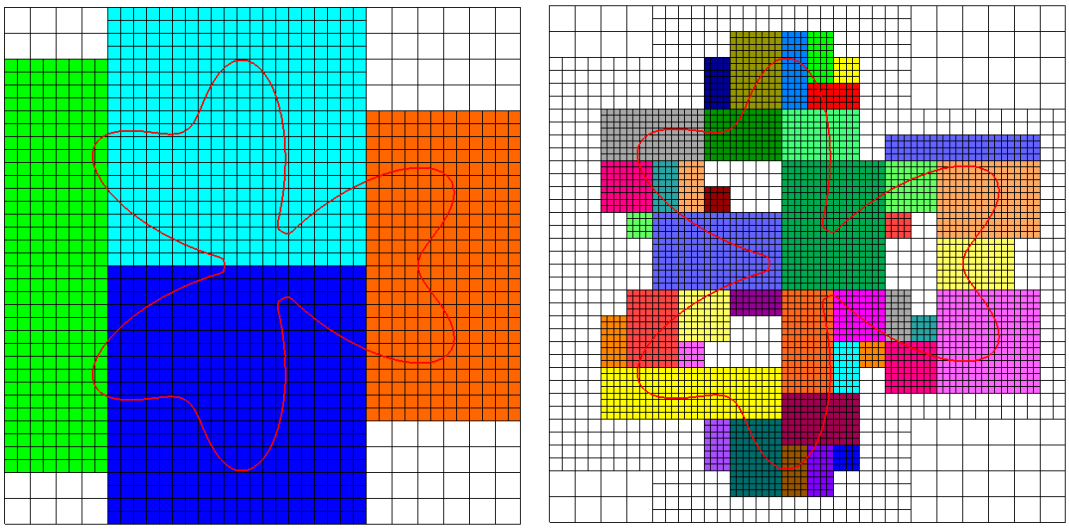


Figure 4.14: The AMR grid in Figure 4.13 with different dyed patches in the second layer (left subfigure) and the third layer (right subfigure).

With the "properly nested" AMR hierarchy, numerical method at level l could be designed using the information merely from level $l+1$ and $l-1$. Furthermore, the numerical scheme could be applied to each patches instead of individual cells. We would like to introduce the numerical method in AMR for transport equations in Section 4.3.1 and elliptic equations in Section 4.3.2.

4.3.1 AMR for transport equations

In this subsection, we focus on the numerical scheme on a grid with AMR for the transport equations, which are first two equations in (1.6). The scheme is mainly based on [11, 13], which solved hyperbolic conservation laws with AMR grid. To sketch the idea in a simple way, we will introduce the scheme in 2D. As can be seen in the following description, the extension to 3D is relatively straightforward due to the usage of background Cartesian grid. Furthermore, we fix the grid hierarchy in this section and leave the discussion of grid generation in Section 4.3.3.

Similar as Section 4.2, the first order explicit temporal discretization (2.2) is

taken for the description of scheme with AMR. Nevertheless, all temporal discretization in Section 2.2 could be adopted with straightforward modification.

To sketch the idea of AMR on the transport equations, we first consider the scheme outside the vicinity of domain boundary. These cells are regular and not effected by the redistribution from irregular cells. Therefore, we do not need to consider the preliminary and redistribution procedure in Section 4.2.1 for these cells, and the scheme is identical to (4.2) or (4.3). However, the definition of flux F^n should be remedied if the flux is defined on the edge which locates at the coarse-fine interface. To illustrate this modification, we take the cell (i_c, j_c) in Figure 4.15 as an example, where two-level grid is considered and the subscript c denotes the index in coarse level and f denotes the one in fine level.

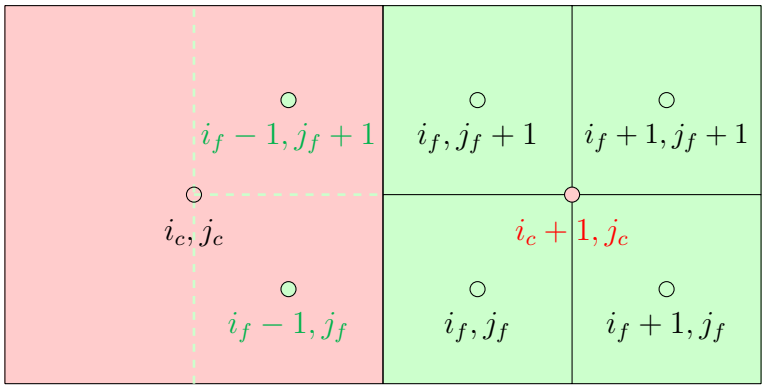


Figure 4.15: Some regular cells in two AMR levels. They are away from the boundary and near the coarse/fine interface of these two levels. The value at (i_c+1, j_c) is evaluated from average of four values in finer grid. The value at (i_f-1, j_f+1) and (i_f-1, j_f) is evaluated from bilinear interpolation on coarser grid.

The scheme on coarse cell (i_c, j_c) is formally given as

$$\frac{(n_e)^{n+1}_{i_c,j_c} - (n_e)^n_{i_c,j_c}}{\tau_n} - \frac{1}{h_{l_c}}\Big(F^n_{i_c+1/2,j_c} - F^n_{i_c-1/2,j_c} + F^n_{i_c,j_c+1/2} - F^n_{i_c,j_c-1/2}\Big) = S^n_{i_c,j_c}, \tag{4.11}$$

where the source term $S^n_{i_c,j_c} = \tilde{\alpha}(|\vec{E}^n_{i_c,j_c}|)\tilde{\mu}_e|\vec{E}^n_{i_c,j_c}|(n_e)^n_{i_c,j_c}$. The scheme on other finer cells is formally identical to (4.11). If all the stencil cells required to construct flux

$F^n_{i_c+1/2,j_c}$ are in $G_{l_c} - G_{l_f}$, then the flux $F^n_{i_c+1/2,j_c}$ is identical to the one introduced in Section 4.2.1, which is constructed by the all the information in the cells at same level. On the other hand, as seen in Figure 4.15, the $F^n_{i_c+1/2,j_c}$ lies in the interface between this level and next finer level. In this case, $F^n_{i_c+1/2,j_c}$ is modified to be the summation of all fluxes which lies in the edge $(i_c + 1/2, j_c)$ and constructed in the finer level

$$F^n_{i_c+1/2,j_c} = \frac{1}{r_{l_c,l_f}} \sum_{k=0}^{r_{l_c,l_f}-1} F^n_{i_f-1/2,j_f+k}, \tag{4.12}$$

where r_{l_c,l_f} is the ratio between level l_c and level l_f, which is equal to 2 in Figure 4.15. As a result, scheme 4.11 can be rewritten as

$$\frac{(n_e)^{n+1}_{i_c,j_c} - (n_e)^n_{i_c,j_c}}{\tau_n} - \frac{1}{h_{l_c}}\Big(\frac{1}{r_{l_c,l_f}} \sum_{k=0}^{r_{l_c,l_f}-1} F^n_{i_f-1/2,j_f+k} - F^n_{i_c-1/2,j_c} + F^n_{i_c,j_c+1/2} - F^n_{i_c,j_c-1/2}\Big) = S^n_{i_c,j_c}. \tag{4.13}$$

This scheme preserve conservation if the source is zero. It should be noted that we take identical time step τ_n for all levels. However, if considering transport equations merely, a better choice should be $\tau^n_c = r_{l_c,l_f}\tau^n_f$. In other words, a better choice should be fixing the ratio of time step and mesh size on each level. In this choice, (4.12) should be modified to accumulate the flux $F^n_{i_f-1/2,j_f+k}$ from t_n to $t_n+(r_{l_c,l_f}-1)\tau^n_f$. We take uniform time step on every levels in this thesis due to the coupling of Poisson equation. The electric field should be calculated at each intermediate finer time step, which should be considered carefully, in particular to the dielectric relaxation constraint.

Before discussing the construction of flux $(i_f - 1/2, j_f)$, we briefly show the implementation of scheme (4.13). In order to implement based on level and avoid changing scheme for different cells in the same level, the formal scheme (4.11) is preferable. As a result, at time step t_n, we take a conservative average of state variable n^n_e on the finer cells to get an estimate of state variable on the coarser level.

In Figure 4.15, this conservative average is given as

$$(n_e)^n_{i_c+1,j_c} = \frac{1}{r^2_{l_c,l_f}} \sum_{p=0}^{r_{l_c,l_f}-1} \sum_{q=0}^{r_{l_c,l_f}-1} (n_e)^n_{i_f+p,j_f+q}. \tag{4.14}$$

After the conservative average, the flux $F^n_{i_c+1/2,j_c}$ is modified to be constructed by state variable on level l_c, which is identical to the grid with single level. Therefore, formal scheme (4.11) can be firstly applied on one level, but modification should be done between levels to recover the scheme (4.13).

The modification from formal scheme (4.11) with conservative average (4.14) to the scheme between levels (4.13) is implemented by introducing variable of difference $\delta F_{i_c+1/2,j_c}$. When advancing the coarse level, we initialize

$$\delta F^n_{i_c+1/2,j_c} = -F^n_{i_c+1/2,j_c}, \tag{4.15}$$

where this formal $F^n_{i_c+1/2,j_c}$ is constructed from level l_c with average (4.14). After formal evolving the coarse level, the finer level is advanced, during which we add the related fine flux to $\delta F_{i_c+1/2,j_c}$ as

$$\delta F_{i_c+1/2,j_c} := \delta F_{i_c+1/2,j_c} + \frac{1}{r_{l_c,l_f}} \sum_{k=0}^{r_{l_c,l_f}-1} F^n_{i_f-1/2,j_f+k}. \tag{4.16}$$

At the end of the evolution of finer level, the solution is corrected by

$$(n_e)^{n+1}_{i_c,j_c} := (n_e)^{n+1}_{i_c,j_c} + \frac{\tau_n}{h_{l_c}} \Big(\delta F^n_{i_c+1/2,j_c} - \delta F^n_{i_c-1/2,j_c} + \delta F^n_{i_c,j_c+1/2} - \delta F^n_{i_c,j_c-1/2} \Big). \tag{4.17}$$

By the prediction formal update (4.11) and the correction in (4.11), the full scheme (4.13) is implemented.

Next we will show the scheme on the fine level, in particular to the cell (i_f, j_f) in Figure 4.15 and the flux $F^n_{i_f-1/2,j_f}$. The scheme on cell (i_f, j_f) is similar to (4.11). However, the construction of fluxes in x direction $F^n_{i_f\pm1/2,j_f}$ need the estimate value at $(i_f - 1, j_f)$. This estimation is given by the bilinear interpolation from coarser gird, and the slopes in all directions have been calculated in the scheme of coarser level. If the stencil need estimate value from a more coarser levels, the bilinear interpolation can be applied recursively on more coarser levels.

The procedure of this level-based scheme in the fixed AMR hierarchy without considering EB is summarized as follows:

1. Coarse to fine: let l iterates from 1 to $l_{\max}$. For each level l:

   i. Estimate the values on the boundary stripe of each patch by the following preferable order: domain boundary condition; values on cells in the same level l; bilinear interpolation from $l-1$ level.

   ii. Evolve every patch in level l independently by scheme (4.11).

2. Fine to coarse: let l iterates from $l_{\max}$ to 2. For each level l:

   i. Average each patch to replace the values for cells in level $l-1$.

   ii. Do correction (4.17) for those cells in $l-1$ which share edge with patch in l but not covered by G_l.

The previous procedure sketch the general idea of multi-level scheme with AMR, which is valid to those cells away from the boundary of domain. For those cells close to the boundary, the procedure should be modified to include the redistribution of (4.5). This redistribution between different levels makes the procedure more complicated. As a result, we only sketch the modification briefly in this thesis, and we would like to refer to [134] for detailed scheme and [37] for detailed implementation.

The redistribution in the AMR hierarchy cannot be taken just in each level and should be applied between two adjacent levels. Generally speaking, the redistribution should be accounted from one cell to coarse or fine cells abut this cell. Similar as the introduction of δF in (4.15) and (4.16), the redistribution among levels first redistributes the conservative error (4.5) in one level and then corrects it after the redistribution in finer level is counted. However, the redistribution is more complicated than flux correction because the redistribution could happens from coarse cells to fine cells or from the fine to the coarse, while the flux correction always modifies coarse flux by fine fluxes. If a cell (i_c, j_c) is located in level l_c and it is not covered by finer grid, the conservative error in this cell is divided into three parts to

be redistributed: the error to cells in the same level but not covered by finer grid; the error to cells in the same level and covered by finer grid; the error to cells in the next coarser level. The first part is redistributed in this level. The second part is redistributed to the cells in next finer level according to the area of finer cells. The third part should be accumulated together with the error of other cells in this level that share the same edge of cell in the coarser level. If a cell (i_c, j_c) is located in level l_c and it is covered by finer grid, the conservative error from this cell to other cells which are in level l_c but not covered by finer grid should not be redistributed, since this error will be redistributed from those finer cells.

4.3.2 AMR for elliptic equations

This subsection discusses the inclusion of AMR hierarchy into the elliptic equations with EB method in Section 4.2.2, which is based mostly on [72, 112].

Similar as the scheme for transport equations in AMR hierarchy in Section 4.3.1, the scheme for elliptic equations is designed on each level of the AMR hierarchy. On each level, the same scheme (4.8) used for EB method is applied to every cell. As can be seen later in Section 4.3.3, we follows the refinement criterion [72] for elliptic equations to refine the irregular cells together with a buffer zone. This criterion makes sure that the interpolation stencil in the irregular cells could use the values in the same level. As a result, we just need to discuss the modification of (4.8) to cells abut the coarse-fine interface and away from the vicinity of boundary.

We adopt the typical variable coefficient elliptic equation (4.7) to illustrate the scheme with AMR hierarchy. Figure 4.15 is taken as an example for cells abut the coarse-fine interface. The scheme (4.8) on coarse cell (i_c, j_c) is modified to replace the flux on the right edge with coefficient $\kappa_{i_c,j_c+1/2}$ by the sum of fluxes in the finer level. This replacement is given as

$$\kappa_{i_c,j_c+1/2}F_{i_c,j_c+1/2} \to \frac{1}{r_{l_c,l_f}} \sum_{k=0}^{r_{l_c,l_f}-1} \kappa_{i_f-1/2,j_f+k}F_{i_f-1/2,j_f+k}, \tag{4.18}$$

where the ratio $r_{l_c,l_f} = 2$ in Figure 4.15. The replacement is similar as (4.12), and it is based on the fluxes calculated in the finer level, which are $F_{i_f-1/2,j_f+k}$ for $k = 0, \cdots, r_{l_c,l_f} - 1$.

