Numerical Simulation of Inverse Compton Scattering: Algorithm and Implementation

Dissertation
zur Erlangung des Doktorgrades
an der Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Physik
der Universität Hamburg

vorgelegt von

Yi-Kai Kan

Hamburg
2023

Gutachter/innen der Dissertation:	Prof. Dr. Franz X. Kärtner Prof. Dr. Sabine Le Borne
Zusammensetzung der Prüfungskommission:	Prof. Dr. Franz X. Kärtner Prof. Dr. Sabine Le Borne Prof. Dr. Daniela Pfannkuche Prof. Dr. Nina Rohringer Dr. Jens-Peter M. Zemke
Vorsitzende/r der Prüfungskommission:	Prof. Dr. Daniela Pfannkuche
Datum der Disputation:	22.02.2023
Vorsitzender des Fach-Promotionsausschusses PHYSIK:	Prof. Dr. Günter H. W. Sigl
Leiter des Fachbereichs PHYSIK:	Prof. Dr. Wolfgang J. Parak
Dekan der Fakultät MIN:	Prof. Dr.-Ing. Norbert Ritter

Acknowledgements

I want to express my genuine gratitude to my thesis supervisors Prof. Franz X. Kärtner, Prof. Sabine Le Borne, and Dr. Jens-Peter M. Zemke, for their guidance, continuous support, and encouragement during my doctoral study. Each of them has several good things that I can learn from and inspire me to improve. I am also thankful for all the colleagues I met in UFOX group, DASHH, and DESY. The scientific discussion or the small talk with them not only improved my knowledge of science but also broadened my horizons. In particular, I would like to thank Dr. Moein Fakhari for the collaboration on the AXSIS simulation. I also want to thank Ms. Uta Freydank for her help with numerous administrative stuff. Special thanks also go to the DASHH coordinators, Dr. Christiane Ehrt and Dr. Heike Hufnagel Martinez, for their support on administrative stuff and many other things.

Furthermore, I would also like to express my gratitude to all of the friends I met in Germany. You brought color to my life, and I had a great time with you.

Lastly, I am sincerely grateful to my family for their unconditional and endless support. Without you, there would be no me.

The research studies presented in this thesis were supported by DASHH (Data Science in Hamburg – HELMHOLTZ Graduate School for the Structure of Matter) with the Grant-No. HIDSS-0002.

Abstract

Inverse Compton Scattering (ICS) has gained much attention recently because of its promise for the development of compact X-ray light sources. Precise and fast simulation is an essential tool for predicting the radiation property of a given machine design and to optimize its parameters. This thesis focuses on the development of the ICS simulation model. In particular, two computational techniques used in the simulation model are considered: the calculation of the radiation spectra and the computation of the space-charge field.

The calculation of radiation spectra is based on the Liénard-Wiechert potential. Instead of the conventional method which directly evaluates the discretized Fourier integral of the Liénard-Wiechert field (called frequency-domain method), we focus on an approach referred to as the time-domain method, where the field is recorded along the observer's time on a uniform time grid which is then used to compute the radiation spectra after completion of the simulation. Besides the derivation and implementation details of the proposed method, we analyze possible parallelization schemes and compare the parallel performance of the proposed time-domain method with the frequency-domain method. We will characterize scenarios/conditions under which one method is expected to outperform the other.

The computation of the space-charge field is based on fast summation methods. Based on Lagrange Interpolation, we formulate a treecode and a fast-multipole method (FMM) for the efficient computation of the relativistic space-charge field. In particular, we propose two approaches to control the interpolation error. In the first approach, we derive a modified admissibility condition (ADMC) from the analysis of the interpolation error; with the modified ADMC, the treecode/FMM can be used directly in the lab-frame. The second approach relies on the transformation of the particle beam to the rest-frame where the conventional admissibility condition can be applied. Our numerical results show that two approaches can effectively control the interpolation error and the approach based on the modified ADMC is more accurate for a particle beam with larger momentum spread. The implementation and the GPU parallelization of the solver are also discussed. Our GPU solver can achieve a speedup of roughly 200 compared to the CPU solver.

Zusammenfassung

Inverse Compton-Scattering (ICS) hat in letzter Zeit viel Aufmerksamkeit auf sich gezogen, da es vielversprechend für die Entwicklung kompakter Röntgen-Lichtquellen ist. Eine präzise und schnelle Simulation ist ein wesentliches Instrument zur Vorhersage der Strahlungseigenschaften eines bestimmten Maschinendesigns und zur Optimierung seiner Parameter. Diese Doktorarbeit befasst sich mit der Entwicklung des ICS-Simulationsmodells. Insbesondere werden zwei in dem Simulationsmodell verwendete Berechnungstechniken betrachtet: die Berechnung der Strahlungsspektren und die Berechnung des Raumladungsfeldes.

Die Berechnung der Strahlungsspektren basiert auf dem Liénard-Wiechert-Potential. Anstelle der konventionellen Methode, die direkt das diskretisierte Fourier-Integral des Liénard-Wiechert-Feldes auswertet (die sogenannte Frequenzbereich-Methode), konzentrieren wir uns auf einen Ansatz, der als Zeitbereich-Methode bezeichnet wird. Dabei wird das Feld entlang der Zeit des Beobachters auf einem einheitlichen Zeitgitter aufgezeichnet, das dann nach Abschluss der Simulation zur Berechnung der Strahlungsspektren verwendet wird. Neben der Herleitung und den Implementierungsdetails der vorgeschlagenen Methode analysieren wir mögliche Parallelisierungsschemata und vergleichen die Leistung der vorgeschlagenen Zeitbereich-Methode mit der Frequenzbereich-Methode. Wir beschreiben Szenarien/Bedingungen, unter denen eine Methode der anderen überlegen ist.

Die Berechnung des Raumladungsfeldes basiert auf schnellen Summationsmethoden. Auf der Grundlage der Lagrange-Interpolation formulieren wir einen Baumcode und eine schnelle Multipolmethode (FMM) für die effiziente Berechnung des relativistischen Raumladungsfeldes. Wir schlagen insbesondere zwei Ansätze zur Kontrolle des Interpolationsfehlers vor. Im ersten Ansatz leiten wir eine modifizierte Zulässigkeitsbedingung (ADMC) aus der Analyse des Interpolationsfehlers ab. Mit der modifizierten ADMC kann der/die Baumcode/FMM direkt im Laborsystem verwendet werden. Der zweite Ansatz beruht auf der Transformation des Teilchenstrahls in das Ruhesystem, wo die konventionelle Zulässigkeitsbedingung angewendet werden kann. Unsere numerischen Ergebnisse zeigen, dass die zwei Ansätze den Interpolationsfehler effektiv kontrollieren können und dass der auf der modifizierten ADMC basierende Ansatz für einen Teilchenstrahl mit größerer Impulsstreuung genauer ist. Die Implementierung und die GPU-Parallelisierung des Lösers werden ebenfalls diskutiert. Unser GPU-Löser kann einen Speedup von ca. 200 im Vergleich zum CPU-Löser erreichen.

List of Figures

List of Tables

List of Algorithms

Contents

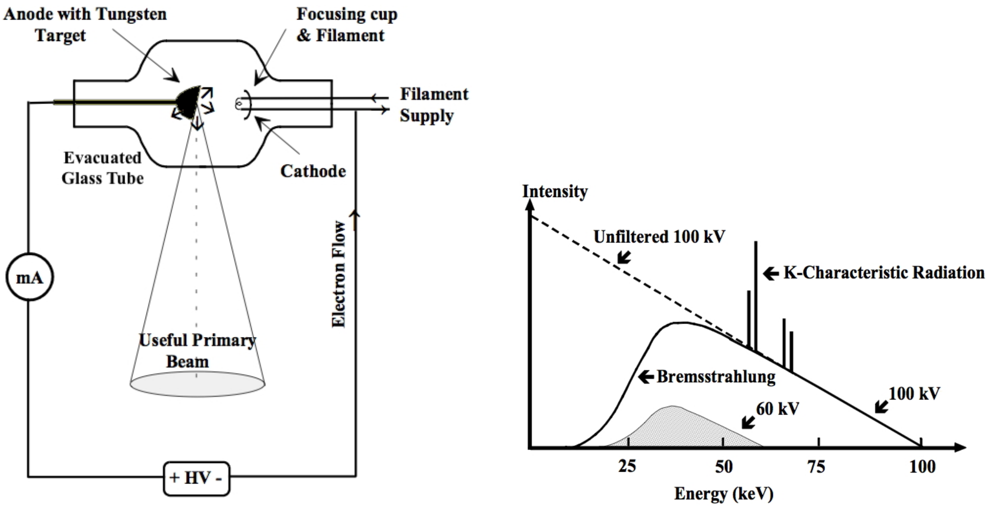
Chapter 1

Introduction

1.1 Background and Motivation

X-rays have been an indispensable tool in both academic and industrial research since its discovery in 1895 by Röntgen [2]; and since then, X-ray tubes have been common devices for the generation of X-ray [3]. In the X-ray tube, an electron beam is accelerated by an applied voltage and is then guided to collide with a metallic target (Figure 1.1a). When the energetic electrons approach the nucleus, they will be deflected by the force-field of the nucleus. The electrons experience an instantaneous acceleration/deceleration and emit Bremsstrahlung with a continuous spectrum. In addition, some characteristic peaks can also be found in the spectrum (Figure 1.1b). When an inner shell electron of an atom is kicked out by an incident electron, an "electron hole" is created in the inner shell. An outer shell electron will fill the electron hole and an X-ray photon with a specific energy will be generated. X-ray tubes are widely commercially available because they are compact and affordable. However, the X-ray tube has limited photo flux due to its working mechanism called electron impact, where the most energy of the electrons is dissipated as heat on the target [4]. Thus, the heat load will limit the operation power of an X-ray tube and then the generated photon flux. Some attempts have been made to overcome the heat load problem, *e.g.*, rotating anode sources [3] and liquid metal jet sources [5, 6]. However, all these methods are reaching the physical limit that the electron impact can achieve [7].

To overcome the physical limitations of the X-ray tubes, accelerator-based light sources have been developed for applications that require X-rays with higher brightness. In accelerator-based light sources, radiation is generated from the free electrons moving in a vacuum chamber. This approach does not suffer from the heat-load problem of the X-ray tubes. Thus, the accelerator-based light sources can be operated at much higher power and produce much more photon flux. Besides, due to the advancement in accelerator technology for the past several decades, electron beams can be focused to a very small transverse size. This further improves the brightness because of the smaller emission cross section. Synchrotron radiation is probably the most common accelerator-based light source nowadays. Synchrotron radiation is generated when a charged particle experiences acceleration and moves on a curved trajectory. A synchrotron facility contains a storage ring, where high-energy electrons are guided by bending magnets to move on a closed-loop trajectory. Besides, an accelerating cavity is deployed on the path of the electrons so that their energy can be refilled and their orbiting can last for several turns. A typical synchrotron facility provides light sources from three devices: bending magnet, wiggler and undulator. When passing through a bending magnet, an electron traverses in a curved trajectory with a radius of curvature proportional to the magnetic field. This produces radiation with a continuous spectrum and the emission power and



(a) structure (b) spectrum

Figure 1.1: Illustration of the X-ray tube. Figure 1.1a shows the simple sketch of an X-ray tube and Figure 1.1b shows the radiation spectrum generated from an X-ray tube (images by Kieranmaher in the public domain).

angular range are proportional to γ^4 and $1/\gamma$, respectively [1]. This is one profound property that makes the accelerator-based sources orders of magnitude brighter than the X-ray tube sources because the relativistic effects increase radiation flux and angular collimation. A wiggler can be considered a sequence of bending magnets with alternating polarities, which wiggles a particle along an axis. The overall radiation spectrum is the incoherent superposition of spectra from the electron motion in each bending magnet, *i.e.*, the overall radiation intensity enhances N_u times for a wiggler with N_u oscillation periods. Thus, the wiggler radiation has higher brightness than a bending magnet and is usually referred to as the second-generation synchrotron source [8]. The undulator is conceptually the same device as the wiggler, but the undulator has smaller magnetic field and the particle is wiggled more gently than in a wiggler. The maximum bending angle for a particle (*i.e.*, the maximum angle between the longitudinal axis and the tangent to the trajectory) moving in an undulator is $\sim K/\gamma$ [8]. Here, K is the undulator parameter which is proportional to the amplitude of the magnetic field. If the amplitude of the magnetic field is chosen with a magnitude so that the maximum bending angle of the particle is less than the angular range of the radiation from a bending magnet, we have the condition

$$\frac{K}{\gamma} \ll \frac{1}{\gamma} \quad \Longrightarrow \quad K \ll 1. \tag{1.1.1}$$

This is a criterion for a typical undulator. If $K \ll 1$ is fulfilled, the radiation emitted from different parts of the particle trajectory can coherently interfere with each other in various harmonics of the fundamental wavelength. The fundamental wavelength of the radiation can be approximately written as

$$\lambda_{\text{rad}} = \frac{\lambda_u}{2\gamma^2}\left(1 + \frac{K^2}{2} + \gamma^2\theta^2\right). \tag{1.1.2}$$

Here, λ_u is the undulator period and θ is the observation angle relative to the longitudinal axis in the 2D plane that a particle wiggles. Because a particle wiggles N_u periods in the undulator, the emitted radiation is like a sinusoidal wave with a finite pulse length of $N_u \lambda_{rad}$. Therefore, the radiation intensity I at a wavelength λ around the fundamental wavelength λ_{rad} is proportional to

$$I(\lambda) \propto \frac{\sin^2(\pi N_u \Delta\omega)}{(\pi\Delta\omega)^2} \quad \text{with} \quad \Delta\omega := \frac{2\pi c_0}{\lambda} - \frac{2\pi c_0}{\lambda_{\mathrm{rad}}},$$

where c_0 is the speed of light. Therefore, the radiation intensity at the fundamental wavelength increases N_u^2 times compared to the radiation intensity of a single bending magnet. Also, the corresponding bandwidth is $\approx 1/N_u$. By using this fact and (1.1.2), we can derive that the radiation angular range covering this bandwidth is $\sim 1/(\sqrt{N_u}\gamma)$. Therefore, the brightness of the undulator radiation can be several orders larger than the wiggler radiation and is also referred to as the third-generation synchrotron source [8]. It is worth mentioning again that the undulator and the wiggler are conceptually the same devices, apart from that the wiggler has a large undulator parameter $K \gg 1$.

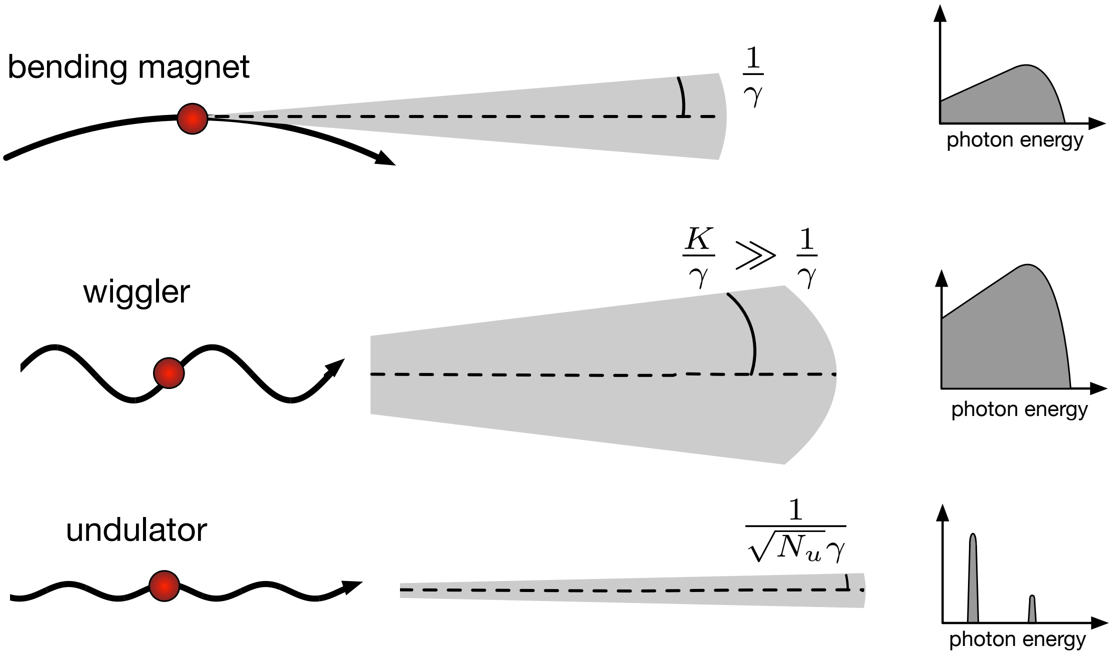


Figure 1.2: The radiation from the bending magnet, wiggler and undulator. The electron (red circle) trajectory and the spectrum for each device are illustrated.