Next, we show the scheme on fine cell (i_f, j_f). It requires a quadratic interpolation from both coarse and fine level, which has broader stencil. To show this interpolation, we enlarge the scope the cells in the coarse-fine interface in Figure 4.16. To evaluate the gradient at the left edge of (i_f, j_f), which is denoted as yellow solid square in Figure 4.16, a quadratic polynomial is constructed from three values along the green dashed line in the same figure. Two points are taken from the fine grid, which is (i_f, j_f) and $(i_f, j_f + 1)$. The other point, which is denoted as blue solid circle, is taken from the quadratic interpolation from three points in coarse grid along with the red dashed line. After the constriction of quadratic polynomial along green dashed line, the gradient of this polynomial is evaluated at the yellow solid square. This gradient replaces the flux $F_{i_f-1/2,j_f}$ in for the scheme (4.8) on the finer cell (i_f, j_f).

The scheme (4.8) has been adjusted to a AMR hierarchy, and we would like to discuss the modification of the multigrid solver at the end of Section 4.2.2, to adjust the AMR hierarchy. This modification is mostly based on [112]. The main difference between the multigrid solver in AMR hierarchy and the single-level multigird solver is that, each level might only contain parts of simulation domain. As a result, we need to handle the coarse-fine interface for each level.

Following the framework in Figure 2.1, the pre-smoothing procedure is applied only to the cells on current level (level 2 in Figure 2.1), where the values in the next coarser level (level 1) is fixed. The residual r_2 is calculated only for those cells in level 2, and the restriction r_1 is applied to these cells. For the other cells in level 1 which is not coarsened from level 2, we denote them as $x_1^{(0)}$ and their residual r_1 should be calculated at this level (need the information from $x_2^{(1)}$). These two calculations (restriction from level 2 and calculating residual on level 1) fill all the residual r_1 in level 1, and then the V-cycle continues to solve $A_1 d_1 = r_1$. Before

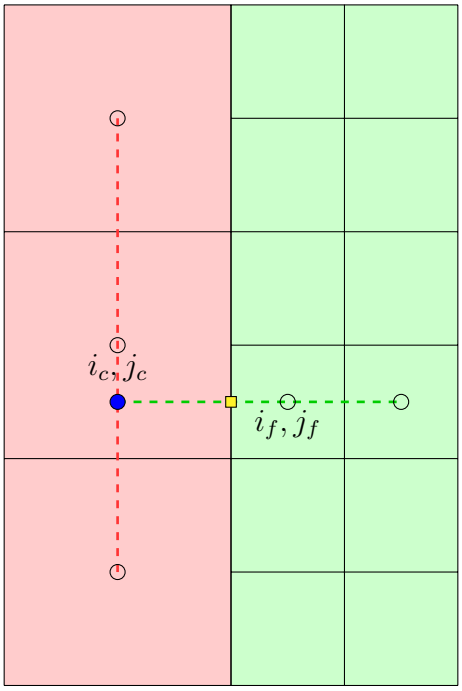


Figure 4.16: Some regular cells in two AMR levels for quadratic interpolation to construct the left flux of (i_f, j_f). Quadratic interpolation is first applied along the red dashed line to evaluate the value at blue solid circle. Then this solid value together with two values at fine level along with the green dashed line constructs the gradient at the yellow solid square.

prolongation to d_2, we correct the solution at those cell in level 1 as $x_1^{(3)} = x_1^{(0)} + d_1$. The prolongation is applied only to those cells in level 1 but covered by cells in level 2. Finally the solution $x_2^{(2)}$ in level 2 is corrected by d_2, and the post-smoothing is applied to $x_2^{(2)}$ with fixed value $x_1^{(3)}$.

4.3.3 Refinement indicators for streamer discharge

In Section 4.3.1 and Section 4.3.2, we introduced the scheme on transport equations and elliptic equations in a fixed hierarchy. In this subsection, we discuss the generation of new grid hierarchy and the refinement indicators for streamer discharge during the generation.

The new grid is generated from the old grid by mainly two steps: tagging cells

and creating patches. A full procedure could be found in [11], and the data structure for maintaining the AMR hierarchy tree could be referred to [12]. After creating a new AMR grid, interpolation and average introduced in the previous two subsections are applied to get the state variables in the new grid.

The tagging procedure starts from the finest level and iterates to the coarsest level one by one. Some refinement indicators, which will be discussed later in this subsection, are applied to all the cells on this level. If the "error" indicated by these indicators is large in some cells, we tag the cells, and they will be covered later by finer patches in the next finer level except the level number reaches prescribed maximum. Then a buffer zone is added, which tags all cells in the vicinity of previous tagged cells. The size of this vicinity is characterized by a prescribed integer number n_{bf}. A large n_{bf} brings more computational cost in evolving solution on more fine cells but decreases the frequency of generating new grid. After adding buffer zone, we tag those cells in this level if it contains a tagged cell in two finer level to keep properly nested.

The creating procedure at each level generates finer rectangle patches to covered tagged cells in current level. It takes some heuristic procedures [11] to cluster nearby cells together. A prescribed efficiency number r_{ef} will be used, which has been illustrated before Figure 4.14. After creating procedure, the new finer grid should be checked to ensure properly nested. If some patches are not, they will be further subdivided and checked repeatedly until properly nested.

After briefing two main procedures in generating new grid, we discuss the refinement indicators in this thesis for streamer discharge. Some of the following indicators has been adopted by previous researchers in [111, 116, 154]:

1. relative gradient of n_e: $\frac{\|\nabla n_e\|}{n_e} h_l \geq \varepsilon_r$ and $n_e \geq \varepsilon_e$;
2. net charge density $(n_p - n_e)$: $|(n_p - n_e)| h_l \geq \varepsilon_n$;
3. ionization coefficient $\tilde{\alpha}$: $\tilde{\alpha} h_l \geq \varepsilon_\alpha$.

The first indicator is intended to capture the shock of electron density, and we restrict the indicator to the region where n_e is larger than a given density ε_e to avoid division over small number. The second indicator could be viewed as the divergence of electric field $\vec{E}$ times mesh size and λ, and it is used to capture the thin layer ahead of the streamer. This thin layer can be clearly seen from Figure 2.8. The last indicator capture those region with high magnitude of ionization rate, which could also be viewed as capture region with higher $|\vec{E}|$ due to the positive relation of $|\vec{E}|$ and $\tilde{\alpha}$. This indicator utilizes the property that streamer propagates by the strong electric field ahead of it.

Besides the previous three indicators, we also tag all irregular cells with a buffer zone. This was discussed at the beginning of Section 4.3.2, which is used to ensure the interpolation stencil for irregular cells could use the values in same level. Moreover, this refinement gives a more accurate representation of curve boundary with affordable price since the boundary is one dimensionally lower than the dimension of domain.

4.3.4 Comparison of different indicators

In this subsection, we adopt the dimensionless model problem in 2D (4.9) to compare different indicators in Section 4.3.3. The second-order semi-implicit temporal scheme (2.12)–(2.13) and the MUSCL finite volume method with MC limiter are taken for discretization hereafter in this section. Similar as Section 4.2.3, domain Ω is taken as the $(0,1) \times (0,1)$ subtracting a round-rod pin, with a longer length 0.25 and same diameter 0.1. Most parameters are taken from (1.6), with identical $\tilde{\mu}_e$, $\tilde{\mu}_p$, $\tilde{\alpha}$ ($\tilde{\alpha} = 4332 \exp(-3.8/|\vec{E}|)$, which means $S = 4332$ and $K = 3.8$ in (4.9)) and λ. The two-dimensional dimensionless diffusion coefficient is taken as $\tilde{D}_e = \mathrm{Diag}(1.1037 \times 10^{-4}, 1.1037 \times 10^{-4})$. The dimensionless applied voltage in the boundary condition is taken as $\phi_0 = 0.5$, i.e., $\phi|_{y=1} = \phi|_{(x,y)\in\text{curve electrode}} = 0.5$ and $\phi|_{y=0} = 0$. Other boundary conditions are identical to (4.9). The initial condition is taken as $n_e(x, y, t = 0) = n_p(x, y, t = 0) = 10^{-5} + 0.1 \exp(-5000[(x - 0.5)^2 + (y -$

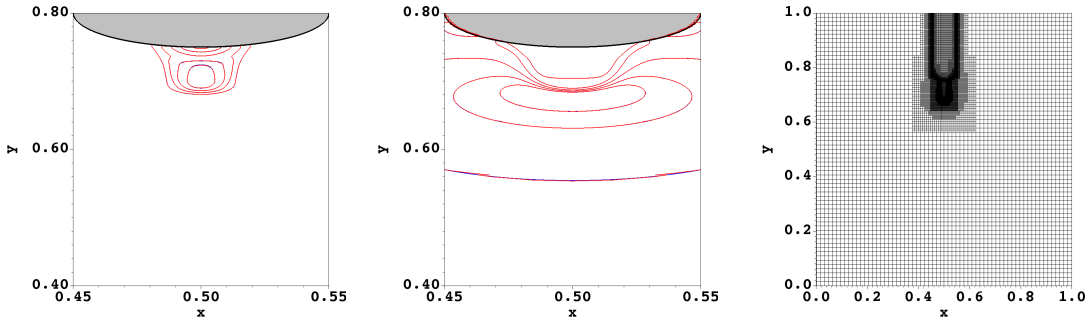
$0.75)^2]$).

We adopt a AMR hierarchy of 5 levels, where $h_l = \Delta x_l = \Delta y_l$ for all level $1 \leq l \leq 5$. h_1 is taken as $1/64$, and the refinement ratio is taken as 2 for all levels, which means finest mesh size $h_5 = 1/(64 \times 2^4) = 1/1024$. To focus on the comparison among different indicators, we fix time step as 7.5×10^{-3} ns and simulate until 9 ns, which gives dimensionless $\tau_n \approx 1.4882 \times 10^{-4}$ and $T \approx 0.17858$. This time step satisfies the CFL-type constraint (4.10) during the simulation. To construct a reference solution, we also simulate the same problem on a uniform mesh with $h = h_5 = 1/1024$. The new mesh is generated every two steps, and the buffer zone number n_{bf} is taken as 2.

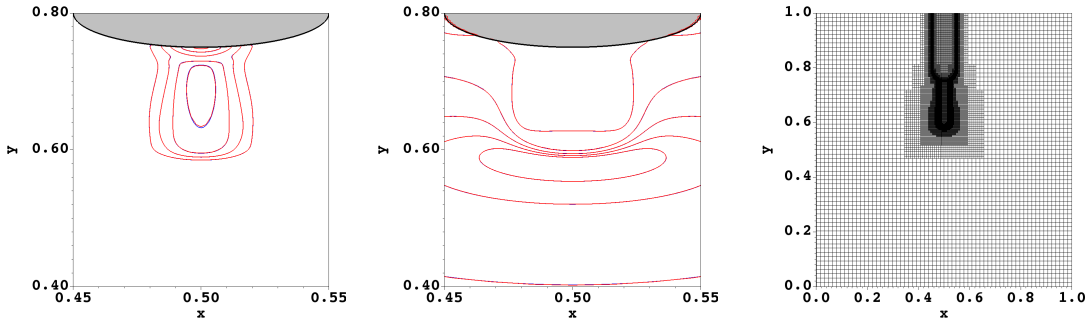
To illustrate the performance of three indicators introduced in Section 4.3.3, we simulate same problem with three indicators separately. It should be noted that the boundary of round-rod pin is always required to be refined, which means refinement is also taken on irregular cells in all levels. Therefore, the irregular cells are always refined in the discussion of three indicators. It should also be noted that this subsection mainly focuses on the comparison among three indicators and therefore, the refinement parameter of each indicator could be further optimized or selected.

We take $\varepsilon_r = 0.3$ and $\varepsilon_e = 10^{-5}$ in the first indicator. ε_e is taken from the background preionization from initial condition, and ε_r could be taken at order $O(1)$. The results of this indicator are summarized in Figure 4.17, where the number of cells on each level is indicated in the title. For a reference, we also show the results from uniform mesh in the same figure.

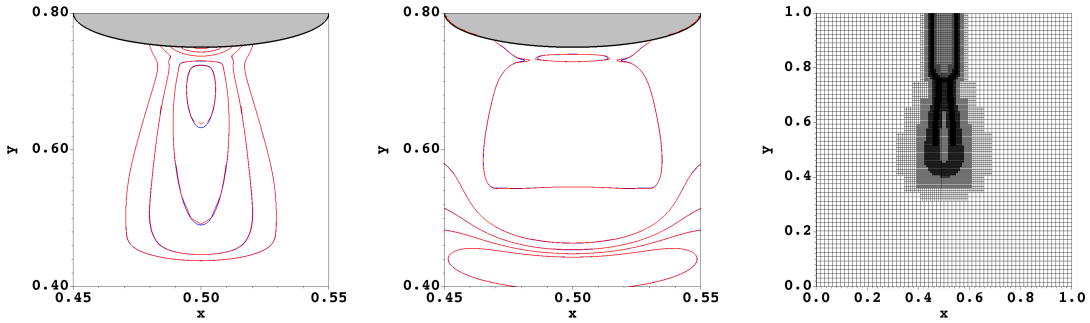
In the second indicator, we take $\varepsilon_n = 10^{-5}$, which is roughly $O(0.1\lambda)$. However, we find this indicator cannot capture the propagation of streamer properly at the initial stage. This can be attributed to the initial condition of assuming neutral plasma, which is $n_e(x, y, t = 0) = n_p(x, y, t = 0)$. At the initial stage when t is very small, $n_e(x, y, t)$ is close to $n_p(x, y, t)$. As a result, the net charge density, which is their difference, is very small in magnitude and nearly equal to 0. Then the



(a) density at $t = 3\,\text{ns}$ (b) electric field at $t = 3\,\text{ns}$ (c) AMR hierarchy at $t = 3\,\text{ns}$; cell number from level 1 to 5: 4096, 1632, 3296, 6720, 12736



(d) density at $t = 6\,\text{ns}$ (e) electric field at $t = 6\,\text{ns}$ (f) AMR hierarchy at $t = 6\,\text{ns}$; cell number from level 1 to 5: 4096, 2240, 4576, 9280, 17184



(g) density at $t = 9\,\text{ns}$ (h) electric field at $t = 9\,\text{ns}$ (i) AMR hierarchy at $t = 9\,\text{ns}$; cell number from level 1 to 5: 4096, 3008, 6752, 13600, 18944

Figure 4.17: Electron density n_e, electric field $|\vec{E}|$ and mesh hierarchy using the first indicator. Contour of n_e takes 5 values from 0.04 to 0.2 with spacing 0.04. Contours of $|\vec{E}|$ takes 5 values from 0.25 to 1.25 with spacing 0.25. The red lines denote results using AMR, while blue lines are reference results using uniform mesh.