The properties of the undulator radiation generated from a single electron have been discussed. Without loss of generality, the radiation intensity from an N-particle bunch can be written as [9]

$$I(\lambda) \propto N + N(N-1) \cdot |F_{\mathrm{L}}(\lambda)|^2,$$

with

$$F_{\mathrm{L}}(\lambda) := \int \rho_L(z) \exp(-j\tfrac{2\pi}{\lambda}z)dz \quad \text{and} \quad \int \rho_L(z)dz = 1.$$

Here, $F_{\mathrm{L}}(\lambda)$ is the longitudinal form factor and $\rho_L(z)$ is the normalized particle density in the longitudinal direction. Consider a Gaussian-distributed particle density with RMS bunch length σ_z

$$\rho_{\mathrm{L}}(\lambda) = \frac{1}{\sigma_z\sqrt{2\pi}} \exp\left(-\frac{1}{2}(\frac{z}{\sigma_z})^2\right)$$

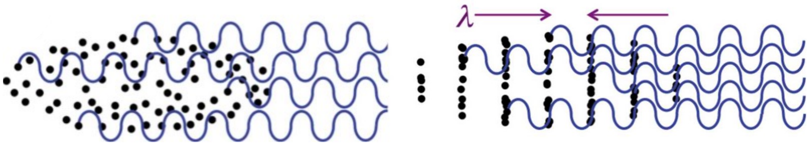
and its corresponding form factor

$$F_{\mathrm{L}}(\lambda) = \exp\left(-2\pi^2(\frac{\sigma_z}{\lambda})^2\right).$$

In a practical scenario, we usually have $\sigma_z/\lambda \gg 1$, *e.g.*, a bunch length in microns and a radiation wavelength in subnanometers. Thus, the form factor is almost zero $F_{\mathrm{L}} \approx 0$ and the radiation intensity is $I(\lambda) \propto N$. This is due to the incoherent interference of the radiation field from the electrons (Figure 1.3a). To produce a coherent interference of the radiation field from the electrons, we may slice the electron bunch into M sheets of electrons (also called microbunches) with a fixed spacing ℓ_z between each sheet. The longitudinal particle density and the corresponding form factor can be written as

$$\rho_L(z) = \frac{1}{M}\sum_{m=0}^{M-1}\delta(z - m\ell_z) \quad \text{and} \quad F_{\mathrm{L}}(\lambda) = \frac{1}{M}\frac{\sin(\pi M\frac{\ell_z}{\lambda})}{\sin(\pi\frac{\ell_z}{\lambda})}.$$

If the spacing between each microbunch is equal to the radiation wavelength $\ell_z = \lambda$, the radiation field emitted from the electrons at the different sheets can be coherently added up (Figure 1.3b). Hence, the form factor becomes $F_{\mathrm{L}}(\lambda) = 1$ and the radiation intensity will be $I(\lambda) \propto N^2$. The process of slicing particles into a sequence of sheets is called microbunching (Figure 1.4) and is a key component of the free-electron laser (FEL). When a particle bunch enters into an undulator, some electrons start to emit undulator radiation in the fundamental wavelength. The superposition of the undulator field and the radiation field produces the so-called longitudinal ponderomotive force which drives electrons to form microbunches [1]. With this microbunching mechanism, an FEL can produce the brightest X-ray pulses today [10, 11]. The undulator usually has a period on the order of 10 cm. According to (1.1.2), a highly



(a) incoherent emission (b) coherent emission

Figure 1.3: Schematics of the radiation wave (sinusoidal curve) generation from the electrons (black dot) with (a) incoherent emission and (b) coherent emission at a wavelength λ. The Images are excerpted from Ref. [1].

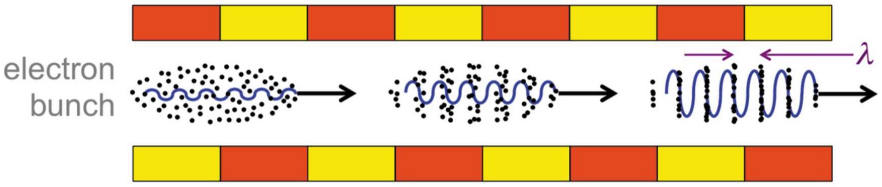


Figure 1.4: The microbunching of electrons in the undulator. The emitted radiation wave (sinusoidal curve) λ interacts with the electrons (black dots) and gradually drives them to form microbunches with the spacing which is the same as the wavelength λ. The Image is excerpted from Ref. [1].

relativistic beam on the order of GeV is needed to produce the undulator radiation in the hard X-ray regime. Therefore, the conventional accelerator-based light sources rely on the large particle accelerators, *e.g.*, synchrotron storage rings usually have the circumferences of $0.5 - 2\,\text{km}$ [12, 13, 14, 15] and the FEL facilities usually have lengths of $1 - 2\,\text{km}$ [16, 17, 18]. These large facilities require high maintenance costs and are only affordable for government-funded research centers.

The inverse Compton scattering (ICS) source might be a potential alternative to the conventional accelerator-based light source. Not only do ICS sources produce X-ray with more brightness than the X-ray tube source, but they are also compact and more affordable than the conventional accelerator-based source. In the ICS process, the electron beam collides with a counter-propagating high-intensity laser pulse (Figure 1.5). The periodically oscillating electric field in the laser pulse wiggles the electrons, and the electrons emit undulator-like radiation. The formula for the fundamental wavelength of the generated radiation is

$$\lambda_{\text{rad}} = \frac{\lambda_\ell}{4\gamma^2}\left(1 + \frac{a_0^2}{2} + \gamma^2\theta^2\right), \tag{1.1.3}$$

where λ_ℓ is the laser wavelength and a_0 is the normalized vector potential characterizing the strength of the laser field [19]. One may notice that (1.1.3) can be obtained by applying the substitutions

$$\lambda_u \to \lambda_\ell/2 \quad \text{and} \quad K \to a_0$$

in (1.1.2). Hence, the counter-propagating laser pulse used in the ICS sources is often called optical undulator. Because the optical undulator period (*i.e.*, the laser wavelength λ_ℓ) is much smaller than the period of the mechanical undulator, only low-energy electron beams are required to produce radiations in the X-ray regime. For example, if a laser with a wavelength $\lambda_\ell = 1\,\mu\text{m}$ and a strength $a_0 = 0.5$ is considered, only a 23 MeV electron beam is needed to produce 10 keV photons. This can significantly reduce the size of the particle accelerator and make the overall facility compact. Therefore, ICS sources have drawn great attention over the last ten years because of the potential to allow for table-top-size hard X-ray light sources with a much improved performance [20, 21] when compared to an X-ray tube, eventually approaching that of a second generation synchrotron [22]. Such X-ray light sources would greatly increase access to high brightness X-ray radiation for fast experimental turn-around rather than waiting many months to use the light source provided by large national facilities [23] including international travel to these facilities. Structural biology, biomedical imaging and materials screening would greatly benefit from such developments [24]. Due to the increasing demand for developing ICS sources [24, 25, 26, 27, 23, 28, 29, 30, 31], numerical simulation is an indispensable tool to understand the generated radiation characteristics. The following chapters of the thesis present our research on the computational techniques for the simulation of ICS.

1.2 Overview of the Thesis

Chapter 2 covers our research on the radiation spectra calculation using the Liénard-Wiechert potential method [32]. We first introduce the Liénard-Wiechert potential and discuss two numerical schemes for solving it. After that, we discuss a simulation model for the ICS radiation spectra calculation based on the Liénard-Wiechert

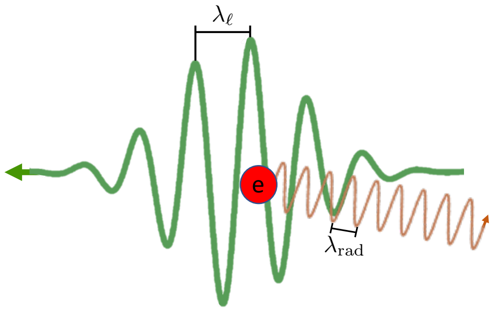


Figure 1.5: Illustration of the ICS process. An electron (red circle) collides with a counter-propagating laser pulse (green line) of wavelength λ_ℓ and generates radiation (brown line) with shorter wavelength λ_{rad}.

potential methods. In this chapter, we argue that the space-charge effect might be negligible for the simulation of conventional ICS light sources. And we first neglect the space-charge effect in the simulation model and propose an efficient time-domain discretization method for solving Liénard-Wiechert potentials. We will compare the time-domain method with the conventional frequency-domain method, and characterize scenarios/conditions under which one method is expected to outperform the other.

Chapter 3, Chapter 4 and Chapter 5 present our studies of modeling the space-charge effect in ICS using a fast summation method. Chapter 3 introduces an interpolation-based treecode for the efficient computation of the relativistic space-charge field [33]. We first give an overview of the commonly-used computational approaches for the evaluation of the space-charge field of a charged particle beam. After that, the idea of an interpolation-based treecode is introduced. Treecode is a well-known approximation technique used for the fast summation of the force-field for an N-body problem. It is famous for scaling as $\mathcal{O}(N \log N)$ compared to the direct summation method with a complexity of $\mathcal{O}(N^2)$. Following the idea of interpolation-based treecode, we propose a so-called "relativistic kernel" and formulate a treecode for the efficient computation of the relativistic space-charge field. Two approaches are proposed to control the interpolation error. Our first approach is based on the relativity transformation. We transform the particle position and momentum to the rest-frame so that our formulated treecode (after transformation) can be performed with the conventional admissibility condition in the rest-frame. Different to the quasi-electrostatic method, the particle momentum spread is included in our formulation. The quasi-electrostatic method assumes that each particle is completely stationary in the rest-frame; hence, the Poisson equation is solved for the force-field and the effect of the particle momentum spread on the force-field is neglected. In the second approach, we analyze the interpolation error and derive a so-called "stretched admissibility condition" of the particle-cluster interaction. With this stretched admissibility condition, our proposed treecode can be performed directly in the lab-frame. Our numerical result shows the approach using stretched admissibility is more accurate than the approach using the relativity transformation when a particle beam with a bigger momentum spread is considered. The reason for the cause will also be explained. In Chapter 4, based on the proposed treecode, we formulate a fast multipole method (FMM) for the efficient evaluation of the relativistic space-charge field. Compared to treecode, FMM can further bring the complexity down to $\mathcal{O}(N)$. The derivation of the stretched admissibility condition for the cluster-cluster interaction is also provided. Our FMM is based on the dual tree traversal and the pseudocode of the whole FMM procedure is also provided. Besides, we demonstrate the performance of the solver and the code profiling result of different FMM kernels. In Chapter 5, we present a GPU parallelization for our dual-tree-traversal-based FMM. The implementation of M2L and P2P will be discussed in detail as they are the most time-consuming kernels in FMM. In particular, we will describe an approach to resolve the race conditions during the GPU execution of M2L and P2P without using the atomic operations provided by CUDA. Our performance result shows that our GPU solver can achieve a speed up of roughly 200 compared to the execution time with one CPU core. We also compare the result with ASTRA (a solver for beam dynamics simulation [34]) by simulating the beam size broadening of a charged particle beam freely propagating in space.

In Chapter 6, we consider the space-charge effect in the ICS simulation model and describe a GPU-parallelized simulation solver. This solver is implemented based on the computational techniques discussed in the preceding chapter of this thesis. We also demonstrate the results of a real-particle ICS simulation with and without the consideration of the space-charge force. Our result shows that the radiation energy spectra for the cases with and without the space-charge force are consistent. This validates our original argument made in Chapter 2 that the space-charge effect might be negligible for the simulation of conventional ICS light sources. Finally, this thesis closes with a summary and outlook in Chapter 7.

Chapter 2

Calculation of Inverse Compton Scattering Radiation Spectra based on Liénard-Wiechert Potentials

The content of this chapter is adapted from our research work published in Ref. [32]. We introduce the Liénard-Wiechert potential for the radiation spectra calculation of ICS sources; and based on that, a simulation model without the consideration of the space-charge effect is discussed. We first describe the frequency-domain method for the computation of the radiation spectra based on the Liénard-Wiechert potential; after that, we introduce our discretization and implementation of the time-domain approach and provide an analysis for two possible distributed parallelization schemes. Finally, we propose and discuss a strategy for choosing parameters when applying the time-domain method and the frequency-domain method to compute ICS radiation spectra. Following this strategy, we analyze the performance of both methods and discuss the scenarios where one method outperforms the other.

2.1 Introduction

A numerical simulation is an indispensable tool for understanding the generated radiation characteristics. To simulate the generated radiation, the Liénard-Wiechert field method is among the commonly used methods [35, 36, 37, 38, 39]. In this method, the radiation field is computed from the charged particle trajectories which are either given beforehand or solved in parallel during the simulation.

One of the challenges when computing radiation spectra generated by relativistic particles lies in the delay between the particle generating its contribution (retarded or emission time) and the observer detecting it (advanced time) which is not constant but depends on the distance between the particle and the observer at the time of emission. Furthermore, to allow for simple and quick computation of spectra via fast Fourier transform, it is preferable to have contributions at the observer on a uniform, equidistant time mesh. One possibility is to define a uniform mesh in the advanced time and then calculate the corresponding emission times ("retarded time scheme"). However, this approach is computationally expensive as it requires numerical root-finding and it is also memory intensive as full particle trajectories must be stored to allow interpolation to the computed emission times. By contrast, the advanced time scheme computes particle trajectories and emissions using a uniform time grid. At each time step, the corresponding advanced time is computed when the generated emission reaches the observer. This approach avoids computationally expensive root-finding but raises the new problem of how to deal with the different (non-uniform) advanced times at which the emissions reach the observer. Since one is typically interested in the amplitude $|E(x, \omega)|$ of the generated field in the frequency-domain [40] and not $E(x, t)$ itself, instead of first depositing contributions in physical space on a mesh and then applying a Fourier transform, it is possible to directly compute the amplitude that the particle contributes to each Fourier mode. This approach is commonly referred to as the frequency-domain method since it calculates directly the spectrum of the generated field but never the field itself [35, 40]. However, as the computation of each Fourier mode is performed independently, this method intrinsically needs a high number of operations, i. e., $\mathrm{O}(N_\omega \cdot N_p \cdot N_s)$ where N_ω, N_p and N_s are the number of frequency points for the spectrum, the number of simulated particles and the number of simulation steps, respectively [40]. As an alternative, Sell et al. [41] use a time-domain approach where particle contributions at a given advanced time are interpolated to a uniform observer time mesh before being (Fourier) transformed to the frequency domain.

This method can benefit from the fast computation of the overall discrete spectrum using the Fast Fourier Transform (FFT) because the full sequence of interpolated fields over the observer time mesh is known. Thus, the time-domain method may have advantages in terms of computational time. This was pointed out in [40], however without a detailed analysis and verification.