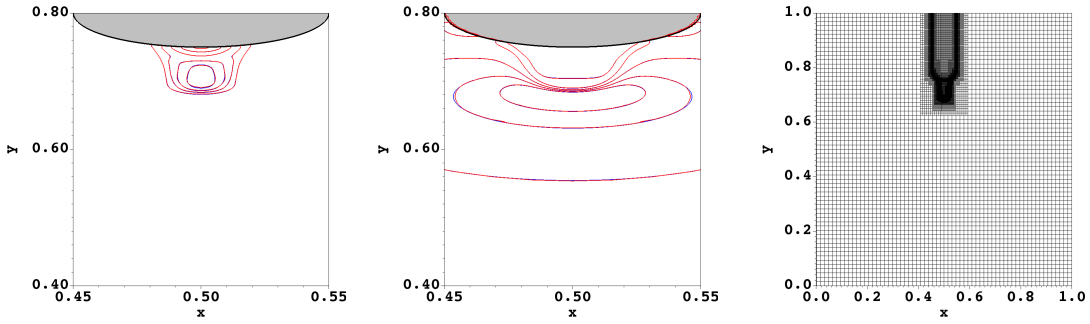
indicators will suggest nowhere to be refined initially, which could bring large error from initial stage. On the other hand, when streamer starts to propagation after a short time, the net charge density will be significant. Therefore, in this testing of the second indicator, we would like to include the first indicator with same parameter for the first 10 time steps. Starting from the 11-th time step, we close the first indicator and just use the second indicator. This initial amendment gives a more fair comparison and the first 10 steps is small compared with the total simulation steps as 1200. We depict the results of the second indicator with amendment in Figure 4.18.

We take $\varepsilon_\alpha = 0.6$ in the third indicator. This value is evaluated from plugging $|\vec{E}| = 0.8$ into the expression of $\tilde{\alpha}$ and times the result with $h_1 = 1/64$. Similar as the previous two indicators, we draw the contour of electron and electric field as well as the mesh in each level in Figure 4.19. We comment here that the contours in Figure 4.19 are significantly discontinuous in some places due to the fact that mesh is very coarse at these places.

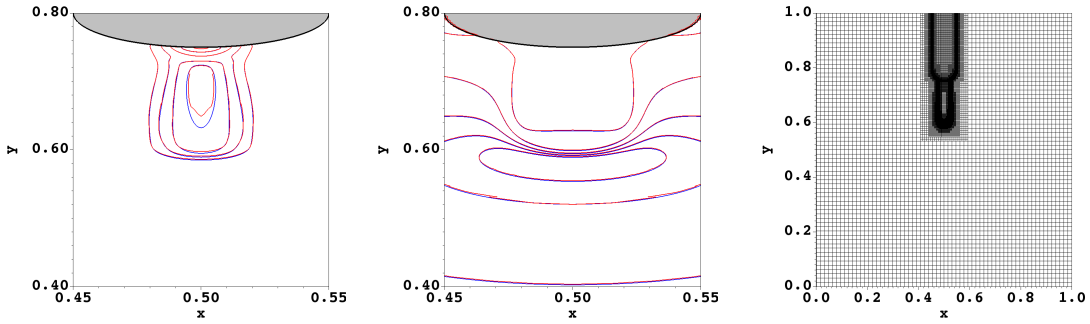
Although the number of cells in each level has been indicated at time $t = 3, 6, 9$ ns in Figures 4.17–4.19, we would like to show the average number of cells in each level during the whole simulation for a more comprehensive understanding of workload. These results are summarized in Table 4.3.

Table 4.3: Average number of cells during the simulation in each level for different refinement indicators. The average is taken over 1200 time steps.

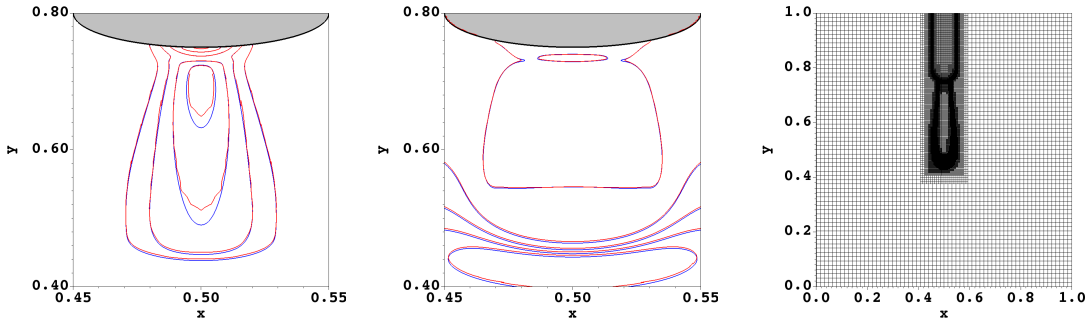
	first indicator	second indicator	third indicator
level 1, $h_1 = 1/64$	4096	4096	4096
level 2, $h_2 = 1/128$	1919.9	1338.7	1943.5
level 3, $h_3 = 1/256$	4094.7	3100.9	3551.8
level 4, $h_4 = 1/512$	8342.7	6806.3	5811.5
level 5, $h_5 = 1/1024$	14555.6	15043.6	9199.1
sum of all levels	33008.9	30385.5	24601.9



(a) density at $t = 3\,\text{ns}$ (b) electric field at $t = 3\,\text{ns}$ (c) AMR hierarchy at $t = 3\,\text{ns}$; cell number from level 1 to 5: 4096, 1152, 2464, 5280, 11648

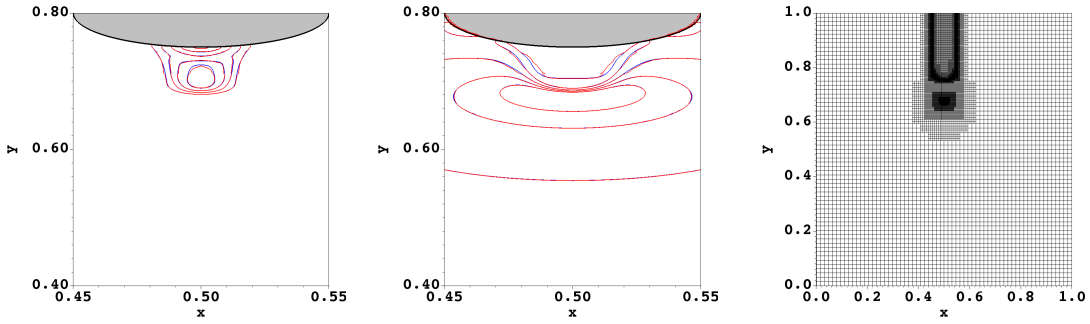


(d) density at $t = 6\,\text{ns}$ (e) electric field at $t = 6\,\text{ns}$ (f) AMR hierarchy at $t = 6\,\text{ns}$; cell number from level 1 to 5: 4096, 1440, 3392, 7456, 16416

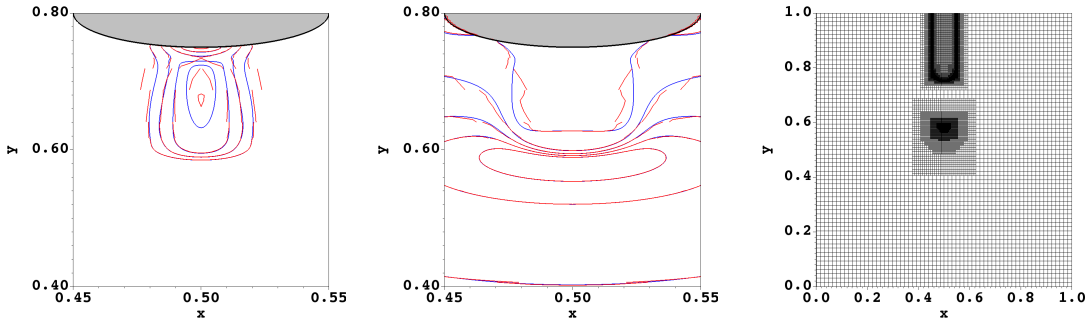


(g) density at $t = 9\,\text{ns}$ (h) electric field at $t = 9\,\text{ns}$ (i) AMR hierarchy at $t = 9\,\text{ns}$; cell number from level 1 to 5: 4096, 1920, 5248, 11504, 25808

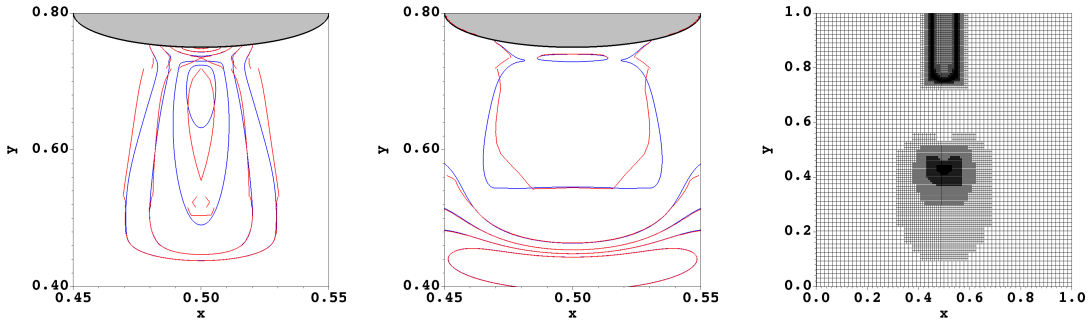
Figure 4.18: Electron density n_e, electric field $|\vec{E}|$ and mesh hierarchy using the second indicator. Contour of n_e takes 5 values from 0.04 to 0.2 with spacing 0.04. Contours of $|\vec{E}|$ takes 5 values from 0.25 to 1.25 with spacing 0.25. The red lines denote results using AMR, while blue lines are reference results using uniform mesh.



(a) density at $t = 3\,\mathrm{ns}$ (b) electric field at $t = 3\,\mathrm{ns}$ (c) AMR hierarchy at $t = 3\,\mathrm{ns}$; cell number from level 1 to 5: 4096, 1568, 3184, 5440, 8928



(d) density at $t = 6\,\mathrm{ns}$ (e) electric field at $t = 6\,\mathrm{ns}$ (f) AMR hierarchy at $t = 6\,\mathrm{ns}$; cell number from level 1 to 5: 4096, 2016, 3616, 6112, 9424



(g) density at $t = 9\,\mathrm{ns}$ (h) electric field at $t = 9\,\mathrm{ns}$ (i) AMR hierarchy at $t = 9\,\mathrm{ns}$; cell number from level 1 to 5: 4096, 3392, 5024, 7456, 9568

Figure 4.19: Electron density n_e, electric field $|\vec{E}|$ and mesh hierarchy using the third indicator. Contour of n_e takes 5 values from 0.04 to 0.2 with spacing 0.04. Contours of $|\vec{E}|$ takes 5 values from 0.25 to 1.25 with spacing 0.25. The red lines denote results using AMR, while blue lines are reference results using uniform mesh.

The comparison among three indicators in Figures 4.17–4.19 illustrates the first indicator gives closest result to the reference solution, while it uses largest number of cells as seen in Table 4.3. However, the largest number 33008.9 is small compared with 1024^2 cells in uniform mesh, and the ratio is $33008.9/1024^2 \approx 3.1\%$. On the other hand, the first indicator might be failed to capture the head of streamer properly, which could be seen in the subfigure (i) in Figure 4.17. In this subfigure, the first indicator gives boarder and coarser mesh near the head of streamer.

The second indicator is able to capture the width of streamer, but might be weaker to follow streamer in the direction of its propagation. This can be found in subfigure (a), (d) and (g) in Figure 4.18. On the other hand, the meshes in (c), (f) and (i) subfigures illustrate that this indicator could refine mesh according to the profile of streamer, and the refinement is concentrated in a narrow channel. The failure of capturing head of streamer might be attributed to the over-narrow channel in front of streamer.

The third indicator has two clear characters. First, it refines the mesh near or ahead of the tip of streamer. This character can be clearly observed in subfigures (c), (f) and (i) in Figure 4.19. Moreover, it considers a broader refinement region but concentrates the finest mesh inside a smaller region. This second character can be found in Table 4.3, where the third indicators use largest number of cell in level 2 but smallest numbers in level 4 and 5. Therefore, the third indicator could capture the density and electric field near the head of streamer, but might not be able to recover the width of streamer properly.

The three refinement indicators have different characters in simulating streamer as discussed above. In this thesis, we would like to adopt a more refined but "safer" grid. Therefore, we would like to combine three indicators above (but with other parameters) together in the applications in Section 4.4. This means the union of tagged cells in three indicators is adopted. We wish this combined indicator could refine broader region by the first indicator, capture the width of streamer by the second indicator and follow the propagation of streamer by the third indicator.

It should be commented here that for all three indicators, the time usage of tagging cells, grid generation, corresponding average and interpolation take roughly 5% of total simulation time. This percentage is acceptable in the simulation of streamer, nevertheless, one could reduce the percentage by decreasing the frequency of generating new grid.

4.4 Applications

In this section, we consider applications in 3D, where the dimensionless coefficients are taken from those after (1.6). Time step is chosen adaptively by (2.20) multiplying a safety factor 0.4, which is similar as the choice before Section 2.7.1. In the AMR method, the refinement ratio is taken as 2 for all levels, and $h_l = \Delta x_l = \Delta y_l = \Delta z_l$ for all levels. We take a buffer zone as $n_{\text{bf}} = 4$ and generate the new grid every 8 steps. The efficiency number n_{ef} is taken as 0.8. The parameters for three refinement indicators are taken as $\varepsilon_r = 0.4$, $\varepsilon_e = 10^{-5}$, $\varepsilon_n = 3 \times 10^{-5}$ and $\varepsilon_\alpha = 1$.

The implementation of EB method with AMR hierarchy in this chapter is based on the Chombo software [2, 39]. It provides a set of tools for implementing finite difference and finite volume methods for the solution of partial differential equations on block-structured adaptively refined rectangular grids. Furthermore, it supports calculations in complex geometries with embedded boundaries. As indicated in Section 2.6.2, the following simulations were performed on the cluster Tianhe2-JK located at Beijing Computational Science Research Center.

4.4.1 Effects of different shapes of anode

In Section 4.1, two typical shapes of pin or anode is introduced. We would like to compare the effects of these two shapes on the propagation of streamer in this subsection.

Two domains Ω_1 and Ω_2 are considered. Ω_1 is $(0, 1)^3$ subtracting a round-rod pin,

where the length is 0.25 and diameter is 0.2. Ω_2 is $(0,1)^3$ subtracting a hyperbolic pin, where the center of pin is $\vec{x}_c = (0.5, 0.5, 0.75)$ and semi-latus rectum is $p = 0.03$. The cross section of Ω_1 and Ω_2 at $y = 0.5$ is depicted together in Figure 4.20.

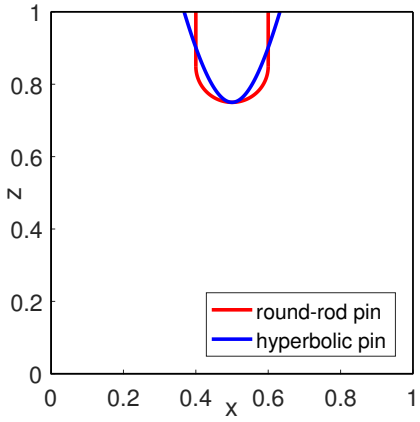


Figure 4.20: Cross section of two domains Ω_1 and Ω_2 at $y = 0.5$ in Section 4.4.1.