In spite of the high computational cost, the frequency-domain method can be efficient in terms of memory cost when a large number of particles needs to be considered [40]. In this scenario, it may not be possible nor necessary to store the full history of all particle trajectories and the calculation of radiation spectra can be done in parallel to the calculation of particle trajectories in a simulation. To study real-world cases where a large number of particles and observation points are usually needed, the use of high-performance computers may be necessary [38, 39].

For the time-domain method, the superposition of particle fields relies on the interpolation onto a predefined uniform (advanced) time mesh. This implies that the particle information in some previous steps is needed when determining the superimposed field. In the implementation proposed by [40], the full history of the particle trajectories is available beforehand. Under this assumption, the computation of the particle trajectories is completely detached from the computation of the radiation field. However, for a simulation with a large number of particles, the storage of their trajectories needs an excessive memory capacity and such simulation may not be possible in practice. To mitigate the memory consumption for storing the particle trajectories, an algorithm which solves the particle trajectory and computes the superimposed field simultaneously might be necessary. One approach belonging to this category can be found in [41]. The particle trajectories are stored only for a certain number of preceding time steps in a so-called "ring buffer" [41] and used to interpolate the field onto a pre-defined uniform time grid. The drawback of this approach is that typically more field evaluations are performed (and stored) than are necessary for the interpolation.

2.2 Radiation Calculation using Liénard-Wiechert Fields

The radiation field that is emitted from a charged particle at position $\boldsymbol{x}'(t_r)$ at the (retarded) time t_r and observed at a fixed position $\boldsymbol{x}$ at time t (see Figure 2.1) is given by the equations

$$\boldsymbol{E}(\boldsymbol{x},t) = \frac{1}{4\pi\varepsilon_0}\cdot\frac{q\boldsymbol{n}\times\Big((\boldsymbol{n}-\boldsymbol{\beta}'(t_r))\times\dot{\boldsymbol{\beta}}'(t_r)\Big)}{c_0(1-\boldsymbol{n}\cdot\boldsymbol{\beta}'(t_r))^3\|\boldsymbol{x}-\boldsymbol{x}'(t_r)\|_2}, \quad \boldsymbol{B}(\boldsymbol{x},t) = \frac{\boldsymbol{n}}{c_0}\times\boldsymbol{E}(\boldsymbol{x},t), \quad (2.2.1)$$

where $\boldsymbol{\beta}' := \boldsymbol{x}'/c_0$ and $\dot{\boldsymbol{\beta}}' := \ddot{\boldsymbol{x}}'/c_0$ denote the particle's velocity and acceleration (normalized by the speed of light c_0 in vacuum), q is the charge of the particle, ε_0 is the vacuum permittivity, defined by $\varepsilon_0 := 1/(\mu_0 c_0^2)$ with the vacuum permeability μ_0, and $\boldsymbol{n} := (\boldsymbol{x}-\boldsymbol{x}')/\|\boldsymbol{x}-\boldsymbol{x}'\|_2$. (Compared to [42, Equations (14.13)+(14.14)], the velocity field (14.14) can be neglected in the radiation problem since the total power of the velocity field decays with the distance.) The observation time t and (retarded) emission time t_r fulfill the retardation condition (see Figure 2.1)

$$t = t_r + \frac{\|\boldsymbol{x}-\boldsymbol{x}'(t_r)\|_2}{c_0}, \quad (2.2.2)$$

i.e., the radiation field generated at position $x'(t_r)$ at time t_r travels with the speed of light c_0 to reach the observation point $\boldsymbol{x}$ at $t = t_r + \|\boldsymbol{x}-\boldsymbol{x}'(t_r)\|_2/c_0$. There are some drawbacks when applying (2.2.1) and (2.2.2) to compute the radiation field:

- Root finding is needed to solve the retardation condition (2.2.2) for t_r which is computationally intensive [43]. (one can prove that only one root t_r can exist in the retardation condition, cf. Section A.2)
- The trajectories of all particles have to be stored since they are required to compute the radiation field.
- In a numerical simulation, the electron trajectories are computed at discrete time points and interpolation is needed when t_r falls between two consecutive time points [43].

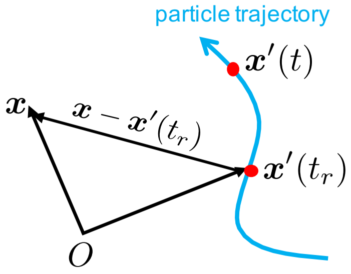
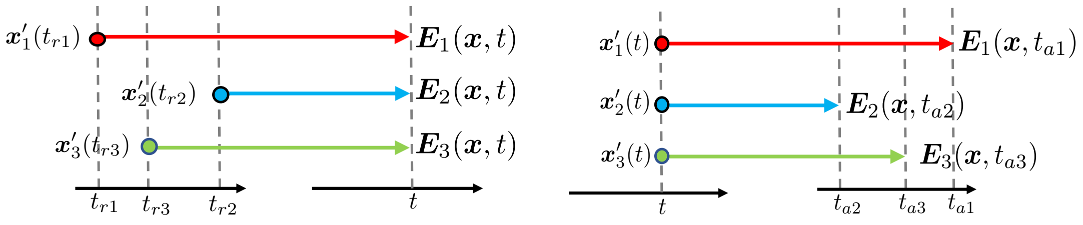


Figure 2.1: Illustration of the retardation condition (2.2.2). Radiation emitted from a particle at time t_r at position $\boldsymbol{x}'(t_r)$ will arrive at the observer at position $\boldsymbol{x}$ at time t. The travel time of the radiation is $\|\boldsymbol{x}-\boldsymbol{x}'(t_r)\|_2/c_0$.

Alternatively, one may evaluate (2.2.1) at a future (advanced) time t_a,

$$t_a = t + \frac{\|\boldsymbol{x}-\boldsymbol{x}'(t)\|_2}{c_0}, \quad (2.2.3)$$

to obtain the field generated by an electron's motion at the current time t, using the substitutions $t \to t_a$ and $t_r \to t$. This scheme does not involve root-finding and we can compute the time t_a^k at which the radiation from particle k arrives at the observation position $\boldsymbol{x}$. However, as implied by (2.2.3), the arrival time of the radiation can be different for different particles and the superposition of particle fields is not straightforward. Both schemes are illustrated in Figure 2.2.



(a) retarted time scheme

(b) advanced time scheme

Figure 2.2: Illustration of (a) retarded time scheme and (b) advanced time scheme. The emission of radiation from three particles is considered and represented by different colors. To compute the field arriving at the observer $\boldsymbol{x}$ at time t from each particle in the retarded time scheme, we need to solve (2.2.2) for the retarded times t_{r1}, t_{r2} and t_{r3}. In the advanced time scheme, the particle trajectory at time t is used to compute the fields at advanced times t_{a1}, t_{a2} and t_{a3}, obtained by evaluating (2.2.3) at which the observer at $\boldsymbol{x}$ receives the radiation field.

When studying radiation phenomena, it is usually valid to consider the far-field approximation, that is the radiation field from a particle which is observed at a distance far from its position $\|\boldsymbol{x}\|_2 \gg \|\boldsymbol{x}'(t)\|_2$, we have

$$\|\boldsymbol{x} - \boldsymbol{x}'(t)\|_2 \approx \|\boldsymbol{x}\|_2 - \boldsymbol{n}\cdot\boldsymbol{x}'(t)$$

where $\boldsymbol{n} := \boldsymbol{x}/\|\boldsymbol{x}\|_2$, by abuse of notation, denotes an approximation of the normal vector $\boldsymbol{n} = (\boldsymbol{x} - \boldsymbol{x}'(t))/\|\boldsymbol{x} - \boldsymbol{x}'(t)\|_2$. The electric field and corresponding advanced time condition can be approximated accordingly,

$$\boldsymbol{E}(\boldsymbol{x}, t_a) \approx \frac{1}{4\pi\varepsilon_0}\cdot\frac{q\boldsymbol{n}\times\Big((\boldsymbol{n} - \boldsymbol{\beta}'(t))\times\dot{\boldsymbol{\beta}}'(t)\Big)}{c_0(1-\boldsymbol{n}\cdot\boldsymbol{\beta}'(t))^3\|\boldsymbol{x}\|_2}, \tag{2.2.4}$$

for

$$t_a \approx t + \frac{\|\boldsymbol{x}\|_2}{c_0} - \frac{\boldsymbol{n}\cdot\boldsymbol{x}'(t)}{c_0}. \tag{2.2.5}$$

This approximation will be used throughout this study.

2.3 Simulation Model

To apply (2.2.4) for the computation of the radiation field, we need to solve the particle's trajectory. The equation of motion of a relativistic charged particle driven by the electromagnetic fields $\boldsymbol{E}$ and $\boldsymbol{B}$ can be written as

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \frac{1}{\gamma}\boldsymbol{p},\\ \frac{d\boldsymbol{p}}{dt} &= \frac{q}{m}(\boldsymbol{E}(\boldsymbol{x},t) + \frac{\boldsymbol{p}}{\gamma}\times\boldsymbol{B}(\boldsymbol{x},t)),\end{aligned} \tag{2.3.1}$$

where m is the mass, $\boldsymbol{p} := \gamma\boldsymbol{\beta}$ is the normalized relativistic momentum and $\gamma = \sqrt{1+\boldsymbol{p}\cdot\boldsymbol{p}}$ is the Lorentz factor of the particle. In the study of accelerator physics, the electromagnetic field interacting with the charged particles can be expressed as

$$\boldsymbol{E} = \boldsymbol{E}_{\text{ext}} + \boldsymbol{E}_{\text{spc}} + \boldsymbol{E}_{\text{rad}} \quad \text{and} \quad \boldsymbol{B} = \boldsymbol{B}_{\text{ext}} + \boldsymbol{B}_{\text{spc}} + \boldsymbol{B}_{\text{rad}}, \tag{2.3.2}$$

which includes the following sources of fields:

1. the external field (subscript "ext") which is the ICS laser field throughout the studies in this thesis,
2. the space-charge field (subscript "spc") which stems from the collective repulsion force of all other charged particles,
3. the radiation field (subscript "rad") generated by all other charged particles.

Throughout the studies in this thesis, the interaction of the radiation field on the charged particle will be neglected because the particle radiation field is very small compared to the ICS laser field. In this chapter, we first neglect the space-charge effect and only consider external field (*i.e.*, $\boldsymbol{E} = \boldsymbol{E}_{\text{ext}}$ and $\boldsymbol{B} = \boldsymbol{B}_{\text{ext}}$) in our the simulation model. This assumption is reasonable for the simulation of most of the ICS light sources because of the short interaction time of the electron beam with a counter-propagating laser. The space-charge force cannot develop significant changes in the particle distribution during such a short time. Note that the space-charge effect might be important in some special scenarios, *e.g.*, a particle beam with micro bunches. In this case, we have coherent emission of radiation from micro bunches. This micro bunch structure might be sensitive to the space-charge effect and can be distorted during the ICS process. Thus, it is also valuable to consider a simulation model with the space-charge effect. We will discuss this model in Chapter 6.

The equation of motion of a charged particle is numerically integrated by the Boris method [44]. There are several variants of the Boris method. In this study, we adapt the following 5-step explicit scheme [45]

$$\begin{aligned}
\boldsymbol{x}^{n+1/2} &= \boldsymbol{x}^n + \frac{\boldsymbol{p}^n}{\gamma^n}\frac{\Delta t}{2}, \\
\boldsymbol{p}^- &= \boldsymbol{p}^n + \frac{\Delta t}{2}\frac{q}{m}\boldsymbol{E}(\boldsymbol{x}^{n+1/2}, t^{n+1/2}), \\
\boldsymbol{p}^+ &= \boldsymbol{p}^- + (\boldsymbol{p}^- + (\boldsymbol{p}^- \times \boldsymbol{t})) \times \boldsymbol{s}, \\
\boldsymbol{p}^{n+1} &= \boldsymbol{p}^+ + \frac{\Delta t}{2}\frac{q}{m}\boldsymbol{E}(\boldsymbol{x}^{n+1/2}, t^{n+1/2}), \\
\boldsymbol{x}^{n+1} &= \boldsymbol{x}^{n+1/2} + \frac{\boldsymbol{p}^{n+1}}{\gamma^{n+1}}\frac{\Delta t}{2}.
\end{aligned} \tag{2.3.3}$$

with the auxiliary quantities

$$\boldsymbol{t} := \frac{q\boldsymbol{B}(\boldsymbol{x}^{n+1/2}, t^{n+1/2})\Delta t}{2m\gamma^-}, \quad \boldsymbol{s} := \frac{2\boldsymbol{t}}{1+t^2} \quad \text{and} \quad \gamma^- = \sqrt{1 + \boldsymbol{p}^- \cdot \boldsymbol{p}^-}.$$

Here, the superscripts n and $n + 1/2$ denote the indices of the integer time-step and half time-step, respectively.

2.4 Frequency-Domain Method

One way to avoid the root-finding problem mentioned in Section 2.2 is to superimpose the fields at the observer in the frequency-domain. In a typical radiation study, one is interested in the spectral-angular distribution of the radiation [42, Eqn. (14.60)]

$$\frac{dI^2}{d\omega d\Omega} = \frac{2}{c_0\mu_0}\|\boldsymbol{x}\|_2^2\|\boldsymbol{E}(\boldsymbol{x}, \omega)\|_2^2,$$

where ω is the frequency and Ω is the solid angle of the observation sphere surface with radius $\|\boldsymbol{x}\|_2$. The electric field $E(\boldsymbol{x}, \omega)$ in the frequency-domain is

$$\boldsymbol{E}(\boldsymbol{x}, \omega) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \boldsymbol{E}(\boldsymbol{x}, t) \exp(j\omega t)\, dt \tag{2.4.1}$$

where $j := \sqrt{-1}$.