The dimensionless applied voltage ϕ_0 is taken as 0.3. The function used in initial condition is taken as

$$n_0(\vec{x}) = 10^{-4} + 0.1 \exp\left(-\frac{(x-0.5)^2 + (y-0.5)^2 + (z-0.75)^2}{2 \times 0.01^2}\right). \tag{4.19}$$

This initial value is used in both two simulations in Ω_1 and Ω_2.

In the AMR hierarchy, we take 4 levels, with coarsest mesh size $h_1 = 1/128$. The two simulations in this subsection are run on only 40 cores (2 nodes), and each simulation (including data output) takes less than two days to 9 ns. Results of two simulations at three different times, which are $3, 6, 9$ ns, are depicted and compared in Figures 4.21–4.23.

Generally speaking, the shape of anode could influence the propagation of streamer greatly in its speed, width and shape. It could be clearly seen in Figures 4.21–4.23 that the streamer in Ω_1, which is the domain with round-rod anode, propagates slower than the streamer in Ω_2 with hyperbolic anode. To see the width of streamer, we could concentrate on Figure 4.22. In this figure, the streamer in Ω_1 at 9 ns propagates around $z = 0.6$, which is the similar as the location of streamer tip in Ω_2 at

6 ns. Comparing the electron density of these two streamers (at two different times), the one in Ω_1 has thinner channel but higher electron density.

The shape of anode could also affect the magnitude and distribution of electric field, which could be found in Figure 4.23. Initially at 3 ns, the electric field ahead of streamer in Ω_1 is weaker than that ahead of streamer in Ω_2. However, at 9 ns, the electric field in Ω_1 becomes stronger in turn. On the other hand, we could compare the distribution of electric field between two streamers which propagate at similar location $z = 0.6$, i.e., the streamer in Ω_1 at 9 ns and the streamer in Ω_2 at 6 ns. The comparison shows electric field in Ω_1 is more concentrated near the head of streamer, while the electric field in Ω_2 has a longer and wider tail along with the streamer. The slower propagation of streamer in Ω_1 might be attributed to the weaker initial electric field, and it is possible that the larger channel of streamer in Ω_2 is due to the wider tail of electric field.

4.4.2 Streamer discharge for interacting streamers

The interaction of streamers in a domain with round-rod pin is studied in this subsection. We consider a domain $\Omega = (0, 0.2)^3$ subtracting a round-rod pin with length 0.1 and diameter 0.04. This domain is to mimic the hyperbolic pin used in [137] but changing the shape of anode from hyperbolic to round-rod. The applied voltage is taken as 2 kV, which means dimensionless $\phi_0 \approx 0.0385$. The initial condition is taken as a summation of four Gaussian plasmas with a homogeneous background as

$$n_0(\vec{x}) = 10^{-3} + \sum_{i=1}^{4} 0.1 \exp\left(-\frac{(x - 0.1 + \delta x_i)^2 + (y - 0.1)^2 + (z - 0.08)^2}{\sigma^2}\right), \qquad (4.20)$$

where $\sigma = 0.0025$, $\delta x_1 = 0.007$, $\delta x_2 = -0.007$, $\delta x_3 = 0.021$ and $\delta x_4 = -0.021$. This domain together with the initial condition is depicted in Figure 4.24. It should be noted that the applied voltage as well as the initial condition is identical to one example used in [137], and we change the anode to be a round-rod pin. In this application, the AMR hierarchy takes 4 levels where h_1 is taken as 1/64.

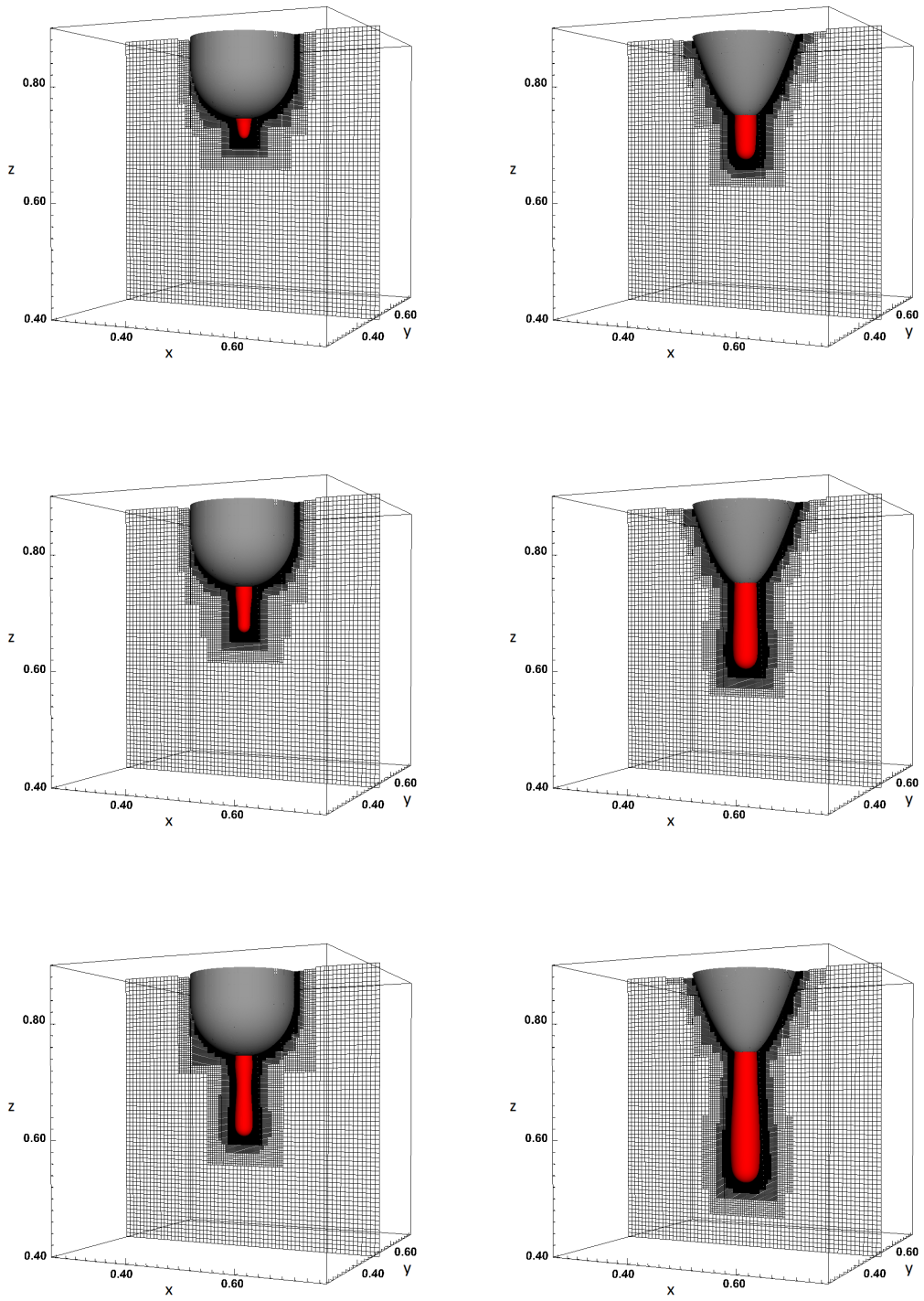


Figure 4.21: Contour surfaces of Electron density $n_e = 0.05$ and slices of mesh hierarchy on plane $y = 0.5$ at three different times $t = 3, 6, 9$ (ns). (ordered from top line to bottom line). Left subfigures: round-rod anode; right subfigures: hyperbolic anode.

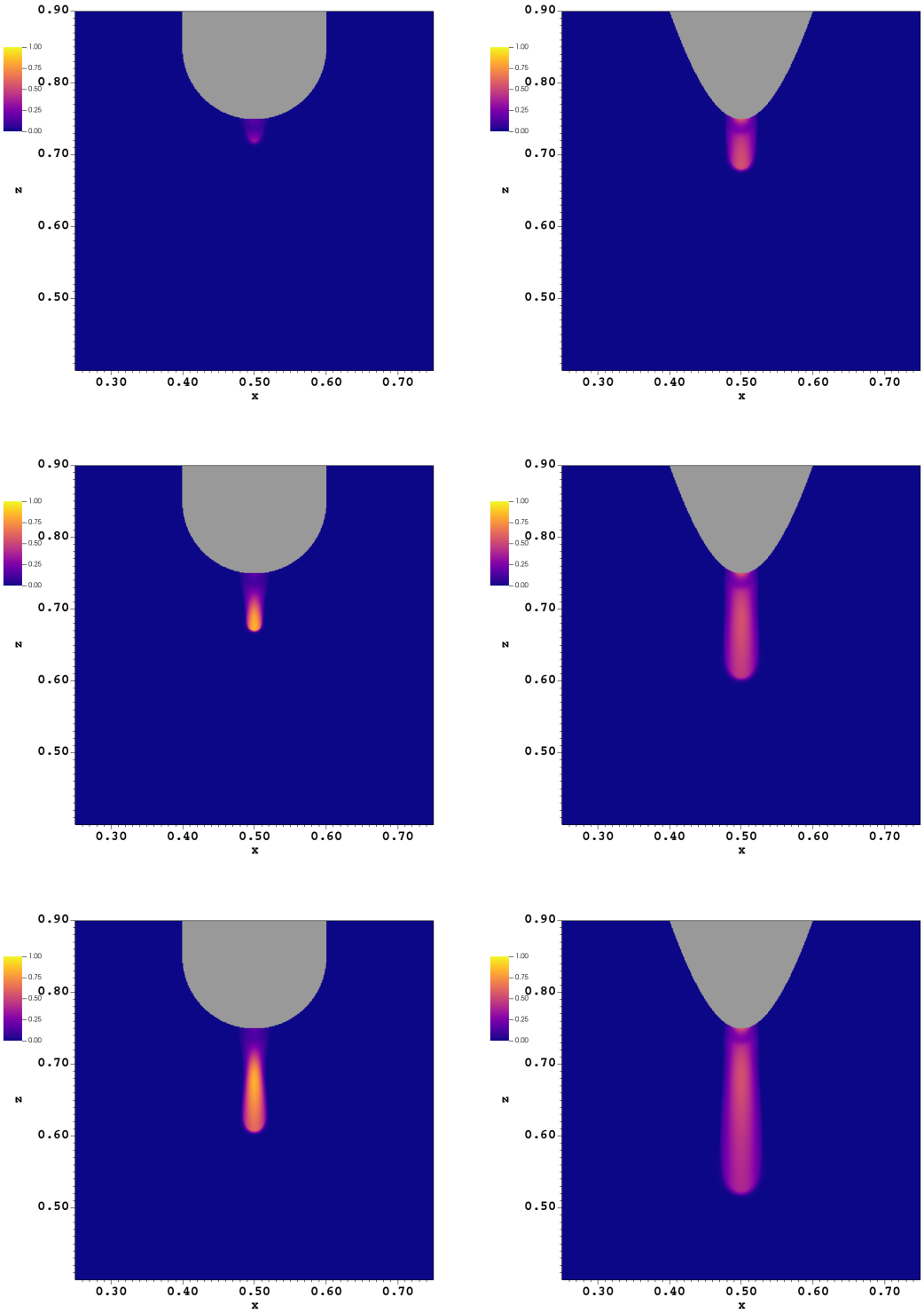


Figure 4.22: Cross section of electron density n_e on plane $y = 0.5$ at three different times $t = 3, 6, 9$ (ns). (ordered from top line to bottom line). Left subfigures: round-rod anode; right subfigures: hyperbolic anode.

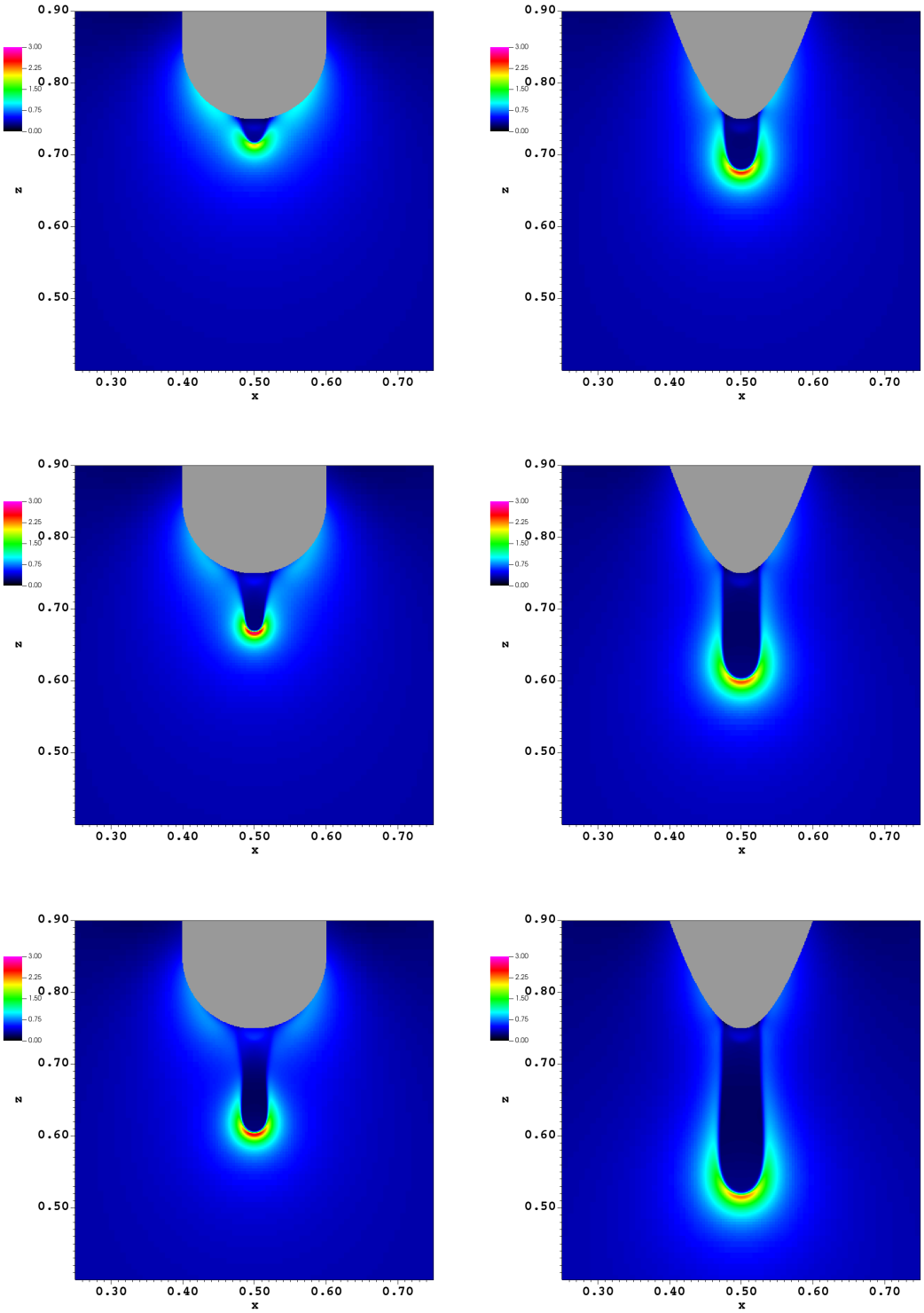


Figure 4.23: Cross section of electric field $|\vec{E}|$ on plane $y = 0.5$ at three different times $t = 3, 6, 9$ (ns). (ordered from top line to bottom line). Left subfigures: round-rod anode; right subfigures: hyperbolic anode.