By the superposition principle, the radiation spectral-angular distribution from a bunch of electrons is given by

$$\begin{aligned} \frac{dI^2}{d\omega d\Omega} &= \frac{2}{c_0\mu_0} \|\boldsymbol{x}\|_2^2 \left\| \sum_{k=1}^{N_p} \boldsymbol{E}_k(\boldsymbol{x}, \omega) \right\|_2^2 \\ &\overset{(2.4.1)}{=} \frac{2}{c_0\mu_0} \|\boldsymbol{x}\|_2^2 \left\| \sum_{k=1}^{N_p} \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \boldsymbol{E}_k(\boldsymbol{x}, t_a) \exp(j\omega t_a)\, dt_a \right\|_2^2 . \end{aligned} \tag{2.4.2}$$

Substitution of (2.2.4) into (2.4.2), using the approximately constant observation direction $\boldsymbol{n} = \boldsymbol{x}/\|\boldsymbol{x}\|_2$, applying the variable transformation given in (2.2.5) (with $dt_a = (1 - \boldsymbol{n} \cdot \boldsymbol{\beta}'_k(t))\, dt$ by normalization of $\boldsymbol{\beta}'_k = \dot{\boldsymbol{x}}'_k/c_0$), and setting $\varepsilon_0 = (\mu_0 c_0^2)^{-1}$ in (2.4.2) results in [42, Eqn. (14.65)]

$$\begin{aligned} &\frac{dI^2}{d\omega d\Omega}(\boldsymbol{n}, \omega) \\ \approx &\frac{q^2}{16\pi^3\varepsilon_0 c_0} \left\| \sum_{k=1}^{N_p} \int_{-\infty}^{\infty} \underbrace{\frac{\boldsymbol{n} \times \Big((\boldsymbol{n} - \boldsymbol{\beta}'_k(t)) \times \dot{\boldsymbol{\beta}}'_k(t) \Big)}{(1 - \boldsymbol{n} \cdot \boldsymbol{\beta}'_k(t))^2}}_{:=\boldsymbol{\mathcal{A}}_k(\boldsymbol{n},t)} \frac{1}{(1 - \boldsymbol{n} \cdot \boldsymbol{\beta}'_k(t))} \exp(j\omega t_a)\, dt_a \right\|_2^2 \\ = &\frac{q^2}{16\pi^3\varepsilon_0 c_0} \left\| \sum_{k=1}^{N_p} \int_{-\infty}^{\infty} \boldsymbol{\mathcal{A}}_k(\boldsymbol{n}, t) \exp\left(j\omega \left(t - \frac{\boldsymbol{n} \cdot \boldsymbol{x}'_k(t)}{c_0} \right) \right) dt \right\|_2^2 \\ &\text{with} \quad \boldsymbol{\mathcal{A}}_k(\boldsymbol{n}, t) := \frac{\boldsymbol{n} \times \Big((\boldsymbol{n} - \boldsymbol{\beta}'_k(t)) \times \dot{\boldsymbol{\beta}}'_k(t) \Big)}{(1 - \boldsymbol{n} \cdot \boldsymbol{\beta}'_k(t))^2}. \end{aligned} \tag{2.4.3}$$

In (2.4.3), it is the change of variables from t_a to t that circumvents the difficulty of superimposing the electrons' fields at asynchronous future time points. We have also dropped the common constant phase term $\exp(j\omega\|\boldsymbol{x}\|_2/c_0)$ in (2.4.3) as it has no impact on the overall amplitude. The time window of the simulation, *i.e.*, when the acceleration of the charged particle by the driving field is nonzero, is chosen as $[0, t_{\text{end}}]$. We use an equidistant discretization in time with N_s intervals, $N_s \Delta t = t_{\text{end}}$, leading to

$$\frac{dI^2}{d\omega d\Omega}(\boldsymbol{n}, \omega) \approx \frac{q^2(\Delta t)^2}{16\pi^3\varepsilon_0 c_0} \left\| \sum_{k=1}^{N_p} \sum_{i=0}^{N_s} \boldsymbol{\mathcal{A}}_k(\boldsymbol{n}, i\Delta t) \exp\left(j\omega \left(i\Delta t - \frac{\boldsymbol{n} \cdot \boldsymbol{x}'_k(i\Delta t)}{c_0} \right) \right) \right\|_2^2 . \tag{2.4.4}$$

Introducing the abbreviations

$$\boldsymbol{\mathcal{E}}_{\boldsymbol{k},\boldsymbol{i}}(\boldsymbol{n}, \omega) := \boldsymbol{\mathcal{A}}_k(\boldsymbol{n}, i\Delta t) \exp\left(j\omega \left(i\Delta t - \frac{\boldsymbol{n} \cdot \boldsymbol{x}'_k(i\Delta t)}{c_0} \right) \right), \tag{2.4.5}$$

$$\boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n}, \omega) := \sum_{k=1}^{N_p} \boldsymbol{\mathcal{E}}_{k,i}(\boldsymbol{n}, \omega), \qquad \boldsymbol{\mathcal{E}}^{\text{sum}}(\boldsymbol{n}, \omega) := \sum_{i=1}^{N_s} \boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n}, \omega),$$

and exchanging the order of summation w. r. t. k and i in (2.4.4), we obtain

$$\frac{dI^2}{d\omega d\Omega}(\boldsymbol{n},\omega) \approx \frac{q^2(\Delta t)^2}{16\pi^3\varepsilon_0 c_0}\left\|\sum_{i=1}^{N_s}\sum_{k=1}^{N_p}\boldsymbol{\mathcal{E}}_{k,i}\right\|_2^2 = \frac{q^2(\Delta t)^2}{16\pi^3\varepsilon_0 c_0}\left\|\sum_{i=1}^{N_s}\boldsymbol{\mathcal{E}}_i^{\text{bunch}}\right\|_2^2 = \frac{q^2(\Delta t)^2}{16\pi^3\varepsilon_0 c_0}\left\|\boldsymbol{\mathcal{E}}^{\text{sum}}\right\|_2^2 .$$

A pseudocode for the frequency-domain method is provided in Algorithm 2.1.

Algorithm 2.1: Frequency-Domain Method

```
Initialization:
    $\boldsymbol{\mathcal{E}}^{\text{sum}}(\boldsymbol{n},\omega) \leftarrow 0$
for simulation step $i$ do
    $\boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n},\omega) \leftarrow 0$
    for particle $k$ do
        update trajectory
        $\boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n},\omega) \leftarrow \boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n},\omega) + \boldsymbol{\mathcal{E}}_{k,i}(\boldsymbol{n},\omega)$
    end
    $\boldsymbol{\mathcal{E}}^{\text{sum}}(\boldsymbol{n},\omega) \leftarrow \boldsymbol{\mathcal{E}}^{\text{sum}}(\boldsymbol{n},\omega) + \boldsymbol{\mathcal{E}}_i^{\text{bunch}}(\boldsymbol{n},\omega)$
end
```

2.5 Time-Domain Method

To start our discussion, we first present the result of a test simulation for 10 particles initially at rest with random initial positions which are samples from a Gaussian distribution with mean 0 and standard deviation 1 in each direction. The particles are driven by a sinusoidal electromagnetic plane wave. In the simulation, the equations of motion for charged particles are simulated using the Boris method [44], and the trajectory of each particle is used to evaluate the radiation field arriving at an observation point and the corresponding observation time by evaluating (2.2.4) and (2.2.5).

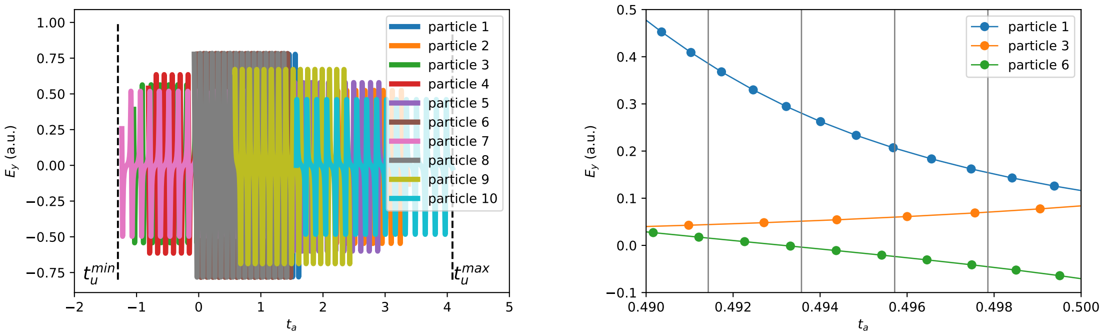
The result of this trial simulation is shown in Figure 2.3. Figure 2.3a shows the E_y component of the radiation field generated by each particle. Figure 2.3b shows a zoom-in view of Figure 2.3a. From the result of this trial simulation, we summarize the following observations:

- The observed pulse duration of the radiation fields generated by the particles is different. This is illustrated in Figure 2.3a and explained by (2.2.5),

$$\int dt_a^k = \int_{t_{\text{start}}}^{t_{\text{end}}} (1 - \boldsymbol{n}\cdot\boldsymbol{\beta}_k'(t))\, dt = t_a^k(t_{\text{end}}) - t_a^k(t_{\text{start}}).$$

  Each particle moves at a different phase of the external electromagnetic wave, and the particle velocities $\boldsymbol{\beta}_k'(t)$ driven by the external field during the simulation can be different. Therefore, the resulting integrals of dt_a^k are different for the particles.
- It is problematic to superimpose the fields of all particles since the radiation fields from the trajectories of different particles evaluated at the same time t are observed at different future times $t_a^k \approx t + \|\boldsymbol{x}\|_2/c_0 - (\boldsymbol{n}\cdot\boldsymbol{x}_k'(t))/c_0$ (2.2.5). This is illustrated in Figure 2.3b.
- To superimpose the radiation fields, we define a uniform time grid $\boldsymbol{t}_u$ (indicated by the gray vertical lines in Figure 2.3b and interpolate the particles' fields to the uniform grid (details on the interpolation will follow in Section 2.5.1).

- To cover the radiation temporal profile of all particles, we need to determine the upper and lower bound (annotated by $t_u^{\max}$ and $t_u^{\min}$ in Figure 2.3a of the uniform time grid $\boldsymbol{t}_u$.



(a) radiation fields of all particles

(b) zoom-in for three selected particles

Figure 2.3: Simulation result for the test simulation. (a) shows the temporal profiles of the radiation fields from all particles. (b) shows the zoom-in view of (a) for three selected particles. Here, the relative arrival time of the particle radiation field is demonstrated instead of the absolute arrival time (*i.e.*, the term $\|\boldsymbol{x}\|_2/c_0$ in (2.2.5) is neglected as it is constant for an observer).

To create a uniform time grid for the radiation field interpolation, we need to know the bound of the radiation temporal profile at different observation positions. For a simulation with N_p particles and N_{obs} observers, we determine the bounds of the uniform (advanced) time grid by

$$t_u^{\min} = \min\left\{t_a^{k,m}(t_{\text{start}}) \mid k = 1, \dots, N_p,\ m = 1, \dots, N_{\text{obs}}\right\}, \tag{2.5.1}$$

$$t_u^{\max} = \max\left\{t_a^{k,m}(t_{\text{end}}) \mid k = 1, \dots, N_p,\ m = 1, \dots, N_{\text{obs}}\right\}, \tag{2.5.2}$$

where $t_{\text{start}}, t_{\text{end}}$ denote the start/end of the simulation and

$$t_a^{k,m}(t) \approx t + \frac{\|\boldsymbol{x}_m\|_2}{c_0} - \frac{\boldsymbol{n}_m \cdot \boldsymbol{x}'_k(t)}{c_0}.$$

($\|\boldsymbol{x}_m\|_2$ is the same for all m observers since they are located on a spherical surface.) The lower bound $t_u^{\min}$ can be easily determined at the beginning of the simulation since we already have the initial positions of all particles. To determine the upper bound $t_u^{\max}$, it might be possible to perform a trial simulation in which we only compute the particles' trajectories and evaluate $t_u^{\max}$ by (2.5.2) at the end of the trial simulation. However, this will become costly when N_p is large. To avoid such a trial simulation, we determine $t_u^{\max}$ by estimating the total radiation pulse duration from a particle bunch as follows.

As can be observed from Figure 2.3a, the total radiation pulse duration from all particles is influenced by the radiation pulse durations of single particles and the different lags in their arrival times. For the ICS problem, the radiation pulse duration from a single particle

$$T_{\text{rad}} \equiv \int dt_a = \int_0^{T_{\text{laser}}} \left(1 - \boldsymbol{n} \cdot \boldsymbol{\beta}'(t)\right)\, dt,$$

can be approximated for highly relativistic particles by

$$T_{\text{rad}} \approx \frac{T_{\text{laser}}}{2\gamma^2} \tag{2.5.3}$$

where $T_{\text{laser}} = t_{\text{end}} - t_{\text{start}}$ is the pulse duration of the counter-propagating laser pulse. The maximum difference in arrival times of the radiation pulses from different particles is caused by their distribution in space and can be estimated by the last term of (2.2.5),

$$T_{\text{rad}}^{\text{bunch}} := \max_{\theta_x,\theta_y} \max_{i,j=1,\ldots,N_p} \left| \frac{\boldsymbol{n} \cdot (\boldsymbol{x}'_i - \boldsymbol{x}'_j)}{c_0} \right| \tag{2.5.4}$$

with

$$\boldsymbol{n} := (\sin\theta_x, \cos\theta_x \sin\theta_y, \cos\theta_x \cos\theta_y).$$

Here, θ_x and θ_y are the angles defining the observers' positions on the sphere (see Figure 2.4). In the highly relativistic scenario, the opening angle of the radiation from a particle is of the order of $1/\gamma \ll 1$. We hence use approximations $\sin\theta \approx \theta$, $\cos\theta \approx 1$, and (2.5.4) becomes

$$T_{\text{rad}}^{\text{bunch}} \approx \max_{\substack{i,j=1,\ldots,N_p \\ |\theta_x|,\,|\theta_y|<1/\gamma}} \left| \frac{\theta_x(x'_i - x'_j) + \theta_y(y'_i - y'_j) + (z'_i - z'_j)}{c_0} \right|,$$

which has the upper bound

$$T_{\text{rad}}^{\text{bunch}} \lessapprox \frac{l_{b,x}}{\gamma c_0} + \frac{l_{b,y}}{\gamma c_0} + \frac{l_{b,z}}{c_0}, \tag{2.5.5}$$

where $l_{b,x}$, $l_{b,y}$ and $l_{b,z}$ denote the size of the particle bunch in each direction at t_{start}.

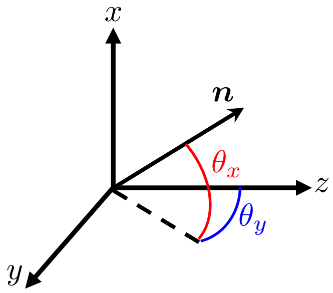


Figure 2.4: Definition of the normal vector $\boldsymbol{n}$ pointing from the particle to the observer. θ_x is the angle between $\boldsymbol{n}$ and its projection to the y-z plane. θ_y is the angle between the z-axis and the projection of $\boldsymbol{n}$ to the z-y plane.

Thus, to cover the total radiation pulse duration from all particles, the length of the required uniform time grid T_u can be approximately chosen as

$$T_u \approx T_{\text{rad}}^{\text{bunch}} + T_{\text{rad}} \approx \frac{l_{b,x}}{\gamma c_0} + \frac{l_{b,y}}{\gamma c_0} + \frac{l_{b,z}}{c_0} + \frac{T_{\text{laser}}}{2\gamma^2}. \tag{2.5.6}$$

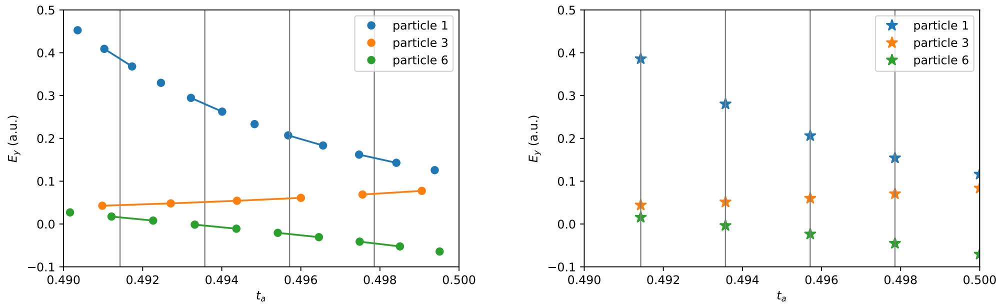
Table 2.1 lists the values for T_u computed from (2.5.6) for different parameter settings. We also provide values for T_u from a trial simulation to verify the theoretical estimates. Once $t_u^{\min}$ and T_u are known, $t_u^{\max} = t_u^{\min} + T_u$ can be immediately determined.

$l_{b,x}$	$l_{b,y}$	$l_{b,z}$	T_{laser}	γ	$T_{\text{rad}}^{\text{bunch}}$	T_{rad}	T_u (theory)	T_u (simulation)
10	10	10	10	40	35.0000	0.003125	35.0031	35.0211
20	10	10	10	40	35.8333	0.003125	35.8365	35.8876
10	20	10	10	40	35.8333	0.003125	35.8365	35.8540
10	10	20	10	40	68.3333	0.003125	68.3365	68.3328
10	10	10	20	40	35.0000	0.006250	35.0063	35.0310

Table 2.1: Theoretical estimation of T_u by (2.5.6) for different parameter settings in comparison with results from trial simulations for the particles' trajectories using (2.5.1) and (2.5.2). The quantities $T_{\text{rad}}^{\text{bunch}}$ and T_{rad} are computed by (2.5.5) and (2.5.3), respectively. Here, all length and time quantities are normalized to $2\pi/\lambda_0$ and $2\pi c_0/\lambda_0$, respectively, with $\lambda_0 = 1000\,\mu\text{m}$.

2.5.1 Interpolation

In the previous subsection, we discussed how to choose the bounds of a uniform time grid in order to determine the temporal positions for the computation of the total radiation field. Next, we interpolate the particles' fields at different temporal (advanced) positions to the uniform grid by piecewise linear interpolation.



(a) local linear interpolation

(b) resulting grid values

Figure 2.5: Linear interpolation scheme for the temporal radiation profile. The gray vertical lines mark the position of the uniform time grid points. The round points in (a) connected by line segments are used for interpolation. (b) shows the result after interpolation. The stars indicate the values of the interpolated field.

To apply linear interpolation, one needs the arrival times and corresponding radiation fields from each particle at the previous and current time steps. This might suggest that we need two memory buffers for storing the field information of adjacent time steps. However, we actually only need one memory buffer to store the field information of the previous step. In every time step, for each particle, we use the field computed at the current time step and the field stored in the memory buffer (*i.e.*, field computed at the previous time step) to compute the values at the grid points with linear interpolation. After that, the interpolated field at a time grid point is directly added to the value of the corresponding total radiation field at the same grid point. The complete procedure is described in Algorithm 2.2 (for a single observation direction $\boldsymbol{n}$).

Algorithm 2.2: Proposed Time-Domain Method

```
Initialization:
    $\boldsymbol{E}_{tot}(\boldsymbol{n}, t_u) \leftarrow 0$ (total field from all particles over a uniform grid with points $t_u$)
    $t_b^k(\boldsymbol{n}) \leftarrow 0$ (buffer for storing the field's arrival time of particle $k$ computed at the previous step)
    $\boldsymbol{E}_b^k(\boldsymbol{n}) \leftarrow 0$ (buffer for storing the arrival field of particle $k$ computed at previous time step)
for simulation step $i$ do
    for particle $k$ do
        update trajectory
        compute $t_a^k(\boldsymbol{n})$ and $E^k(\boldsymbol{n}, t_a^k)$
        for $t_u$ : $t_b^k(\boldsymbol{n}) < t_u \le t_a^k(\boldsymbol{n})$ do
            $\boldsymbol{E}_{tot}(\boldsymbol{n}, t_u) \leftarrow \boldsymbol{E}_{tot}(\boldsymbol{n}, t_u) + E_b^k(\boldsymbol{n}) \frac{t_a^k(\boldsymbol{n}) - t_u}{t_a^k(\boldsymbol{n}) - t_b^k(\boldsymbol{n})} + \boldsymbol{E}^k(\boldsymbol{n}, t_a^k) \frac{t_u - t_b^k(\boldsymbol{n})}{t_a^k(\boldsymbol{n}) - t_b^k(\boldsymbol{n})}$
        end
    end
    $t_b^k(\boldsymbol{n}) \leftarrow t_a^k(\boldsymbol{n})$
    $\boldsymbol{E}_b^k(\boldsymbol{n}) \leftarrow \boldsymbol{E}^k(\boldsymbol{n}, t_a^k)$
end
```

2.5.2 Parallelization

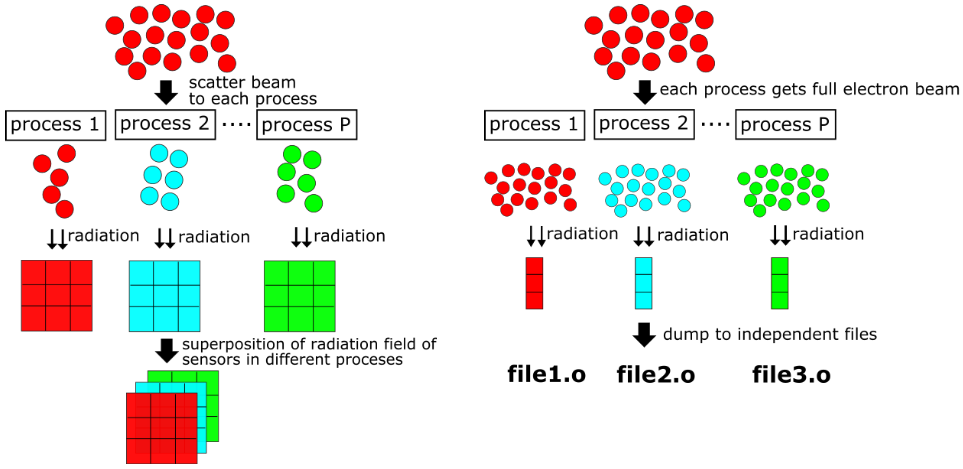
Before discussing parallelization, we briefly introduce key data structures used in the code development. Here, the memory cost is measured by the number of floating-point numbers to be stored.

Beam: A structure to store the state of motion (position, momentum, velocity and acceleration) for a bunch of particles at a specific time. For a particle bunch with N_p particles in three-dimensional space, $12N_p$ floating-point numbers are required.

Sensor: A sensor records the total radiation field from a bunch of particles over the uniform time grid. For N_{T_u} uniform time grid points, $3N_{T_u}$ floating-point numbers are required. In addition, each particle's radiation arrival time and radiation field are also stored in a memory buffer which requires $4N_p$ floating-point numbers.

Detector: A detector contains N_{obs} sensors and each sensor has a different observation position. The memory cost for the detector is thus $N_{\text{obs}}(4N_p + 3N_{T_u})$ floating-point numbers.

There are at least two possible parallelization schemes for the simulation, see Figure 2.6. One is beam parallelization in which the particle bunch is divided into several small bunches and each small bunch is assigned to an MPI process. Each process creates its own detector object which receives the radiation from a small bunch. At the end of the simulation, the radiation data in each detector object is superimposed to the radiation data in the detector object created by the master task. The other possible scheme is detector parallelization where a global sensor in the detector is divided into several local detectors. In the beginning of the simulation, each process creates a copy of the entire electron bunch and a local detector consisting of a subset of sensors from the global detector. During the simulation, the local detector receives the radiation from the copy of the entire electron bunch owned by each process. At the end of the simulation, each process dumps the data from its local detector separately.



(a) beam parallelization

(b) detector parallelization

Figure 2.6: Two possible parallelization schemes: (a) beam parallelization and (b) detector parallelization.

The simulation of ICS can involve a large number of particles and sensors. Therefore, the memory requirement per process for both parallelization schemes is an important consideration. For beam parallelization, the memory cost per process is

$$M_{\text{beam}}(P, N_p, N_{\text{obs}}, N_{T_u}) = 12\tfrac{N_p}{P} + N_{\text{obs}}\left(4\tfrac{N_p}{P} + 3N_{T_u}\right)$$

where P is the number of computer processes. For detector parallelization, the memory cost per process is

$$M_{\text{detector}}(P, N_p, N_{\text{obs}}, N_{T_u}) = 12N_p + \tfrac{N_{\text{obs}}}{P}\left(4N_p + 3N_{T_u}\right).$$

In order to determine which scheme has lower memory cost, we define the function

$$\begin{aligned}\phi(P, N_p, N_{\text{obs}}, N_{T_u}) &:= \frac{M_{\text{beam}}(P, N_p, N_{\text{obs}}, N_{T_u})}{M_{\text{detector}}(P, N_p, N_{\text{obs}}, N_{T_u})} = \frac{12\frac{N_p}{P} + N_{\text{obs}}\left(4\frac{N_p}{P} + 3N_{T_u}\right)}{12N_p + \frac{N_{\text{obs}}}{P}\left(4N_p + 3N_{T_u}\right)} \\ &= 1 + \frac{3\left(1 - \frac{1}{P}\right)\left(N_{\text{obs}}N_{T_u} - 4N_p\right)}{12N_p + \frac{N_{\text{obs}}}{P}\left(4N_p + 3N_{T_u}\right)}.\end{aligned} \tag{2.5.7}$$

If $\phi > 1$, detector parallelization has a lower memory footprint and otherwise beam parallelization. Since the factor $(1 - 1/P)$ in the numerator and all terms in the denominator of the second term are positive, we conclude that

$$\phi(P, N_p, N_{\text{obs}}, N_{T_u}) \quad \begin{cases} \geq 1 & \text{if } N_{\text{obs}}N_{T_u} - 4N_p \geq 0, \\ < 1 & \text{else.} \end{cases} \tag{2.5.8}$$

Therefore, the ratio $N_{\text{obs}}N_{T_u}/(4N_p)$ determines which parallelization scheme is more favorable in terms of memory consumption.

The number of particles N_p can be computed from the total charge of the electron bunch. The number of uniform grid points N_{T_u} should be chosen according to the Nyquist theorem

$$N_{T_u} = 2\frac{\omega_{\max}}{\Delta\omega}. \tag{2.5.9}$$

Here, $\omega_{\max}$ is the maximum radiation angular-frequency we want to observe which can be estimated by

$$\omega_{\max} = 4\gamma^2\omega_0 \tag{2.5.10}$$

for a typical ICS problem where ω_0 is the angular-frequency of the counter-propagating laser. The resolution for the angular-frequency is determined by the duration of the total radiation pulse from particles

$$\Delta\omega = \frac{2\pi}{T_u}. \tag{2.5.11}$$

Combining (2.5.9), (2.5.10), (2.5.11) and (2.5.6), we can estimate

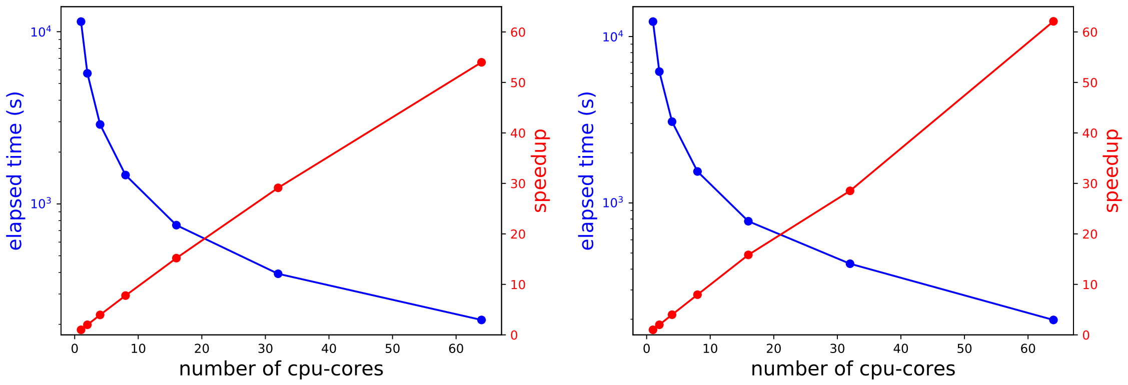
$$N_{T_u} \approx 8\gamma^2 \left(\frac{l_{b,x}}{\gamma\lambda_0} + \frac{l_{b,y}}{\gamma\lambda_0} + \frac{l_{b,z}}{\lambda_0} + \frac{c_0 T_{\text{laser}}}{2\gamma^2\lambda_0} \right)$$

where the identity $\omega_0 = 2\pi c_0/\lambda_0$ is used.

In Table 2.2, we use parameters from different experimental projects of ICS sources to compute the ratio $4N_p/N_{T_u}$ which is the number of observers where both parallelization schemes break even ($\phi = 1$). Hence, according to (2.5.8), for a larger number of observers, detector parallelization is preferable whereas a smaller number of observers should be computed with beam parallelization. In general, we need at least a few thousand observation angles (*i.e.*, $N_{\text{obs}} > 1000$) to sufficiently resolve the radiation angular distribution. Therefore, detector parallelization is more favorable and is thus implemented in the solver.

2.5.3 Implementation

We use C++ and MPI to implement the time-domain algorithm and the detector parallelization scheme described in this section. The parallel performance for the code is demonstrated in Figure 2.7a. For comparison, the parallel performance of our implementation of the frequency-domain method is also given in Figure 2.7b. A parallelized post-processing code written in Julia [46] and MPI.jl is used to transform the time-domain field to the radiation spectra. The two solvers can be accessed in the repository ICSSpectra.



(a) Time-domain method

(b) Frequency-domain method

Figure 2.7: Parallel performance for: (a) proposed time-domain simulation with $N_p = 10^5$, $N_{\text{obs}} = 1024$ and $N_{T_u} = 10^4$; (b) frequency-domain simulation with $N_p = 1024$, $N_{\text{obs}} = 64$ and $N_\omega = 500$.

project name	beam energy (MeV)	l_z (µm)	charge (pC)	T_{laser} (ps)	λ_0 (µm)	$4N_p/N_{T_u}$
AXSIS [24]	19.5	2.9	1	1	1	714.3
ODU CLS [47, 48]	25	382	10	1.57	1	34.2
ASU CXFEL [27]	35	1.5	1	1.5	1.03	443.5
ASU CXLS [23]	40	147	100	3.0	1.03	357.1
XFI [28]	65.9	3	10	1.7	0.8	498.8
MuLCS [29]	45	15000	1000	26	1.064	28.6
ThomX [30, 31]	70	6000	1000	11.75	1.03	28.6

Table 2.2: The ratio $4N_p/N_{T_u}$ for different experimental projects. Here, the electron beam transverse sizes l_x and l_y are not considered in the evaluation of N_{T_u} since their contribution is minor for a high-energy electron beam.

2.6 Numerical Comparison of the Time-Domain and Frequency-Domain Methods

2.6.1 Accuracy

To compare the accuracy of the radiation spectra computed by both methods, we measure the component-wise relative error

$$\text{error}(\omega,\theta) := \text{abs}\left(\left.\frac{dI}{d\omega d\Omega}\right|_{\text{simulation}} - \left.\frac{dI}{d\omega d\Omega}\right|_{\text{theory}}\right) / \max\left(\left.\frac{dI}{d\omega d\Omega}\right|_{\text{theory}}\right). \quad (2.6.1)$$

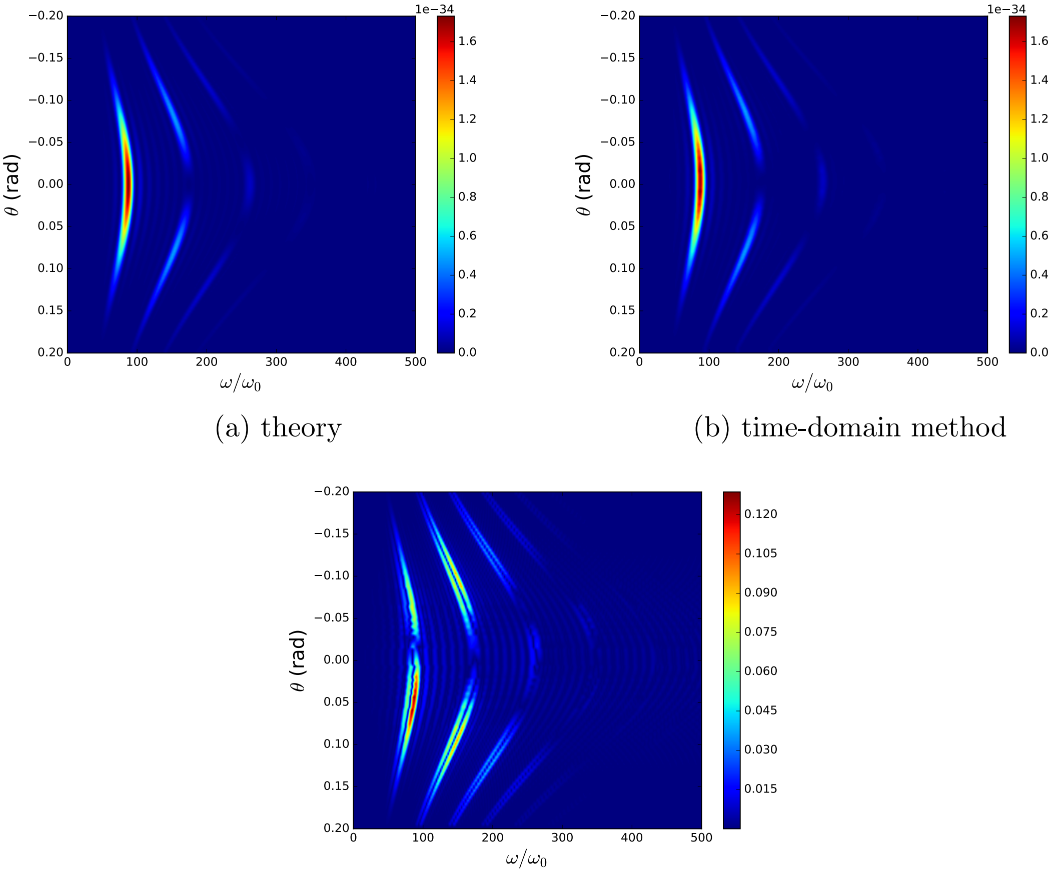
between the theoretical and computed results [39]. The theoretical result is computed by the formula proposed by Esarey et al. [19] which considers the radiation spectral-angular distribution from a single particle interacting with a finite number of periods of a sinusoidal electromagnetic wave.