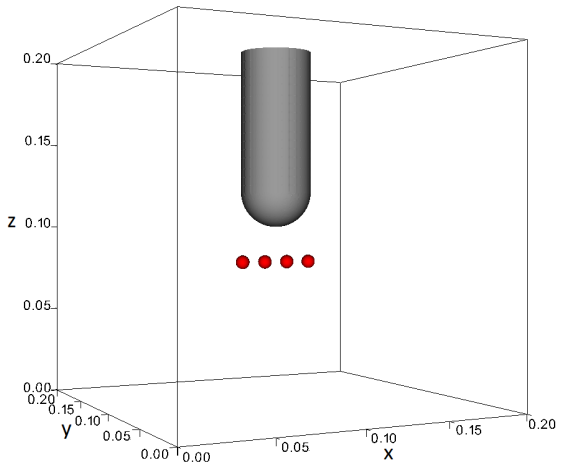


Figure 4.24: Simulation domain and initial condition. The round-rod pin is dyed as gray color, and contour surface of electron density 0.01 is dyed as red color.

The contours of electron density for $n_e = 0.002$ as well as the grid hierarchy are depicted in Figures 4.25 and 4.26. It can be seen that four streamers are attracted by the round-rod anode and merged near the tip of the anode. The channels of streamers would become thinner after reaching the anode. On the other hand, the refinement indicator gives a fine mesh near the heads of streamers and the tip of anode. The mesh is coarsen near the middle part of streamers when the channels do not change significantly.

In order to see more detailed interaction of streamers in the cross section $y = 0.1$, we draw the electron density at this cross section in Figure 4.27 and the magnitude of electric field in Figure 4.28. It could be clearly seen in Figure 4.27 that the electron propagates to the anode, and the direction of this propagation seems to be perpendicular to the anode. After touching the anode, the width of channels of streamers and the magnitude of electrons decrease. On the other hand, the streamers seems difficult to propagate downwards. This could be attributed to the weaker electric field located at the bottom heads of streamers compared with stronger field at the top heads of streamers, which can be seen in Figure 4.28.

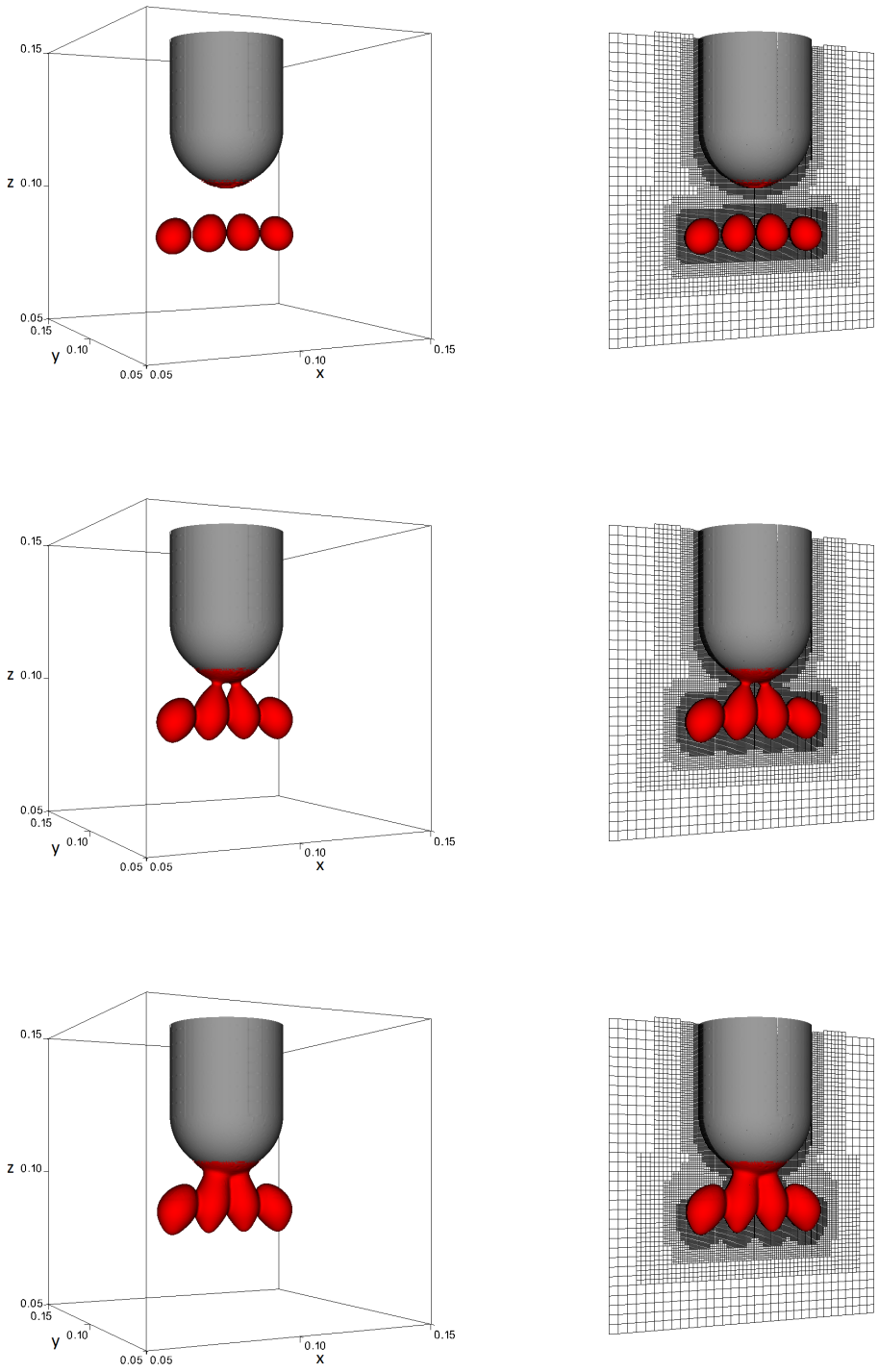


Figure 4.25: Left subfigures: the contour surfaces (red color) of electron density $n_e = 0.002$ at three different times $t = 0.37$, 0.77, 0.97 (ns). Right subfigures: slices of mesh at $y = 0.1$ of the corresponding left subfigures.

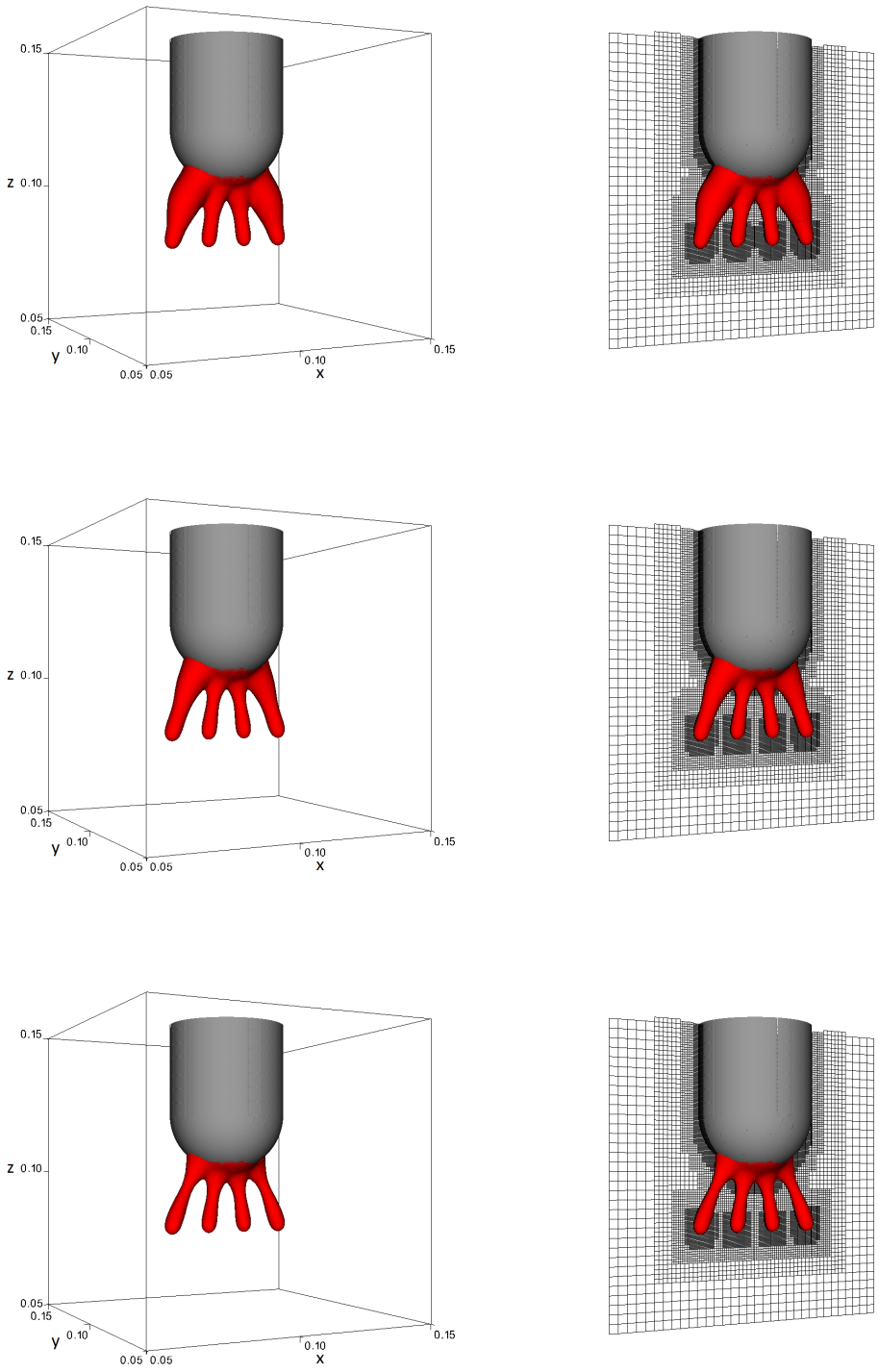


Figure 4.26: Left subfigures: the contour surfaces (red color) of electron density $n_e = 0.002$ at three different times $t = 2.40$, 3.02 and 3.83 (ns). Right subfigures: slices of mesh at $y = 0.1$ of the corresponding left subfigures.

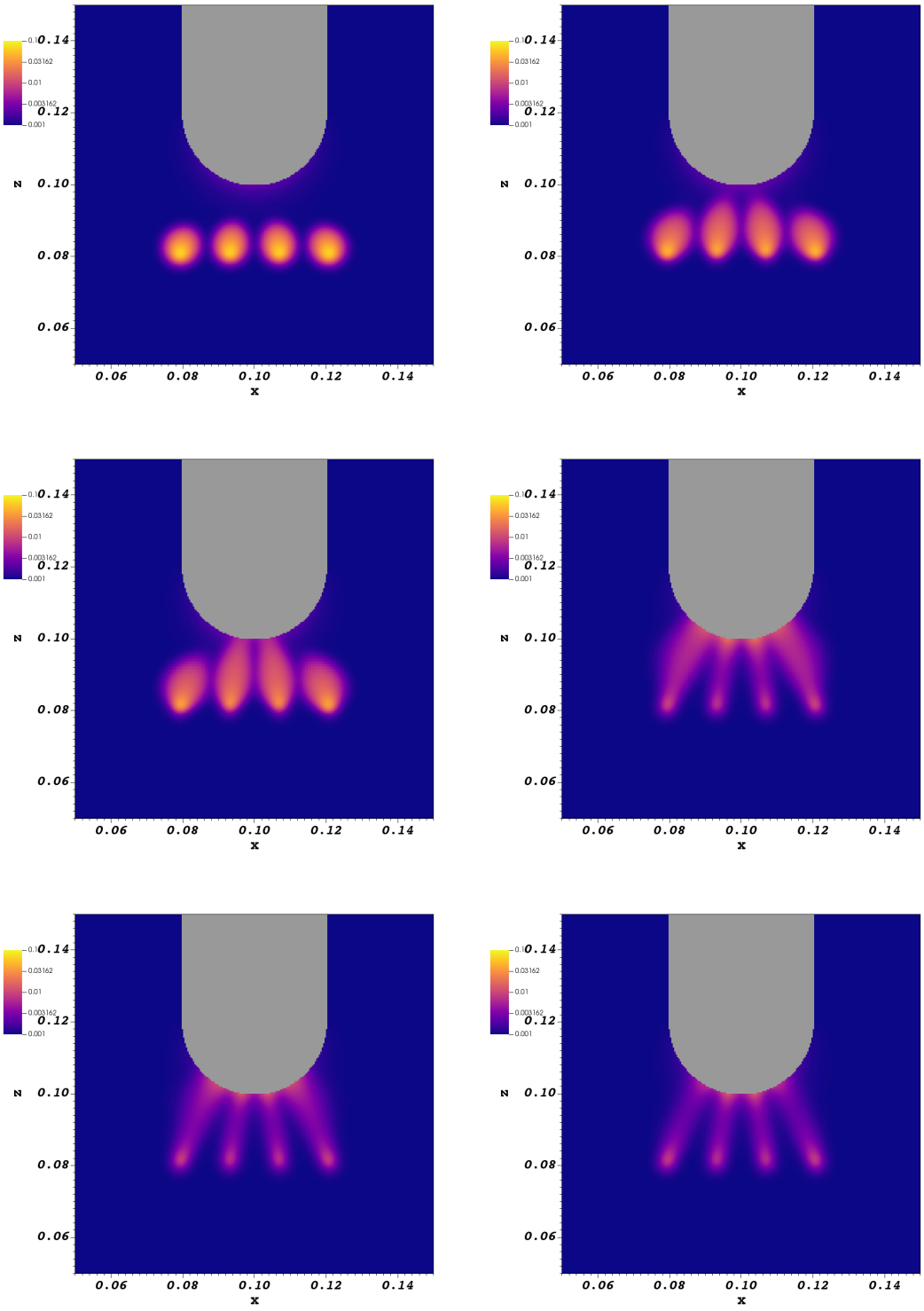


Figure 4.27: Cross section of electron density n_e at $y = 0.1$. Six subfigures are depicted at time $t = 0.37$, 0.77, 0.97, 2.40, 3.02 and 3.83 (ns), ordered from left to right and top to bottom.