We perform a single particle simulation in which a particle moves with initial energy $\gamma = 5$ in the $+z$ direction and collides with 7 periods of a counter-propagating sinusoidal wave. The radiation is collected in observation directions in the y-z plane (i. e., $\theta_x = 0$ and $\theta_y = \theta$ in Figure 2.4). The radiation spectral-angular distribution and the corresponding errors computed by (2.6.1) are illustrated in Figure 2.8 for the time-domain method and in Figure 2.9 for the frequency-domain method.

In addition, the maximum and mean values of normalized errors for the radiation spectral-angular distribution with different numbers of frequency points N_ω and different numbers of observation angles N_θ are shown in Table 2.3 and Table 2.4, respectively. The relative error over the full spectral-angular distribution for the time-domain method and frequency-domain method has an asymmetric distribution with respect to $\theta = 0$. This is due to the discretized particle trajectory in the simulation and can be reduced by decreasing the step size for solving the particle trajectory [39]. From Table 2.4 and Table 2.3, we see that both methods achieve an acceptable relative error and that the time-domain method reaches the same level of accuracy as the frequency-domain method by increasing the number of observation points.

2.6.2 Performance

We will now discuss the computational complexity of the two schemes with respect to the following parameters:



(a) theory

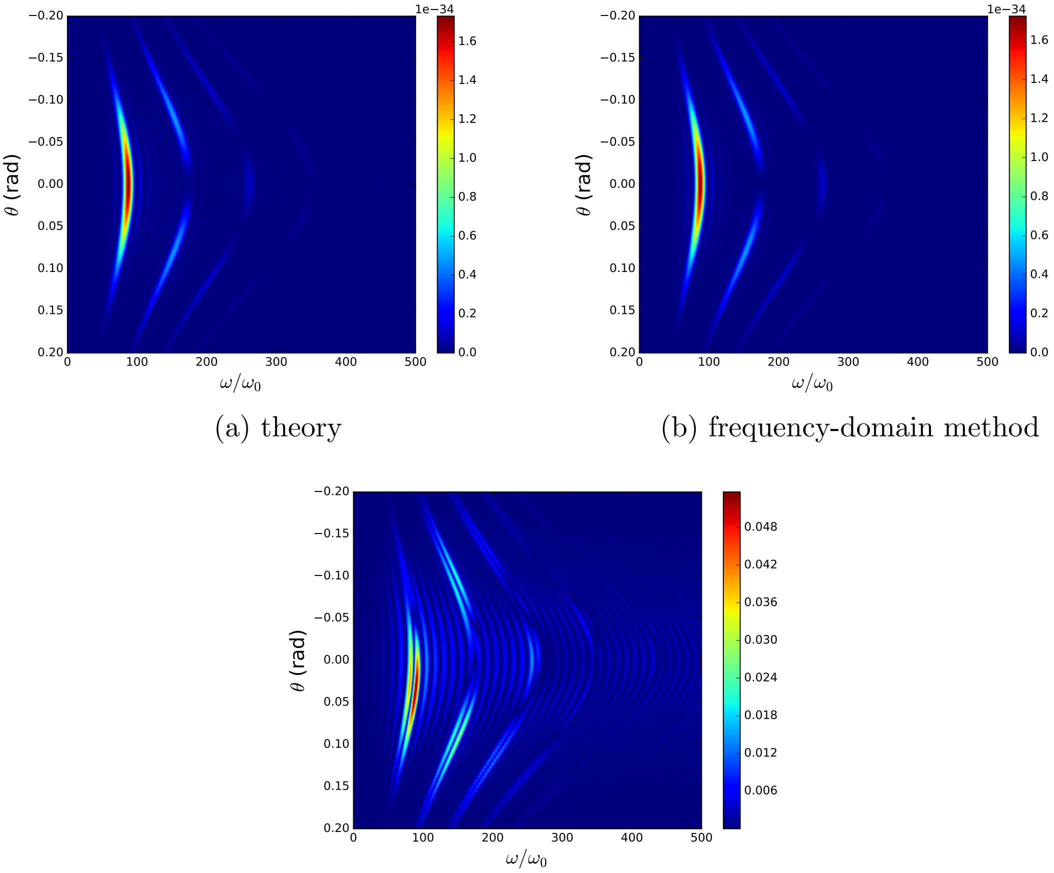
(b) time-domain method

(c) relative error

Figure 2.8: Demonstration of the accuracy of the time-domain method. The radiation spectral-angular distribution is computed by (a) theory and (b) the time-domain method for the configuration stated in Section 2.6.1. The relative error (c) is computed by (2.6.1).

$N_{\text{obs}} = 101$	$N_\omega = 5 \times 10^3$	$N_\omega = 1 \times 10^4$	$N_\omega = 2 \times 10^4$
TDM			
$\max(\text{error}(\omega, \theta))$	1.287×10^{-1}	1.290×10^{-1}	1.291×10^{-1}
$\text{mean}(\text{error}(\omega, \theta))$	3.520×10^{-3}	3.522×10^{-3}	3.524×10^{-3}
FDM			
$\max(\text{error}(\omega, \theta))$	5.366×10^{-2}	5.366×10^{-2}	5.366×10^{-2}
$\text{mean}(\text{error}(\omega, \theta))$	1.589×10^{-3}	1.589×10^{-3}	1.580×10^{-3}

Table 2.3: Normalized relative error of the time-domain method (TDM) and the frequency-domain method (FDM) for different numbers of frequency points.



(a) theory

(b) frequency-domain method

(c) relative error

Figure 2.9: Demonstration of the accuracy of the frequency-domain method. The radiation spectral-angular distribution is computed by (a) theory and (b) the frequency-domain method for the configuration stated in Section 2.6.1. The relative error (c) is computed by (2.6.1).

$N_\omega = 5 \times 10^3$	$N_{\text{obs}} = 101$	$N_{\text{obs}} = 201$	$N_{\text{obs}} = 401$
TDM			
$\max(\text{error}(\omega, \theta))$	1.291×10^{-1}	8.587×10^{-2}	6.609×10^{-2}
$\text{mean}(\text{error}(\omega, \theta))$	3.524×10^{-3}	2.175×10^{-3}	1.547×10^{-3}
FDM			
$\max(\text{error}(\omega, \theta))$	5.366×10^{-2}	5.366×10^{-2}	5.366×10^{-2}
$\text{mean}(\text{error}(\omega, \theta))$	1.580×10^{-3}	1.596×10^{-3}	1.599×10^{-3}

Table 2.4: Normalized relative error of the time-domain method (TDM) and the frequency-domain method (FDM) for different numbers of observation points.

N_s the number of simulation time steps,
N_p the number of particles,
N_{obs} the number of observers,
L_t the operation count to compute the advanced time t_a, see (2.2.5),
L_f the operation count to evaluate the radiation field, see (2.2.4),
L_I the operation count to interpolate the particle field,
N_{T_u} the number of grid points of the uniform time grid,
N_ω the number of grid points of the uniform frequency grid,
L_ω the operation count to evaluate a particle's contribution in the frequency-domain method, see (2.4.5).

The total operation count for the time-domain method is given by

$$\text{OP}_t = \text{const} \cdot N_s \cdot N_p \cdot N_{\text{obs}} \cdot (L_t + L_f + L_I) + \text{const} \cdot N_{\text{obs}} \cdot N_{T_u} \cdot \log N_{T_u}, \quad (2.6.2)$$

whereas for the frequency-domain method we have

$$\text{OP}_\omega = \text{const} \cdot N_s \cdot N_p \cdot N_{\text{obs}} \cdot N_\omega \cdot L_\omega. \quad (2.6.3)$$

The second term in the operation count for the time-domain method stems from an additional post-processing phase in which the radiation data on the uniform time grid is transformed to the frequency-domain by the FFT algorithm in order to obtain the radiation spectra. To easily measure the performance, we define a wall-clock time model for both the time-domain and frequency-domain method. If the memory latency is neglected and the wall-clock time only depends on the operation count of the method, i. e., OP_t and OP_ω, the wall-clock time for the time-domain method can be split into the two parts

$$W_t^1(N_s, N_p, N_{\text{obs}}) = C_t^1 \cdot N_s \cdot N_p \cdot N_{\text{obs}}, \quad W_t^2(N_{\text{obs}}, N_{T_u}) = C_t^2 \cdot N_{\text{obs}} \cdot N_{T_u} \cdot \log N_{T_u},$$

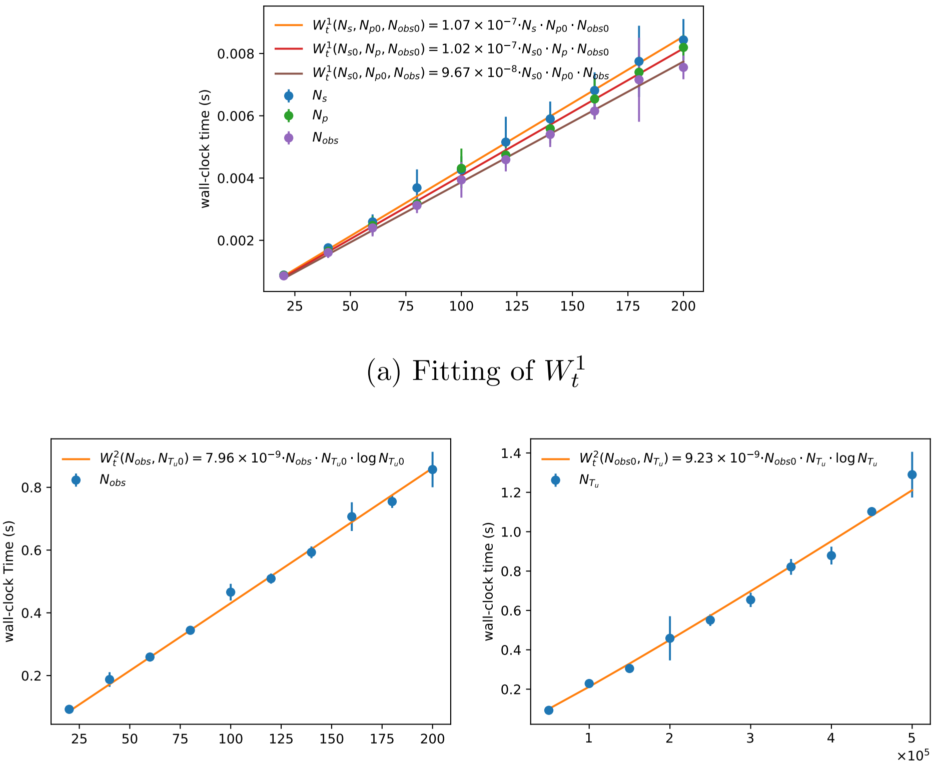
and the wall-clock time for the frequency-domain method is denoted by

$$W_\omega(N_s, N_p, N_{\text{obs}}, N_\omega) = C_\omega \cdot N_s \cdot N_p \cdot N_{\text{obs}} \cdot N_\omega.$$

The leading constants translate the operation count into wall-clock time and depend on the implementation and computer architecture (*e.g.*, implementation details, operating system, hardware, and compiler, which are not our primary concerns in this study). Since our analysis only involves the ratio of the constants, the dependency of the computer architecture at which the simulation is performed will most likely be canceled out. We can determine the leading constants for both models by the regression of several benchmark runs of both methods (Figure 2.10 and Figure 2.11) and obtain

$$C_t^1 = 1.02 \times 10^{-7}, \quad C_t^2 = 8.61 \times 10^{-9}, \quad C_\omega = 1.17 \times 10^{-7} \quad (2.6.4)$$

for our implementation and computer architecture.



(a) Fitting of W_t^1

(b) Fitting of W_t^2

Figure 2.10: Fitting of (a) W_t^1 and (b) W_t^2 by splitting the multi-variate problem to multiple single-variate fitting problems. Here, $N_{s0} = 20$, $N_{p0} = 20$, $N_{obs0} = 20$ and $N_{T_u0} = 5 \times 10^4$. Each data point represents the average wall-clock time of 50 runs. The leading constants $C_t^1 = 1.02 \times 10^{-7}$ and $C_t^2 = 8.61 \times 10^{-9}$ are determined as the averages of all respective single-variate fitting constants.

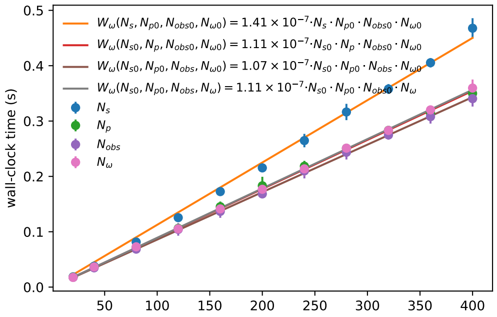


Figure 2.11: Fitting of W_ω by splitting the multi-variate problem to multiple single-variate fitting problems. $N_{s0} = 20$, $N_{p0} = 20$, $N_{obs0} = 20$ and $N_{\omega 0} = 20$. The leading constant $C_\omega = 1.17 \times 10^{-7}$ is determined as the average of all single-variate fitting constants.

The wall-clock time ratio of the time-domain and frequency-domain method is given by

$$\frac{W_t^1 + W_t^2}{W_\omega} = \frac{C_t^1 \cdot N_s \cdot N_p + C_t^2 \cdot N_{T_u} \cdot \log N_{T_u}}{C_\omega \cdot N_s \cdot N_p \cdot N_\omega}. \tag{2.6.5}$$

If we set $N_{T_u} = 2N_\omega$ for the reason of sampling theory, the ratio simplifies to

$$\frac{W_t^1 + W_t^2}{W_\omega} = \frac{C_t^1}{C_\omega}\frac{1}{N_\omega} + \frac{C_t^2}{C_\omega}\frac{2}{N_s N_p}\log 2N_\omega.$$

The ratio is greater than one (*i.e.*, the frequency-domain method outperforms the time-domain method) if

$$N_\omega > \frac{1}{2}\exp\left(\frac{1}{2}\frac{C_\omega}{C_t^2}\cdot N_s \cdot N_p\right),$$

which is typically not satisfied with the parameters from practical scenarios. For a simulation of a 1 pC particle bunch colliding with a 1 µm laser pulse with 1 ps duration, the required simulation parameters are $N_s = 6000$ (0.167 fs per time step) and $N_p = 6.25 \times 10^6$. In this case, using the experimentally fitted constants in (2.6.4), the criterion for the frequency-domain method to outperform the time-domain method is $N_\omega > 0.5\exp(2.5479 \cdot 10^{11}) = \mathcal{O}\left(10^{10^{11}}\right)$.

In the previous discussion, we assumed that the number of time points for the time-domain method is twice the number of frequency points for the frequency-domain method which originated from the assumption that the frequency resolutions for both methods are the same. However, this condition is not necessary in practice. For the time-domain method, the upper limit of $\Delta\omega$ is restrictively determined (due to the sampling theorem) by the total radiation pulse duration. It could be possible that the upper limit of $\Delta\omega$ is far less than the actual resolution that we need to study a problem. For the frequency-domain method, on the other hand, there is no such limitation. The resolution $\Delta\omega$ can be chosen arbitrarily. The bandwidth $\overline{\omega}$ of the maximum frequency which refers to the first harmonic frequency of ICS is [19]

$$\overline{\omega} = \frac{\lambda_0}{c_0 T_\text{laser}}\omega_\text{max}.$$

Thus, when applying the frequency-domain method,

$$\Delta\omega = \frac{1}{N_{\overline{\omega}}}\frac{\lambda_0}{c_0 T_\text{laser}}\omega_\text{max} \quad \text{and} \quad N_\omega = N_{\overline{\omega}}\frac{c_0 T_\text{laser}}{\lambda_0}, \tag{2.6.6}$$

where $N_{\overline{\omega}}$ is the number of grid points needed for $\overline{\omega}$. To find a condition under which the time-domain method is slower than the frequency-domain method, equation (2.6.5) can be expressed as

$$C_t^2 \cdot N_{T_u} \cdot \log N_{T_u} > (C_\omega \cdot N_\omega - C_t^1) N_s \cdot N_p$$

and rewritten by

$$N_{T_u} \cdot \log N_{T_u} > \frac{C_\omega}{C_t^2} N_\omega \cdot N_s \cdot N_p. \tag{2.6.7}$$

Here, C_t^1 should be of the same order of magnitude as C_ω and is negligible compared to $C_\omega N_\omega$. From (2.6.7), we can have two immediate conclusions:

1. The time-domain method is favorable when the laser pulse duration or the charge of the particle beam is large.
2. The frequency-domain method is better when a particle beam with high energy or long bunch length is considered.

AXSIS	ODU CLS	ASU CXFEL	ASU CXLS	XFI	MuLCS	ThomX
4.75×10^{-7}	6.09×10^{-9}	3.77×10^{-10}	1.69×10^{-10}	1.89×10^{-10}	3.91×10^{-11}	1.79×10^{-10}

Table 2.5: The ratio of $N_{T_u} \cdot \log N_{T_u}$ and $C_\omega/C_t^2 \cdot N_\omega \cdot N_s \cdot N_p$ for different experimental projects is computed. Here, the electron beam transverse size l_x and l_y are not considered in the evaluation of N_{Tu} as their contribution is minor for a high-energy electron beam. The reference of each experimental project can be found in Table 2.2.

In addition, the factor C_ω/C_t^2 in (2.6.7) can also have significant contribution. The value of C_ω depends not only on the performance of hardware but also on the algorithm for solving the particle trajectory. If a more sophisticated algorithm is utilized (*e.g.*, particle-in-cell method [37, 38]), the time-domain method may become more favorable. In Table 2.5, we demonstrate the ratio of $N_{T_u} \cdot \log N_{T_u}$ and $C_\omega/C_t^2 \cdot N_\omega \cdot N_s \cdot N_p$ for different experimental projects. We can observe that the time-domain method is still faster than the frequency-domain method even if we choose N_ω so that $\Delta\omega$ fulfills (2.6.6).

Although the time-domain method (TDM) outperforms the frequency-domain method (FDM) for large particle numbers typically used in ICS sources, it is interesting to study the transition where the TDM outperforms the FDM for low particle numbers since ICS is also used in other applications. To numerically demonstrate the situation where FDM outperforms the TDM, we consider a bunch of particles uniformly distributed along the longitudinal direction with length of 100 µm. This particle bunch interacts with 5 periods of a 1 µm sinusoidal wave. We measure the elapsed time for the simulation with different numbers of particles by both TDM and FDM. The result is demonstrated in Figure 2.12, the FDM wins when $N_p \leq 40$ and the TDM wins when $N_p > 40$.

Although the performance model is verified by the execution times in serial, our conclusions can be extended to the scenario with parallelization (*i.e.*, with detector parallelization) in which the total operation counts for the time-domain and frequency-domain methods can be written as OP_t/P and OP_ω/P, respectively.

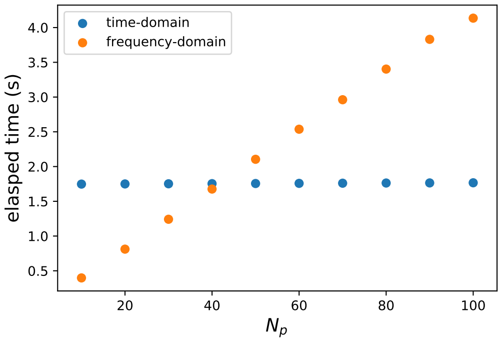


Figure 2.12: Elapsed times for the simulation ($N_s = 100$) with different numbers of particles (N_p) by the time-domain method ($N_{Tu} = 1.28 \times 10^6$) and the frequency-domain method ($N_\omega = 100$). Each data point is the average elapsed time from 50 simulation runs. In this case, the frequency-domain method outperforms the time-domain method when $N_p \leq 40$.

2.7 Summary

In this chapter, we systematically introduce the Liénard-Wiechert potential method for the calculation of the radiation spectra from the charged particles. Besides, an efficient time-domain method for solving Liénard-Wiechert potentials is proposed. We also compare the time-domain method with the frequency-domain method by considering the parameters of real-world experiment projects. Our result shows that the time-domain method is in general, *i.e.*, within the specification of real-world experiment projects, more favorable than the frequency-domain method in terms of execution time in serial and in parallel when applied to compute the radiation spectra of ICS sources. Besides, we also show that the frequency-domain method can outperform the time-domain method in some circumstances.

Chapter 3

Relativistic Space-Charge Force Calculation by Interpolation-Based Treecode

The content of this chapter is adapted from our research work published in Ref. [33]. In this chapter, we formulate a treecode based on Lagrangian interpolation for computing the relativistic space-charge field. Two approaches are introduced to control the interpolation error. In the first approach, based on the theoretical error analysis, a modified admissibility condition is proposed for which the treecode can be used directly in the lab-frame. The second approach is based on the Lorentz transformation of the particle beam to the rest-frame where the conventional admissibility condition can be used. Numerical simulation results using both methods will be compared and discussed.

3.1 Introduction

Space-charge effects are important in accelerator physics and can lead to many unwanted phenomena. For example, the space-charge force limits the intensity of the electron current emitted from the cathode inside the electron gun [49] and causes the broadening of ultrafast electron packets in the free-space propagation [50, 51]. In numerical simulations, grid-based methods have been the standard choice. Among all grid-based methods, the particle-in-cell (PIC) method is probably the most popular choice. PIC is a self-consistent model considering the field generation from the charged particle and the field-particle interaction [52]. In the framework of electromagnetic particle-in-cell (EM-PIC), the particle trajectory is used to obtain charge or current density over a spatial grid by a charge deposition scheme [53, 54]. With the charge and current density, the corresponding electromagnetic field is then evaluated by solving Maxwell's equations [55, 56]. In combination with a suitable numerical integrator [44], the solution of the particle field and a given external field are used to push the particles to their new states of motion. For non-relativistic particle beams, the electrostatic particle-in-cell (ES-PIC) method is usually used where Poisson's equation is solved to compute the particle field [57]. In addition to the electromagnetic model, the relativistic particle beam can be simulated by a quasi-static model [58, 59] which solves the electrostatic field in the rest-frame of the particle beam and applies the corresponding electromagnetic field in the lab-frame. However, PIC has several numerical issues despite its popularity. For example, the standard EM-PIC based on the finite-difference time-domain (FDTD) method has numerical dispersion due to the approximation with finite-difference stencils [60]. The pseudo-spectral methods, *e.g.* the pseudo-spectral time-domain (PSTD) [56] method or the pseudo-spectral analytical time-domain (PSATD) method [61] were used to mitigate this problem by evaluating the spatial derivative in the spectral domain. However, because of the usage of the fast Fourier transform (FFT), the pseudo-spectral method is computationally more demanding compared to FDTD and its performance cannot scale over many computing nodes [62]. Besides, standard PIC uses a fixed-size grid to discretize the spatial domain and is inefficient for non-uniform particle distributions [63].

The computation of the space-charge field can also be achieved by a direct N-body summation (also called brute-force method). One major advantage of this method is the consideration of the Coulomb collision effect, which is especially critical in some problems of the accelerator physics [64, 65]. However, a direct N-body method requires a computational cost of $\mathcal{O}(N^2)$ and may not be applicable if a large number of particles N is considered. Therefore, many efforts have been devoted to the development of tree-based methods where particles are subdivided into a hierarchy of clusters and the hierarchical relation of each cluster is stored in a tree data structure called cluster tree. Treecodes [66] rely on the approximation of particle-cluster interactions and have a complexity down to $\mathcal{O}(N \log N)$. The force field on each particle is computed through an independent tree traversal starting from the root cluster and the applicability of the particle-cluster approximation is determined by the multiple acceptance criterion (MAC) which is similar to the admissibility condition in the study of hierarchical (H-) matrices [67, 68]. The fast multipole method (FMM) [69] can further reduce the complexity down to $\mathcal{O}(N)$ by considering the cluster-cluster interactions. In traditional FMMs, each cluster at the same level is covered by a cubic box of the same size and the cluster-cluster interaction list is determined by a cluster's neighbor boxes and its parent's neighbor boxes. Compared to treecodes, one major drawback of traditional FMMs is that the well-separation condition relies on bounding boxes of a fixed size. Such a definition of the well-separation condition leads to two problems:

1. Unlike the MAC in treecode, the well-separation condition cannot be flexibly controlled.
2. It excludes the usage of tight bounding boxes (can be of rectangular shape) which have an adaptive size depending on the cluster and are favorable for non-uniform particle distributions.

Therefore, traditional FMMs are inefficient in treating non-uniform particle distributions [70]. There also exist hybrid FMMs merging the strengths from both treecode and traditional FMMs [71, 72, 73]. One effort of hybrid FMMs is based on the dual tree traversal [71, 72] where the cluster-cluster interaction list is determined by traversing the source and target cluster trees simultaneously and the well-separation condition can be defined as flexible as the MAC.

There have been many efforts using tree-based methods to model Coulomb interaction in the study of non-relativistic charged particles, *e.g.* plasma dynamics [74, 75], electron dynamics in ultrafast electron microscopy [76] and proton dynamics in synchrotrons [77]. For the dynamics of relativistic electron beams, the relativistic interaction kernel needs to be considered as the particles' field lines get compressed in the transverse direction [42]. The evaluation of the relativistic kernel based on the brute-force method was used in some studies [78, 41, 79]. To the best of our knowledge, there are few studies on using tree-based methods to calculate the relativistic space-charge field; only some former efforts based on the quasi-static model can be found [80, 81, 82]: the space-charge field is solved in the rest-frame of particle beam by using the treecode/FMM .

In this study, we first introduce the general concept of a treecode. After that, based on the treecode proposed by Wang *et al.* [83], we formulate an interpolation-based treecode for computing the relativistic space-charge field. In particular, we propose two methods to control the interpolation error:

1. Based on the analytic estimation of the interpolation error bound, a modified admissibility condition is derived so that the formulated treecode can be performed directly in the lab-frame.
2. The system is first transformed to the rest-frame of the average particle momentum in which the particle field is computed by a treecode and is then transformed back to the lab-frame. By using the relativistic transformation to the rest-frame, the formulated treecode can work with the conventional admissibility condition.

Our numerical results show that the treecode based on the modified admissibility condition has better accuracy than the treecode based on the relativity transformation when a particle beam with momentum spread is considered; an explanation is also provided. Besides, we demonstrate that the proposed treecode scales like $\mathcal{O}(N \log N)$.

3.2 The Idea of Treecode

In this section, we provide a 2D example to illustrate the idea of treecode. The example provided here can easily be generalized to 3D, all our computations are done in 3D.

In the N-body problem, the total force field f experienced by a target point $\boldsymbol{x} \neq \boldsymbol{x}_j$ can be written as

$$f(\boldsymbol{x}) = \sum_{j \in \widehat{S}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j \qquad (3.2.1)$$

where j denotes the index of the source particle with the position $\boldsymbol{x}_j$ in a cluster S and $\widehat{S}$ denotes the index set of the particles in S. Through an interaction kernel $g(\boldsymbol{x}, \boldsymbol{x}_j)$, the source particle at $\boldsymbol{x}_j$ applies the force field with the magnitude proportional to a physical quantity m_j of the particle.

When evaluating (3.2.1), we can divide the source particles contained in S into several smaller clusters and split the summation corresponding to

$$f(\boldsymbol{x}) = \sum_{j \in \widehat{S}^{(1)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{j \in \widehat{S}^{(1)}_{(2,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{j \in \widehat{S}^{(1)}_{(1,2)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{j \in \widehat{S}^{(1)}_{(2,2)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j, \quad (3.2.2)$$

where $S^{(1)}_{(1,1)}$, $S^{(1)}_{(2,1)}$, $S^{(1)}_{(1,2)}$ and $S^{(1)}_{(2,2)}$ are the sub-clusters located in the relative locations at the left-top, left-bottom, right-top, and right-bottom from the domain Q_S of the cluster S, respectively (Figure 3.1a). To express things in a general manner, we use $S^{(\ell)}_{(i,j)}$ to denote the sub-cluster from the ℓ-th subdivision (also called level) of the root cluster S with the location indices $i, j \in \{1, \ldots, 2^{\ell}\}$. If the distance from the target point $\boldsymbol{x}$ in $S^{(1)}_{(1,1)}$ to $S^{(1)}_{(2,1)}$, $S^{(1)}_{(1,2)}$, and $S^{(1)}_{(2,2)}$ is far enough (*i.e.*, fulfills a sort of admissibility condition similar to the far-field condition) and the kernel function can be approximated by a separable expansion with a rank N_r over the domain Q_S of a cluster S by

$$g(\boldsymbol{x}, \boldsymbol{x}_j) \approx \sum_{\nu=1}^{N_r} a_{S,\nu}(\boldsymbol{x}) \cdot b_{S,\nu}(\boldsymbol{x}_j) \quad \boldsymbol{x}_j \in S, \qquad (3.2.3)$$

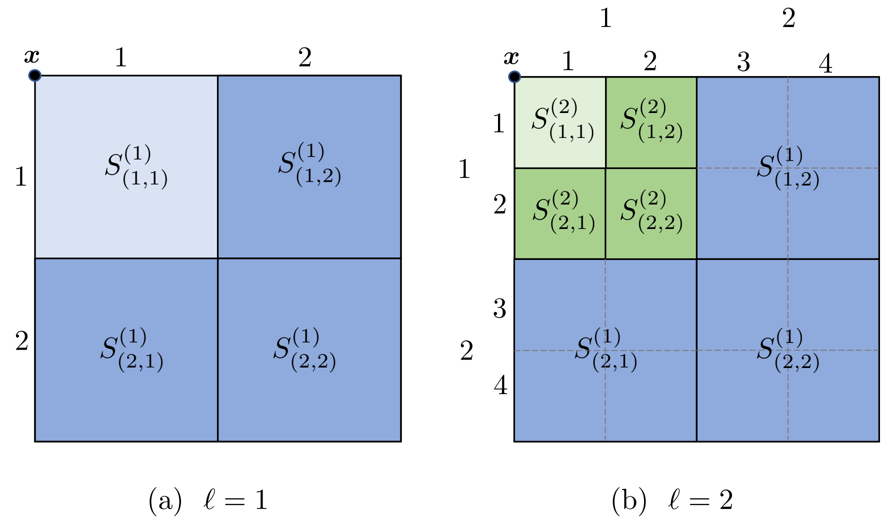


Figure 3.1: Illustration of the subdivision of the particle cluster at the levels (a) $\ell = 1$ and (b) $\ell = 2$. Here, the dark color denotes the far-field clusters and the light color denotes the near-field clusters.