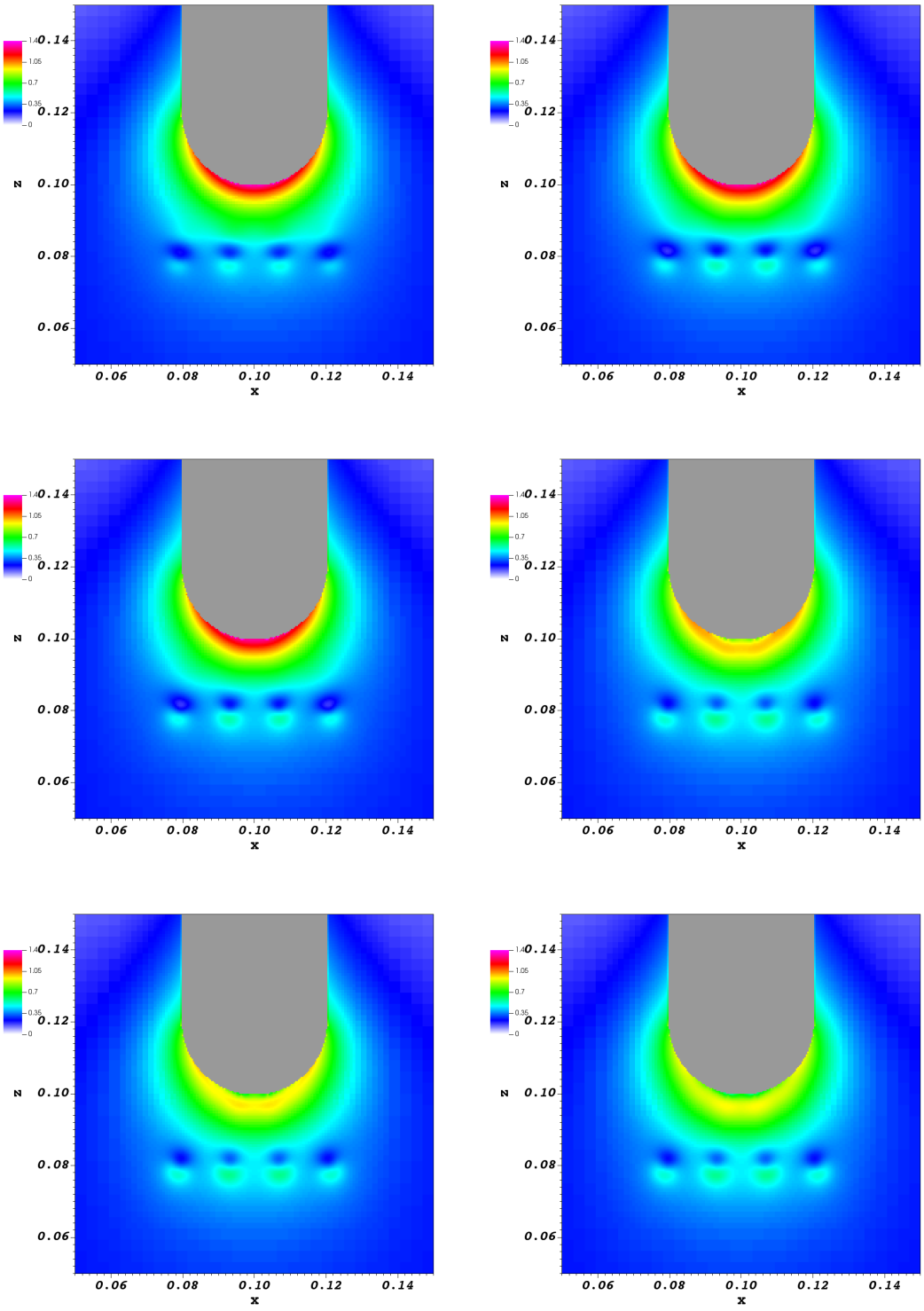


Figure 4.28: Cross section of magnitude of electric field $|\vec{E}|$ at $y = 0.1$. Six subfigures are depicted at time $t = 0.37$, 0.77, 0.97, 2.40, 3.02 and 3.83 (ns), ordered from left to right and top to bottom.

Chapter 5

Conclusions and Future Work

This thesis focuses on efficient and accurate simulations for streamer discharge in three dimensions based on the fluid model. Different numerical methods are designed, compared and applied to the fluid model. With these methods, we simulate the propagation of streamer by parallel computing. The main work in this thesis is summarized as follows:

1. Propose a new second-order semi-implicit scheme for the fluid model. To relax the dielectric relaxation time constraint and achieve a higher order convergence, a new second-order semi-implicit scheme is designed. It can be solved explicitly and requires solving elliptic equation only once at each time step. The resulting variable-coefficient elliptic equation is solved by multigrid preconditioned FGMRES method efficiently. These methods are implemented using MPI parallelism, and the parallel efficiency is satisfactory. The propagation of streamer between two flat electrodes is simulated, and the numerical performance of different temporal schemes and algebraic solvers are studied as well.

2. Evaluate photoionization in the classical integral model by the kernel-independent fast multipole method. Photoionization is important in streamer discharges, and the classical modeling of it is an integral model. The kernel-independent fast multipole method is introduced to evaluate this integral. The

accuracy of this evaluation is greatly improved compared with two other popular PDE-based methods. Quantitative comparison is studied among different domains and various pressures to demonstrate the robustness of different methods. Besides accuracy, computational cost and scalability of the fast multipole method are studied numerically.

3. Apply the embedded boundary and adaptive mesh refinement methods to streamer discharge inside general domains. Streamer can be observed in general electrodes, therefore, the simulator should be adapted to general domains. The embedded boundary method is applied to the handle general domain boundary using interpolation on a background Cartesian grid, and its convergence and stability are numerically studied. Adaptive mesh refinement method with some indicators is adopted to capture the structure of streamer and improve the efficiency of simulation. Numerical results of streamer propagation in different anodes and the interaction of streamers are reported.

Apart from the accuracy, the developed simulator possesses satisfactory performance with additional three main advantages: efficient in both fluid equations and photoionization source term; applicable to domain with complex geometry; parallelly efficient in cluster. These advantages suggest that the proposed simulator can be applied to broader research studies including high-voltage engineering, sprite discharge simulations, and discharge applications. However, there are three limitations in this study. Firstly, the simulator was based on deterministic fluid model, which was difficult to predict the branching of streamer. The second limitation is that the convergence was second order, and the embedded boundary method required more effort to achieve higher order. Thirdly, the electric field is found not so satisfactory near the embedded boundary though potential is well calculated. As a result, there are several possible future works:

- Apply the proposed simulator to some realistic setting and compare the numerical results with experiments.

- Do some analysis on the convergence, stability and efficiency of the studied numerical schemes.
- Extend the simulator to more sophisticated fluid models which include more particles, field-dependent coefficients and other types of boundary conditions.
- Model the stochastic effect in streamer discharge properly.
- Further improve the parallel efficiency of proposed simulator, in particular the load balancing in a AMR hierarchy.
- Consider the skill of mixed finite element, which calculates both electric field and potential in high-order convergence.
- Tolerate much weaker stability condition by the implicit finite volume method.

Bibliography

[1] M. Adams, M. Brezina, J. Hu, and R. Tuminaro, *Parallel multigrid smoothing: polynomial versus gauss–seidel*, Journal of Computational Physics, 188 (2003), pp. 593–610.

[2] M. Adams, P. Colella, D. T. Graves, J. Johnson, N. Keen, T. J. Ligocki, D. F. Martin, P. McCorquodale, D. Modiano, P. Schwartz, T. Sternberg, and B. V. Straalen, *EBAMRTools: EBChombo's adaptive refinement library*, tech. report, Lawrence Berkeley National Laboratory, Berkeley, CA, 2019.

[3] M. Aints, A. Haljaste, and L. Roots, *Photoionizing radiation of positive corona in moist air*, in Proc. 8th Int. Symp. on High Pressure Low Temperature Plasma Chemistry (Estonia, 21–25 July 2002), 2002.

[4] A. Ali, *The electron avalanche ionization of air and a simple air chemistry model*, tech. report, NAVAL RESEARCH LABORATORY Washington, D.C., 1982.

[5] N. Y. Babaeva, A. N. Bhoj, and M. J. Kushner, *Streamer dynamics in gases containing dust particles*, Plasma Sources Science and Technology, 15 (2006), p. 591.

[6] B. Bagheri and J. Teunissen, *The effect of the stochasticity of photoionization on 3d streamer simulations*, Plasma Sources Science and Technology, 28 (2019), p. 045013.

[7] B. Bagheri, J. Teunissen, U. Ebert, M. M. Becker, S. Chen, O. Ducasse, O. Eichwald, D. Loffhagen, A. Luque, D. Mihailova, J. M. Plewa, J. van Dijk, and M. Yousfi, *Comparison of six simulation codes for positive streamers in air*, Plasma Sources Science and Technology, 27 (2018), p. 095002.

[8] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, R. T. Mills, T. Munson, K. Rupp, P. Sanan, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang, *PETSc Web page*. http://www.mcs.anl.gov/petsc, 2018.

[9] M. S. Barnes, T. J. Cotler, and M. E. Elta, *Large-signal time-domain modeling of low-pressure rf glow discharges*, Journal of Applied Physics, 61 (1987), pp. 81–89.

[10] F. Benkhaldoun, J. Fořt, K. Hassouni, J. Karel, G. Scarella, and D. Trdlička, *A full 3-d dynamically adaptive unstructured grid finite-volume approach to simulate multiple branching in streamer propagation*, IEEE Transactions on Plasma Science, 42 (2014), pp. 2420–2421.

[11] M. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, Journal of Computational Physics, 82 (1989), pp. 64–84.

[12] M. J. Berger, *Data structures for adaptive grid generation*, SIAM Journal on Scientific and Statistical Computing, 7 (1986), pp. 904–916.

[13] M. J. Berger and J. Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, Journal of Computational Physics, 53 (1984), pp. 484–512.

[14] D. Bessières, J. Paillol, A. Bourdon, P. Ségur, and E. Marode, *A new one-dimensional moving mesh method applied to the simulation of streamer discharges*, Journal of Physics D: Applied Physics, 40 (2007), p. 6559.

[15] C. K. Birdsall, *Particle-in-cell charged-particle simulations, plus monte carlo collisions with neutral atoms, pic-mcc*, IEEE Transactions on Plasma Science, 19 (1991), pp. 65–85.

[16] L. D. Boggs, N. Liu, M. Splitt, S. Lazarus, C. Glenn, H. Rassoul, and S. A. Cummer, *An analysis of five negative sprite-parent discharges and their associated thunderstorm charge structures*, Journal of Geophysical Research: Atmospheres, 121 (2016), pp. 759–784.

[17] J. P. Boris and D. L. Book, *Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works*, Journal of Computational Physics, 11 (1973), pp. 38–69.

[18] A. Bourdon, V. P. Pasko, N. Y. Liu, S. Célestin, P. Ségur, and E. Marode, *Efficient models for photoionization produced by non-thermal gas discharges in air based on radiative transfer and the helmholtz equations*, Plasma Sources Science and Technology, 16 (2007), pp. 656–678.

[19] J. Brannick, X. Hu, C. Rodrigo, and L. Zikatanov, *Local fourier analysis of multigrid methods with polynomial smoothers and aggressive coarsening*, Numerical Mathematics: Theory, Methods and Applications, 8 (2015), pp. 1–21.

[20] T. M. P. Briels, E. M. van Veldhuizen, and U. Ebert, *Positive streamers in air and nitrogen of varying density: experiments on similarity laws*, Journal of Physics D: Applied Physics, 41 (2008), p. 234008.

[21] O. Buneman and D. Dunn, *Computer experiments in plasma physics*, tech. report, Institute of Plasma Research, Stanford University, Stanford, California, 1965.

[22] P. Burger, *Elastic collisions in simulating one-dimensional plasma diodes on the computer*, The Physics of Fluids, 10 (1967), pp. 658–666.

[23] J. Capeillère, P. Ségur, A. Bourdon, S. Célestin, and S. Pancheshnyi, *The finite volume method solution of the radiative transfer equation for photon transport in non-thermal gas discharges: application to the calculation of photoionization in streamer discharges*, Journal of Physics D: Applied Physics, 41 (2008), p. 234018.

[24] S. Celestin, *Study of the dynamics of streamers in air at atmospheric pressure*, theses, Ecole Centrale Paris, Dec. 2008.

[25] O. Chanrion and T. Neubert, *A pic-mcc code for simulation of streamer propagation in air*, Journal of Computational Physics, 227 (2008), pp. 7222–7245.

[26] T. Charoy, J. P. Boeuf, A. Bourdon, J. A. Carlsson, P. Chabert, B. Cuenot, D. Eremin, L. Garrigues, K. Hara, I. D. Kaganovich, A. T. Powis, A. Smolyakov, D. Sydorenko, A. Tavant, O. Vermorel, and W. Villafana, *2d axial-azimuthal particle-in-cell benchmark for low-temperature partially magnetized plasmas*, Plasma Sources Science and Technology, 28 (2019), p. 105010.

[27] S. Chen, F. Wang, Q. Sun, and R. Zeng, *Branching characteristics of positive streamers in nitrogen-oxygen gas mixtures*, IEEE Transactions on Dielectrics and Electrical Insulation, 25 (2018), pp. 1128–1134.

[28] S. Chen, R. Zeng, and C. Zhuang, *The diameters of long positive streamers in atmospheric air under lightning impulse voltage*, Journal of Physics D: Applied Physics, 46 (2013), p. 375203.

[29] S. Chen, R. Zeng, C. Zhuang, X. Zhou, and Y. Ding, *Experimental study on branch and diffuse type of streamers in leader restrike of long air gap discharge*, Plasma Science and Technology, 18 (2016), pp. 305–310.

[30] H. Cheng, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm in three dimensions*, Journal of computational physics, 155 (1999), pp. 468–498.

[31] I.-L. Chern and P. Colella, *A conservative front tracking method for hyperbolic conservation laws*, LLNL Rep. No. UCRL-97200, Lawrence Livermore National Laboratory, (1987).

[32] D. K. Clarke, M. D. Salas, and H. A. Hassan, *Euler calculations for multielement airfoils using cartesian grids*, AIAA Journal, 24 (1986), pp. 353–358.

[33] B. Cockburn and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework*, Mathematics of Computation, 52 (1989), pp. 411–435.

[34] ——, *The Runge–Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems*, Journal of Computational Physics, 141 (1998), pp. 199–224.

[35] P. Colella, *Multidimensional upwind methods for hyperbolic conservation laws*, Journal of Computational Physics, 87 (1990), pp. 171–200.

[36] P. Colella, M. R. Dorr, and D. D. Wake, *A conservative finite difference method for the numerical solution of plasma fluid equations*, Journal of Computational Physics, 149 (1999), pp. 168–193.

[37] P. Colella, D. Graves, T. Ligocki, and D. Modiano, *Ebamrtools: Ebchombo's adaptive refinement library*, tech. report, Lawrence Berkeley National Laboratory, Berkeley, CA, 2001.

[38] P. Colella, D. T. Graves, B. J. Keen, and D. Modiano, *A cartesian grid embedded boundary method for hyperbolic conservation laws*, Journal of Computational Physics, 211 (2006), pp. 347–366.

[39] P. Colella, D. T. Graves, T. J. Ligocki, G. Miller, D. Modiano, P. Schwartz, B. V. Straalen, J. Pillod, D. Trebotich, M. Barad, B. Keen, A. Nonaka, and C. Shen, *EBChombo software package for cartesian grid, embedded boundary applications*, tech. report, Lawrence Berkeley National Laboratory, Berkeley, CA, 2000.

[40] Y. Cui, C. Zhuang, X. Zhou, and R. Zeng, *The dynamic expansion of leader discharge channels under positive voltage impulse with different rise times in long air gap: Experimental observation and simulation results*, Journal of Applied Physics, 125 (2019), p. 113302.

[41] S. A. Cummer, N. Jaugey, J. Li, W. A. Lyons, T. E. Nelson, and E. A. Gerken, *Submillisecond imaging of sprite development and structure*, Geophysical Research Letters, 33 (2006).

[42] S. A. Cummer, J. Li, F. Han, G. Lu, N. Jaugey, W. A. Lyons, and T. E. Nelson, *Quantification of the troposphere-to-ionosphere charge transfer in a gigantic jet*, Nature Geoscience, 2 (2009), pp. 617–620.

[43] C. L. da Silva and V. P. Pasko, *Dynamics of streamer-to-leader transition at reduced air densities and its implications for propagation of lightning leaders and gigantic jets*, Journal of Geophysical Research: Atmospheres, 118 (2013), pp. 13,561–13,590.

[44] A. J. Davies, C. S. Davies, and C. J. Evans, *Computer simulation of rapidly developing gaseous discharges*, Proceedings of the Institution of Electrical Engineers, 118 (1971), pp. 816–823.

[45] D. DeZeeuw and K. G. Powell, *An adaptively refined cartesian mesh solver for the euler equations*, Journal of Computational Physics, 104 (1993), pp. 56–68.

[46] S. K. Dhali and P. F. Williams, *Two-dimensional studies of streamers in gases*, Journal of Applied Physics, 62 (1987), pp. 4696–4707.

[47] Q. Du and D. Wang, *Tetrahedral mesh generation and optimization based on centroidal voronoi tessellations*, International Journal for Numerical Methods in Engineering, 56 (2003), pp. 1355–1373.

[48] M. Duarte, Z. Bonaventura, M. Massot, and A. Bourdon, *A numerical strategy to discretize and solve the poisson equation on dynamically adapted multiresolution grids for time-dependent streamer discharge simulations*, Journal of Computational Physics, 289 (2015), pp. 129–148.

[49] A. Dubinova, D. Trienekens, U. Ebert, S. Nijdam, and T. Christen, *Pulsed positive discharges in air at moderate pressures near a dielectric rod*, Plasma Sources Science and Technology, 25 (2016), p. 055021.

[50] O. Ducasse, L. Papageorghiou, O. Eichwald, N. Spyrou, and M. Yousfi, *Critical analysis on two-dimensional point-to-plane streamer simulations using the finite element and finite volume methods*, IEEE Transactions on Plasma Science, 35 (2007), pp. 1287–1300.

[51] U. Ebert, C. Montijn, T. M. Briels, W. Hundsdorfer, B. Meulenbroek, A. Rocco, and E. M. van Veldhuizen, *The multiscale nature of streamers*, Plasma Sources Science and Technology, 15 (2006), p. S118.

[52] U. Ebert, S. Nijdam, C. Li, A. Luque, T. Briels, and E. van Veldhuizen, *Review of recent results on streamer discharges and discussion of their relevance for sprites and lightning*, Journal of Geophysical Research: Space Physics, 115 (2010).

[53] U. Ebert and D. D. Sentman, *Streamers, sprites, leaders, lightning: from micro- to macroscales*, Journal of Physics D: Applied Physics, 41 (2008), p. 230301.

[54] P. R. Eiseman, *Grid generation for fluid mechanics computations*, Annual Review of Fluid Mechanics, 17 (1985), pp. 487–522.

[55] F. Filbet and S. Jin, *A class of asymptotic-preserving schemes for kinetic equations and related problems with stiff sources*, Journal of Computational Physics, 229 (2010), pp. 7625–7648.

[56] G. E. Georghiou, R. Morrow, and A. C. Metaxas, *A two-dimensional, finite-element, flux-corrected transport algorithm for the solution of gas discharge problems*, Journal of Physics D: Applied Physics, 33 (2000), p. 2453.

[57] C. Geuzaine and J.-F. Remacle, *Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities*, International Journal for Numerical Methods in Engineering, 79 (2009), pp. 1309–1331.

[58] J. Goodman, *The formation of thin polymer films in the gas discharge*, Journal of Polymer Science, 44 (1960), pp. 551–552.

[59] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, Journal of computational physics, 73 (1987), pp. 325–348.

[60] L. F. Greengard and J. Huang, *A new version of the fast multipole method for screened coulomb interactions in three dimensions*, Journal of Computational Physics, 180 (2002), pp. 642–658.

[61] N. A. Gumerov and R. Duraiswami, *Fast multipole method for the biharmonic equation in three dimensions*, Journal of Computational Physics, 215 (2006), pp. 363–383.

[62] F. H. Harlow, *The particle-in-cell computing method for fluid dynamics*, Methods Comput. Phys., 3 (1964), pp. 319–343.

[63] A. Harten, *High resolution schemes for hyperbolic conservation laws*, Journal of computational physics, 49 (1983), pp. 357–393.

[64] F. Hecht, *New development in FreeFem++*, J. Numer. Math., 20 (2012), pp. 251–265.

[65] R. W. Hockney and J. W. Eastwood, *Computer simulation using particles*, Taylor & Francis Group, 1988.

[66] T. Huiskamp, W. Sengers, F. J. C. M. Beckers, S. Nijdam, U. Ebert, E. J. M. van Heesch, and A. J. M. Pemen, *Spatiotemporally resolved imaging of streamer discharges in air generated in a wire-cylinder reactor with (sub)nanosecond voltage pulses*, Plasma Sources Science and Technology, 26 (2017), p. 075009.

[67] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2 ed., 2008.

[68] B. Jacob, S. Ng, and D. Wang, *Memory systems: cache, DRAM, disk*, Morgan Kaufmann, 2010.

[69] A. Jaworek, A. Krupa, and T. Czech, *Modern electrostatic devices and methods for exhaust gas cleaning: A brief review*, Journal of Electrostatics, 65 (2007), pp. 133–155.

[70] M. Jiang, Y. LI, H. Wang, W. Ding, and C. Liu, *3d PIC-MCC simulation of corona discharge in needle-plate electrode with external circuit*, Plasma Sources Science and Technology, 29 (2020), p. 015020.

[71] M. Jiang, Y. Li, H. Wang, P. Zhong, and C. Liu, *A photoionization model considering lifetime of high excited states of n2 for pic-mcc simulations of positive streamers in air*, Physics of Plasmas, 25 (2018), p. 012127.

[72] H. Johansen and P. Colella, *A cartesian grid embedded boundary method for poisson's equation on irregular domains*, Journal of Computational Physics, 147 (1998), pp. 60–85.

[73] R. P. Joshi and S. M. Thagard, *Streamer-like electrical discharges in water: Part ii. environmental applications*, Plasma Chemistry and Plasma Processing, 33 (2013), pp. 17–49.

[74] M. D. Judd, Li Yang, and I. B. B. Hunter, *Partial discharge monitoring of power transformers using uhf sensors. part i: sensors and signal interpretation*, IEEE Electrical Insulation Magazine, 21 (2005), pp. 5–14.

[75] S. Kacem, O. Eichwald, O. Ducasse, N. Renon, M. Yousfi, and K. Charrada, *Full multi grid method for electric field computation in point-to-plane streamer discharge in air at atmospheric pressure*, Journal of Computational Physics, 231 (2012), pp. 251–261.

[76] L. E. Kline, *Calculations of discharge initiation in overvolted parallel-plane gaps*, Journal of Applied Physics, 45 (1974), pp. 2046–2054.

[77] U. Kogelschatz, B. Eliasson, and W. Egli, *From ozone generators to flat television screens: history and future potential of dielectric-barrier discharges*, Pure and Applied Chemistry, 71 (1999), pp. 1819–1828.

[78] B. Koren, *A robust upwind discretization method for advection, diffusion and source terms*, Notes on Numerical Fluid Mechanics, Vieweg, Germany, 1993, pp. 117–138.

[79] I. N. Kosarev, A. Y. Starikovskiy, and N. L. Aleksandrov, *Development of high-voltage nanosecond discharge in strongly non-uniform gas*, Plasma Sources Science and Technology, 28 (2019), p. 015005.

[80] A. A. Kulikovsky, *The role of photoionization in positive streamer dynamics*, Journal of Physics D: Applied Physics, 33 (2000), pp. 1514–1524.

[81] E. Kunhardt and P. Williams, *Direct solution of poisson's equation in cylindrically symmetric geometry: A fast algorithm*, Journal of Computational Physics, 57 (1985), pp. 403–414.

[82] G. Lapenta, F. Iinoya, and J. U. Brackbill, *Particle-in-cell simulation of glow discharges in complex geometries*, IEEE Transactions on Plasma Science, 23 (1995), pp. 769–779.

[83] E. W. Larsen, G. Thömmes, A. Klar, M. Seaid, and T. Götz, *Simplified PN approximations to the equations of radiative heat transfer and applications*, Journal of Computational Physics, 183 (2002), pp. 652–675.

[84] R. J. LeVeque, *Finite volume methods for hyperbolic problems*, vol. 31, Cambridge university press, 2002.

[85] D. Levko, M. Pachuilo, and L. L. Raja, *Particle-in-cell modeling of streamer branching in CO2 gas*, Journal of Physics D: Applied Physics, 50 (2017), p. 354004.

[86] R. Löhner, K. Morgan, J. Peraire, and M. Vahdati, *Finite element flux-corrected transport (fem–fct) for the euler and navier–stokes equations*, International Journal for Numerical Methods in Fluids, 7 (1987), pp. 1093–1109.

[87] R. Löhner, K. Morgan, M. Vahdati, J. P. Boris, and D. L. Book, *Fem-fct: Combining unstructured grids with high resolution*, Communications in Applied Numerical Methods, 4 (1988), pp. 717–729.

[88] C. Li, W. J. M. Brok, U. Ebert, and J. J. A. M. van der Mullen, *Deviations from the local field approximation in negative streamer heads*, Journal of Applied Physics, 101 (2007), p. 123305.

[89] C. Li, U. Ebert, and W. J. M. Brok, *Avalanche-to-streamer transition in particle simulations*, IEEE Transactions on Plasma Science, 36 (2008), pp. 910–911.

[90] C. Li, U. Ebert, W. J. M. Brok, and W. Hundsdorfer, *Spatial coupling of particle and fluid models for streamers: where nonlocality matters*, Journal of Physics D: Applied Physics, 41 (2008), p. 032005.

[91] C. Li, U. Ebert, and W. Hundsdorfer, *Spatially hybrid computations for streamer discharges with generic features of pulled fronts: I. planar fronts*, Journal of Computational Physics, 229 (2010), pp. 200–220.

[92] ——, *Spatially hybrid computations for streamer discharges : Ii. fully 3d simulations*, Journal of Computational Physics, 231 (2012), pp. 1020–1050. Special Issue: Computational Plasma Physics.

[93] R. Li, T. Tang, and P. Zhang, *Moving mesh methods in multiple dimensions based on harmonic maps*, Journal of Computational Physics, 170 (2001), pp. 562–588.

[94] Z. Liang, Z. Gimbutas, L. Greengard, J. Huang, and S. Jiang, *A fast multipole method for the rotne–prager–yamakawa tensor and its applications*, Journal of Computational Physics, 234 (2013), pp. 133–139.

[95] G. Liao and D. Anderson, *A new approach to grid generation*, Applicable Analysis, 44 (1992), pp. 285–298.

[96] B. Lin, C. Zhuang, Z. Cai, R. Zeng, and W. Bao, *Accurate and efficient calculation of photoionization in streamer discharges using the fast multipole method*, Plasma Sources Science and Technology, to appear, (2020), arXiv:2006.03515.

[97] ——, *An efficient and accurate mpi-based parallel simulator for streamer discharges in three dimensions*, Journal of Computational Physics, 401 (2020), p. 109026.

[98] N. Liu, S. Célestin, A. Bourdon, V. P. Pasko, P. Ségur, and E. Marode, *Application of photoionization models based on radiative transfer and the helmholtz equations to studies of streamers in weak electric fields*, Applied Physics Letters, 91 (2007), p. 211501.

[99] N. Liu, J. R. Dwyer, H. C. Stenbaek-Nielsen, and M. G. McHarg, *Sprite streamer initiation from natural mesospheric structures*, Nature Communications, 6 (2015), pp. 1–9.

[100] N. Liu and V. P. Pasko, *Effects of photoionization on propagation and branching of positive and negative streamers in sprites*, Journal of Geophysical Research: Space Physics, 109 (2004).

[101] L. B. Loeb and A. F. Kip, *Electrical discharges in air at atmospheric pressure the nature of the positive and negative point-to-plane coronas and the mechanism of spark propagation*, Journal of Applied Physics, 10 (1939), pp. 142–160.

[102] Q.-Z. Luo, N. D'Angelo, and R. L. Merlino, *Shock formation in a negative ion plasma*, Physics of Plasmas, 5 (1998), pp. 2868–2870.

[103] A. Luque and U. Ebert, *Density models for streamer discharges: Beyond cylindrical symmetry and homogeneous media*, Journal of Computational Physics, 231 (2012), pp. 904–918.

[104] A. Luque, U. Ebert, and W. Hundsdorfer, *Interaction of streamer discharges in air and other oxygen-nitrogen mixtures*, Phys. Rev. Lett., 101 (2008), p. 075005.

[105] A. Luque, U. Ebert, C. Montijn, and W. Hundsdorfer, *Photoionization in negative streamers: Fast computations and two propagation modes*, Applied Physics Letters, 90 (2007), p. 081501.

[106] W. A. Lyons, R. A. Armstrong, E. A. Bering III, and E. R. Williams, *The hundred year hunt for the sprite*, Eos, Transactions American Geophysical Union, 81 (2000), pp. 373–377.

[107] D. Malhotra and G. Biros, *PVFMM: A parallel kernel independent FMM for particle and volume potentials*, Communications in Computational Physics, 18 (2015), p. 808–830.

[108] E. Marode, *The mechanism of spark breakdown in air at atmospheric pressure between a positive point and a plane. i. experimental: Nature of the streamer track*, Journal of Applied Physics, 46 (1975), pp. 2005–2015.

[109] R. Marskar, *An adaptive cartesian embedded boundary approach for fluid simulations of two-and three-dimensional low temperature plasma filaments in complex geometries*, Journal of Computational Physics, 388 (2019), pp. 624–654.

[110] R. Marskar, *Adaptive multiscale methods for 3d streamer discharges in air*, Plasma Research Express, 1 (2019), p. 015011.

[111] ——, *3d fluid modeling of positive streamer discharges in air with stochastic photoionization*, Plasma Sources Science and Technology, 29 (2020), p. 055007.

[112] D. F. Martin and K. L. Cartwright, *Solving Poisson's equation using adaptive mesh refinement*, Citeseer, 1996.

[113] P. S. Maruvada, *Corona performance of high-voltage transmission lines*, Research Studies Press Baldock, Herfordshire, United Kingdom, 2000.

[114] J. M. Meek, *A theory of spark discharge*, Phys. Rev., 57 (1940), pp. 722–728.

[115] W.-G. Min, H.-S. Kim, S.-H. Lee, and S.-Y. Hahn, *An investigation of fem-fct method for streamer corona simulation*, IEEE Transactions on Magnetics, 36 (2000), pp. 1280–1284.

[116] ——, *A study on the streamer simulation using adaptive mesh generation and fem-fct*, IEEE Transactions on Magnetics, 37 (2001), pp. 3141–3144.

[117] R. Mittal and G. Iaccarino, *Immersed boundary methods*, Annual Review of Fluid Mechanics, 37 (2005), pp. 239–261.

[118] D. Modiano and P. Colella, *A higher-order embedded boundary method for time-dependent simulation of hyperbolic conservation laws*, tech. report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2000.

[119] C. Montijn and U. Ebert, *Diffusion correction to the raether–meek criterion for the avalanche-to-streamer transition*, Journal of Physics D: Applied Physics, 39 (2006), pp. 2979–2992.

[120] C. Montijn, W. Hundsdorfer, and U. Ebert, *An adaptive grid refinement strategy for the simulation of negative streamers*, Journal of Computational Physics, 219 (2006), pp. 801–835.

[121] R. Morrow, *Numerical solution of hyperbolic equations for electron drift in strongly non-uniform electric fields*, Journal of Computational Physics, 43 (1981), pp. 1–15.

[122] R. Morrow, *Space-charge effects in high-density plasmas*, Journal of Computational Physics, 46 (1982), pp. 454–461.

[123] R. Morrow and J. J. Lowke, *Streamer propagation in air*, Journal of Physics D: Applied Physics, 30 (1997), pp. 614–627.

[124] D. Moudry, H. Stenbaek-Nielsen, D. Sentman, and E. Wescott, *Imaging of elves, halos and sprite initiation at 1ms time resolution*, Journal of Atmospheric and Solar-Terrestrial Physics, 65 (2003), pp. 509–518.

[125] G. V. Naidis, *On photoionization produced by discharges in air*, Plasma Sources Science and Technology, 15 (2006), pp. 253–255.

[126] S. Nijdam, F. M. J. H. van de Wetering, R. Blanc, E. M. van Veldhuizen, and U. Ebert, *Probing photo-ionization: experiments on positive streamers in pure gases and mixtures*, Journal of Physics D: Applied Physics, 43 (2010), p. 145204.

[127] M. M. Nudnova and A. Y. Starikovskii, *Streamer head structure: role of ionization and photoionization*, Journal of Physics D: Applied Physics, 41 (2008), p. 234003.

[128] C. Oosterlee and T. Washio, *On the use of multigrid as a preconditioner*, in Proceedings of Ninth International Conference on Domain Decomposition Methods, 1996, pp. 441–448.

[129] S. Pancheshnyi, *Role of electronegative gas admixtures in streamer start, propagation and branching phenomena*, Plasma Sources Science and Technology, 14 (2005), pp. 645–653.

[130] S. Pancheshnyi, *Photoionization produced by low-current discharges in o2, air, n2and CO2*, Plasma Sources Science and Technology, 24 (2014), p. 015023.

[131] S. Pancheshnyi, M. Nudnova, and A. Starikovskii, *Development of a cathode-directed streamer discharge in air at different pressures: Experiment and comparison with direct numerical simulation*, Phys. Rev. E, 71 (2005), p. 016407.

[132] S. Pancheshnyi, P. Ségur, J. Capeillère, and A. Bourdon, *Numerical simulation of filamentary discharges with parallel adaptive mesh refinement*, Journal of Computational Physics, 227 (2008), pp. 6574–6590.

[133] S. V. Pancheshnyi, S. M. Starikovskaia, and A. Y. Starikovskii, *Role of photoionization processes in propagation of cathode-directed streamer*, Journal of Physics D: Applied Physics, 34 (2000), pp. 105–115.

[134] R. B. Pember, J. B. Bell, P. Colella, W. Y. Curtchfield, and M. L. Welcome, *An adaptive cartesian grid method for unsteady compressible flow in irregular regions*, Journal of Computational Physics, 120 (1995), pp. 278–304.

[135] G. W. Penney and G. T. Hummert, *Photoionization measurements in air, oxygen, and nitrogen*, Journal of Applied Physics, 41 (1970), pp. 572–577.

[136] C. S. Peskin, *Flow patterns around heart valves: A numerical method*, Journal of Computational Physics, 10 (1972), pp. 252–271.

[137] J.-M. Plewa, O. Eichwald, O. Ducasse, P. Dessante, C. Jacobs, N. Renon, and M. Yousfi, *3D streamers simulation in a pin to plane configuration using massively parallel computing*, Journal of Physics D: Applied Physics, 51 (2018), p. 095206.

[138] D. Q. Posin, *The townsend coefficients and spark discharge*, Physical Review, 50 (1936), p. 650.

[139] R. Quey, P. Dawson, and F. Barbe, *Large-scale 3d random polycrystals for the finite element method: Generation, meshing and remeshing*, Computer Methods in Applied Mechanics and Engineering, 200 (2011), pp. 1729–1745.

[140] Y. Saad, *A flexible inner-outer preconditioned GMRES algorithm*, SIAM Journal on Scientific Computing, 14 (1993), pp. 461–469.

[141] Y. Saad, *Iterative methods for sparse linear systems*, vol. 82, siam, 2003.

[142] W. Samaranayake, Y. Miyahara, T. Namihira, S. Katsuki, T. Sakugawa, R. Hackam, and H. Akiyama, *Pulsed streamer discharge characteristics of ozone production in dry air*, IEEE Transactions on Dielectrics and Electrical Insulation, 7 (2000), pp. 254–260.

[143] P. Schwartz, M. Barad, P. Colella, and T. Ligocki, *A cartesian grid embedded boundary method for the heat equation and poisson's equation*

in three dimensions, Journal of Computational Physics, 211 (2006), pp. 531–550.

[144] P. Ségur, A. Bourdon, E. Marode, D. Bessieres, and J. H. Paillol, *The use of an improved Eddington approximation to facilitate the calculation of photoionization in streamer discharges*, Plasma Sources Science and Technology, 15 (2006), pp. 648–660.

[145] F. Shi, N. Liu, and J. R. Dwyer, *Three-dimensional modeling of two interacting streamers*, Journal of Geophysical Research: Atmospheres, 122 (2017), pp. 10,169–10,176.

[146] C.-W. Shu, *Total-variation-diminishing time discretizations*, SIAM Journal on Scientific and Statistical Computing, 9 (1988), pp. 1073–1084.

[147] H. C. Stenbaek-Nielsen, T. Kanmae, M. G. McHarg, and R. Haaland, *High-speed observations of sprite streamers*, Surveys in Geophysics, 34 (2013), pp. 769–795.

[148] J. Stephens, M. Abide, A. Fierro, and A. Neuber, *Practical considerations for modeling streamer discharges in air with radiation transport*, Plasma Sources Science and Technology, 27 (2018), p. 075007.

[149] H. Sundar, G. Stadler, and G. Biros, *Comparison of multigrid algorithms for high-order continuous finite element discretizations*, Numerical Linear Algebra with Applications, 22 (2015), pp. 664–680.

[150] P. Šunka, *Pulse electrical discharges in water and their applications*, Physics of plasmas, 8 (2001), pp. 2587–2594.

[151] O. Tatebe, *The multigrid preconditioned conjugate gradient method*, in The Sixth Copper Mountain Conference on Multigrid Methods, Part 2, 1993, pp. 621–634.

[152] M. J. Taylor, M. A. Bailey, P. D. Pautet, S. A. Cummer, N. Jaugey, J. N. Thomas, N. N. Solorzano, F. Sao Sabbas, R. H. Holzworth, O. Pinto, and N. J. Schuch, *Rare measurements of a sprite with halo event driven by a negative lightning discharge over argentina*, Geophysical Research Letters, 35 (2008).

[153] J. Teunissen and U. Ebert, *3d PIC-MCC simulations of discharge inception around a sharp anode in nitrogen/oxygen mixtures*, Plasma Sources Science and Technology, 25 (2016), p. 044005.

[154] J. Teunissen and U. Ebert, *Simulating streamer discharges in 3D with the parallel adaptive Afivo framework*, Journal of Physics D: Applied Physics, 50 (2017), p. 474001.

[155] J. Teunissen, A. Sun, and U. Ebert, *A time scale for electrical screening in pulsed gas discharges*, Journal of Physics D: Applied Physics, 47 (2014), p. 365203.

[156] F. Tholin, D. L. Rusterholtz, D. A. Lacoste, D. Z. Pai, S. Celestin, J. Jarrige, G. D. Stancu, A. Bourdon, and C. O. Laux, *Images of a nanosecond repetitively pulsed glow discharge between two point electrodes in air at 300 k and at atmospheric pressure*, IEEE Transactions on Plasma Science, 39 (2011), pp. 2254–2255.

[157] J. F. Thompson, Z. U. Warsi, and C. W. Mastin, *Numerical grid generation: foundations and applications*, vol. 45, North-holland Amsterdam, 1985.

[158] T. Toulorge, C. Geuzaine, J.-F. Remacle, and J. Lambrechts, *Robust untangling of curvilinear meshes*, Journal of Computational Physics, 254 (2013), pp. 8–26.

[159] J. S. Townsend, *Electricity in gases*, Oxford at the Clarendon Press, 1915.

[160] T. N. Tran, I. O. Golosnoy, P. L. Lewin, and G. E. Georghiou, *Numerical modelling of negative discharges in air with experimental validation*, Journal of Physics D: Applied Physics, 44 (2010), p. 015203.

[161] U. Trottenberg, C. W. Oosterlee, and A. Schuller, *Multigrid*, Elsevier, 2000.

[162] O. A. van der Velde, J. Bór, J. Li, S. A. Cummer, E. Arnone, F. Zanotti, M. Füllekrug, C. Haldoupis, S. NaitAmor, and T. Farges, *Multi-instrumental observations of a positive gigantic jet produced by a winter thunderstorm in europe*, Journal of Geophysical Research: Atmospheres, 115 (2010).

[163] B. Van Leer, *Towards the ultimate conservative difference scheme. iv. a new approach to numerical convection*, Journal of computational physics, 23 (1977), pp. 276–299.

[164] ——, *Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method*, Journal of Computational Physics, 32 (1979), pp. 101–136.

[165] P. L. Ventzek, T. J. Sommerer, R. J. Hoekstra, and M. J. Kushner, *Two-dimensional hybrid model of inductively coupled plasma sources for etching*, Applied Physics Letters, 63 (1993), pp. 605–607.

[166] A. Villa, L. Barbieri, M. Gondola, A. R. Leon-Garzon, and R. Malgesini, *Stability of the discretization of the electron avalanche phenomenon*, Journal of Computational Physics, 296 (2015), pp. 369–381.

[167] ——, *A pde-based partial discharge simulator*, Journal of Computational Physics, 345 (2017), pp. 687–705.

[168] A. Villa, L. Barbieri, M. Gondola, and R. Malgesini, *An asymptotic preserving scheme for the streamer simulation*, Journal of Computational Physics, 242 (2013), pp. 86–102.

[169] A. L. Ward, *Calculations of electrical breakdown in air at near-atmospheric pressure*, Phys. Rev., 138 (1965), pp. A1357–A1362.

[170] D. Welch, T. Genoni, R. Clark, and D. Rose, *Adaptive particle management in a particle-in-cell code*, Journal of Computational Physics, 227 (2007), pp. 143–155.

[171] W. J. Yi and P. F. Williams, *Experimental study of streamers in pure n2and n2/o2mixtures and a ≈13 cm gap*, Journal of Physics D: Applied Physics, 35 (2002), pp. 205–218.

[172] L. Ying, G. Biros, and D. Zorin, *A kernel-independent adaptive fast multipole algorithm in two and three dimensions*, Journal of Computational Physics, 196 (2004), pp. 591–626.

[173] L. Ying, G. Biros, D. Zorin, and H. Langston, *A new parallel kernel-independent fast multipole method*, in SC'03: Proceedings of the 2003 ACM/IEEE conference on Supercomputing, IEEE, 2003, pp. 14–14.

[174] K. Yoshida and H. Tagashira, *Computer simulation of a nitrogen discharge at high overvoltages*, Journal of Physics D: Applied Physics, 9 (1976), pp. 491–505.

[175] ——, *Computer simulation of a nitrogen discharge considering the radial electron drift*, Journal of Physics D: Applied Physics, 9 (1976), pp. 485–490.

[176] M. Zakari, H. Caquineau, P. Hotmar, and P. Ségur, *An axisymmetric unstructured finite volume method applied to the numerical modeling of an atmospheric pressure gas discharge*, Journal of Computational Physics, 281 (2015), pp. 473–492.

[177] S. T. Zalesak, *Fully multidimensional flux-corrected transport algorithms for fluids*, Journal of Computational Physics, 31 (1979), pp. 335–362.

[178] R. Zeng and S. Chen, *The dynamic velocity of long positive streamers observed using a multi-frame ICCD camera in a 57 cm air gap*, Journal of Physics D: Applied Physics, 46 (2013), p. 485201.

[179] M. B. Zheleznyak, A. K. Mnatsakanian, and S. V. Sizykh, *Photoionization of nitrogen and oxygen mixtures by radiation from a gas discharge*, High Temperature Science, 20 (1982), pp. 423–428.

[180] Y. Zheng, Q. Li, Y. Chen, S. Shu, and C. Zhuang, *Breakdown path and condition of air-insulated rod–plane gap with polymeric barrier inserted under alternating voltages*, AIP Advances, 9 (2019), p. 105207.

[181] C. Zhuang, M. Huang, and R. Zeng, *Numerical simulations for the quasi-3d fluid streamer propagation model: Methods and Applications*, Communications in Computational Physics, 24 (2018), pp. 1259–1278.

[182] C. Zhuang, B. Lin, R. Zeng, L. Liu, and M. Li, *3-d parallel simulations of streamer discharges in air considering photoionization*, IEEE Transactions on Magnetics, 56 (2020), pp. 1–4.

[183] C. Zhuang and R. Zeng, *A local discontinuous Galerkin method for 1.5-dimensional streamer discharge simulations*, Applied Mathematics and Computation, 219 (2013), pp. 9925–9934.

[184] C. Zhuang, R. Zeng, B. Zhang, and J. He, *2-D discontinuous Galerkin method for streamer discharge simulations in nitrogen*, IEEE Transactions on Magnetics, 49 (2013), pp. 1929–1932.

An efficient and accurate parallel simulator for streamer discharge in three dimensions Lin Bo 2020