then the force field from the particles can be approximated by

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{j \in \widehat{S}^{(1)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{\tau \in T} \sum_{j \in \widehat{S}^{(1)}_{\tau}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j \\ &\approx \sum_{j \in \widehat{S}^{(1)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{\tau \in T} \sum_{j \in \widehat{S}^{(1)}_{\tau}} \sum_{\nu=1}^{N_r} a_{S^{(1)}_{\tau},\nu}(\boldsymbol{x}) \cdot b_{S^{(1)}_{\tau},\nu}(\boldsymbol{x}_j) m_j \\ &= \sum_{j \in \widehat{S}^{(1)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{\tau \in T} \sum_{\nu=1}^{N_r} a_{S^{(1)}_{\tau},\nu}(\boldsymbol{x}) \sum_{j \in \widehat{S}^{(1)}_{\tau}} b_{S^{(1)}_{\tau},\nu}(\boldsymbol{x}_j) m_j \\ &= \sum_{j \in \widehat{S}^{(1)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j + \sum_{\tau \in T} \sum_{\nu=1}^{N_r} a_{S^{(1)}_{\tau},\nu}(\boldsymbol{x}) \cdot m_{S^{(1)}_{\tau},\nu} \end{aligned}$$

with

$$m_{S^{(\ell)}_{\tau},\nu} := \sum_{j \in \widehat{S}^{(\ell)}_{\tau}} b_{S^{(\ell)}_{\tau},\nu}(\boldsymbol{x}_j) m_j. \tag{3.2.4}$$

Here, $T = \{(2,1),(1,2),(2,2)\}$ is the set of indices for the far-field clusters at the current level. In this illustrative example, the target point $\boldsymbol{x}$ is in the top-left corner of the domain such that the indices of far-field clusters in each level all belong to the set $T = \{(2,1),(1,2),(2,2)\}$. In general, depending on the position of the target point, the index of a far-field cluster can be (i,j) with $i,j \in \{1,\ldots,2^{\ell}\}$. The effective physical quantity of the macro particle with the index ν is denoted by $m_{S^{(\ell)}_{\tau},\nu}$ in (3.2.4). The physical meaning of $m_{S^{(\ell)}_{\tau},\nu}$ is that the particles in a cluster are aggregated to a few macro particles; during the evaluation of the force field on the target particle, instead of traversing each real particle in the source cluster, we only consider a few macro particles if the approximation (3.2.3) is accurate, which is typically the case if the distance to the cluster is large enough.

To evaluate the force field from the source particles inside the cluster $S^{(1)}_{(1,1)}$, we can apply a similar trick as before. We first subdivide each sub-cluster from the level 1 into four sub-clusters ($S^{(2)}_{(1,1)}$, $S^{(2)}_{(2,1)}$, $S^{(2)}_{(1,2)}$, and $S^{(2)}_{(2,2)}$) and then use them to compute the approximated field (Figure 3.1b). Assume that at each level ℓ of the subdivision only one cluster $S^{(\ell)}_{(1,1)}$ does not fulfill the far-field condition. We can compute the force field from $S^{(\ell)}_{(2,1)}$, $S^{(\ell)}_{(1,2)}$, and $S^{(\ell)}_{(2,2)}$ by the far-field approximation and subdivide $S^{(\ell)}_{(1,1)}$ to get the sub-clusters of the next level $\ell + 1$. This procedure can be applied repeatedly until a maximum level κ is reached and the approximated force field can be computed by

$$f(\boldsymbol{x}) \approx \underbrace{\sum_{j \in \widehat{S}^{(\kappa)}_{(1,1)}} g(\boldsymbol{x}, \boldsymbol{x}_j) m_j}_{\text{near-field}} + \underbrace{\sum_{\ell=1}^{\kappa} \sum_{\tau \in T} \sum_{\nu=1}^{N_r} a_{S^{(\ell)}_{\tau},\nu}(\boldsymbol{x}) \cdot m_{S^{(\ell)}_{\tau},\nu}}_{\text{far-field}} . \tag{3.2.5}$$

This means that the evaluation of the force-field can be decomposed into near-field and far-field terms. The far-field term is computed approximately by the effective physical quantities of the macro particles inside the far-field clusters of different levels. The near-field term is evaluated directly from the physical quantities of micro particles inside the near-field clusters.

Consider a root cluster containing N uniformly distributed particles. If every cluster of the finest level contains N_0 particles (*i.e.*, $|\widehat{S}^{(\kappa)}_{(i,j)}| = N_0$ for all $i, j \in \{1, \ldots, 2^\kappa\}$), the number of subdivisions is given by

$$\kappa = \frac{1}{2 \log 2} \log \left(\frac{N}{N_0} \right) .$$

From a practical point of view, it is reasonable to choose $N_0 = N_r$ so that the number of macro particles N_r in the clusters (3.2.3) is not less than N_0. Since there are three far-field clusters at each level and one near-field cluster at the finest level, we need to traverse $1 + 3\kappa$ clusters to evaluate the field at $\boldsymbol{x}$. Therefore, the number of terms to evaluate (3.2.5) becomes

$$N_r \cdot (1 + 3\kappa) = N_r + \frac{3}{2 \log 2} \cdot N_r \cdot \log \left(\frac{N}{N_r} \right) .$$

If N_r is bounded and independent of N, the computation cost for evaluating the force field experienced by a single particle becomes $\mathcal{O}(\log N)$. This argument can be applied to each particle in the root cluster S and the total computation cost for evaluating the interaction force of N particles in S is $\mathcal{O}(N \log N)$.

From the example discussed above, we can see that the success of treecode relies on the hierarchical subdivision of particles into clusters and the approximation of the kernel function through a separable expansion (3.2.3). Thus, in the following sections, the following two main questions will be discussed:

- How can we subdivide the particles into a set of hierarchical clusters?
- How can we construct a separable approximation of a kernel function and bound the error of this approximation?

3.3 Cluster Tree

A cluster tree [67, 68] is a space-partition data structure (similar to the k-d tree [84]) which provides an efficient way for finding the interaction list in the tree-based method. In this section, we briefly introduce the terminology and the construction of the cluster tree.

Definition 3.3.1 (Cluster Tree)**.** *A cluster tree is a labeled tree associated with an index set I and fulfills the following requirements:*

- *For the root node r, its label set $\widehat{r}$ is given by the index set I,* i.e., $\widehat{r} = I$*.*
- *If s is a non-leaf node, its label set is the union of the label sets of its children nodes (*i.e., $\widehat{s} = \bigcup_{s' \in \text{children}(s)} \widehat{s'}$*).*
- *If $t', s' \in$ children(s), then $\widehat{s'} \bigcap \widehat{t'} = \emptyset$.*

We call the label sets associated with nodes of the tree (index) clusters.

Let $S = \{\boldsymbol{x}_1, \ldots \boldsymbol{x}_j\} \subset \mathbb{R}^3$ be given by a set of particle positions (which we also call a *cluster* of particles) with index set $\widehat{S} = \{1, \ldots, j\}$. We denote the entries in a vector by $\boldsymbol{x} = (x, y, z)^T$. The tightest rectangular box covering S is called the *bounding box* and is defined as

$$\text{bbox}(S) := [a_x, b_x] \times [a_y, b_y] \times [a_z, b_z],$$

with $a_g = \min_{i \in \widehat{S}}\{g_i\}$, $b_g = \max_{i \in \widehat{S}}\{g_i\}$ and $g \in \{x, y, z\}$. To subdivide a cluster S containing j particles, we first determine bbox(S) which enables us to find the coordinate direction with the biggest interval $k = \arg\max_{g \in \{x,y,z\}}(b_g - a_g)$. We then choose the particle position with the $\lfloor j/2 \rfloor$-th greatest value in the k-component of its position as the splitting point $\boldsymbol{x}_{\text{split}}$ such that the cluster S can be subdivided into two sub-clusters

$$S_1 = \{\boldsymbol{x}_i \mid k_i \leq k_{\text{split}},\ i \in \widehat{S}\} \quad \text{and} \quad S_2 = \{\boldsymbol{x}_i \mid k_i > k_{\text{split}},\ i \in \widehat{S}\}, \quad S = S_1 \bigcup S_2.$$

We can apply this subdivision procedure to each of the resulting sub-clusters repeatedly until the number of particles in the clusters at the deepest level is smaller than a pre-selected number N_0. Different to what has been used in the illustrative example of the previous section, this cardinality-balanced subdivision strategy is used in the remainder of this study since it has the following benefits:

- Each subdivided cluster contains roughly the same number of particles so that the cluster tree is balanced regardless of the particle distribution.
- The subdivision based on the coordinate direction with the biggest interval can shrink the diameter of the cluster fast.

An illustration of the cluster tree is demonstrated in Figure 3.2.

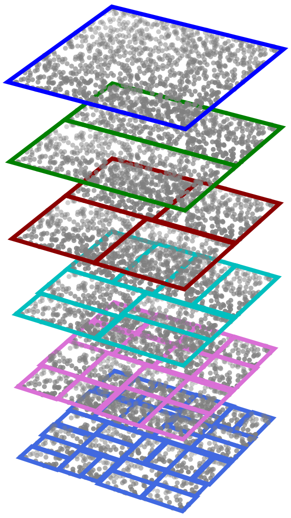


Figure 3.2: A cluster tree for 2D particles (gray dots). The root particle cluster (top layer) is subdivided into a hierarchy of clusters. The total number of particles is $N = 1024$ and the maximum number of particles in the cluster at the deepest level is $N_0 = 32$.

3.4 Approximation for the Kernel Function of the Space-Charge Field

The space-charge field from a relativistic particle at the position $\boldsymbol{x}_j$ experienced by a target position $\boldsymbol{x}$ can be expressed as [42, 41]

$$\boldsymbol{E}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) = \frac{q}{4\pi\varepsilon_0}\gamma_j \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) \quad \text{and} \quad \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) = \frac{q}{4\pi\epsilon_0 c_0}\boldsymbol{p}_j \times \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) \tag{3.4.1}$$

with

$$\boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) := \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\left|\|\boldsymbol{x} - \boldsymbol{x}_j\|_2^2 + (\boldsymbol{p}_j \cdot (\boldsymbol{x} - \boldsymbol{x}_j))^2\right|^{3/2}}. \tag{3.4.2}$$

Here, $\gamma_j = 1/\sqrt{1 - \|\boldsymbol{\beta}_j\|_2^2}$ and $\boldsymbol{p}_j = \gamma_j \boldsymbol{\beta}_j$ are the Lorentz factor and normalized momentum of the particle, respectively, with $\boldsymbol{\beta}_j$ the particle velocity normalized to the speed of light. Although not necessary, throughout this study, we assume that each particle in the system has the same charge q. In the physics of particle beams moving in the z-direction, the paraxial approximation can be usually applied

$$\overline{p}_z \gg \overline{p}_{\{x,y\}} \quad \text{and} \quad \overline{p}_{\{x,y,z\}} \gg \Delta p_{\{x,y,z\}}$$

where $\overline{\boldsymbol{p}} := \frac{1}{N}\sum_{j=1}^{N} \boldsymbol{p}_j$, $\boldsymbol{p}_j =: \overline{\boldsymbol{p}} + \Delta\boldsymbol{p}_j$ and N is the number of particles. Therefore, the kernel function can be approximated as

$$\begin{aligned} \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) &= \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\left|\|\boldsymbol{x} - \boldsymbol{x}_j\|_2^2 + ((\overline{\boldsymbol{p}} + \Delta\boldsymbol{p}_j) \cdot (\boldsymbol{x} - \boldsymbol{x}_j))^2\right|^{3/2}} \\ &\approx \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\left|\|\boldsymbol{x} - \boldsymbol{x}_j\|_2^2 + (\overline{\boldsymbol{p}} \cdot (\boldsymbol{x} - \boldsymbol{x}_j))^2\right|^{3/2}}. \end{aligned} \tag{3.4.3}$$

Since $\overline{p}_z \gg \overline{p}_{\{x,y\}}$, we may use $\overline{p}_z \approx \|\overline{\boldsymbol{p}}\|_2$ to further simplify equation (3.4.3) as in (cf. Section B.1)

$$\begin{aligned}\boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j) &\approx \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\Big((x - x_j)^2 + (y - y_j)^2 + (1 + \overline{p}_z^2)(z - z_j)^2\Big)^{3/2}} \\ &\approx \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\Big((x - x_j)^2 + (y - y_j)^2 + \overline{\gamma}^2(z - z_j)^2\Big)^{3/2}} =: \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{x}_j) \qquad (3.4.4)\end{aligned}$$

where $\overline{\gamma}^2 := 1 + \|\overline{\boldsymbol{p}}\|_2^2$. The space-charge field from the source particle can be approximated by

$$\boldsymbol{E}(\boldsymbol{x}, \boldsymbol{x}_j) \approx \frac{q}{4\pi\epsilon_0}\gamma_j \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{x}_j) \quad \text{and} \quad \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{x}_j) \approx \frac{q}{4\pi\epsilon_0}\boldsymbol{p}_j \times \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{x}_j).$$

The separable approximation of the kernel function can be achieved by a tensor product interpolation of the source variable $\boldsymbol{x}_j$ with Lagrangian polynomials [67, 68]

$$\begin{aligned}\widetilde{\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{x}_j) &= \sum_{\nu_1=0}^{n}\sum_{\nu_2=0}^{n}\sum_{\nu_3=0}^{n} \boldsymbol{g}\big(\boldsymbol{x}, (\xi_{\nu_1}^x, \xi_{\nu_2}^y, \xi_{\nu_3}^z)\big) \cdot \ell_{\nu_1}^x(x_j) \cdot \ell_{\nu_2}^y(y_j) \cdot \ell_{\nu_3}^z(z_j) \\ &=: \sum_{\boldsymbol{\nu} \in \{0,\ldots,n\}^3} \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\xi}_{\boldsymbol{\nu}})\ell_{\boldsymbol{\nu}}(\boldsymbol{x}_j),\end{aligned}$$

where the Lagrange basis polynomials are defined as

$$\ell_j^k(x) := \prod_{\substack{i=0 \\ i\neq j}}^{n} \frac{x - \xi_i^k}{\xi_j^k - \xi_i^k}, \quad j = 1, \ldots, n, \quad k \in \{x, y, z\} \qquad (3.4.5)$$

with the interpolation points ξ_i^k and the polynomial degree n.

The error bound of the tensor product interpolation of the kernel function with respect to the source point variable $\boldsymbol{x}_j$ in a rectangular domain $Q_S = [a_x, b_x] \times [a_y, b_y] \times [a_z, b_z] \subset \mathbb{R}^3$ is [68]

$$\|\boldsymbol{g}(\boldsymbol{x}, \cdot) - \widetilde{\boldsymbol{g}}(\boldsymbol{x}, \cdot)\|_{\infty, Q_S} \leq \text{const} \cdot \sum_{k \in \{x,y,z\}} (b_k - a_k)^{n+1} \cdot \frac{\|\partial_k^{n+1}\boldsymbol{g}(\boldsymbol{x}, \cdot)\|_{\infty, Q_S}}{(n+1)!}. \qquad (3.4.6)$$

As indicated by (3.4.6), to control the interpolation error, we need to find a condition which guarantees a bound for the right-hand-side terms. In the literature, this condition is called admissibility condition [67, 68]. In particular, when determining an admissibility condition for a kernel function $g(\boldsymbol{x}, \cdot)$, we need to find a bound on the term $\|\partial_k^{n+1}\boldsymbol{g}(\boldsymbol{x}, \cdot)\|_{\infty, Q_S}$.

3.5 Admissibility Condition for the Relativistic Kernel

As mentioned in Section 3.4, to derive an admissibility condition for a kernel function, it is necessary to bound the higher-order derivatives of the kernel function. For studying the free space electrostatic problem, the force kernel can be written

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{x}_j) := \frac{\boldsymbol{x} - \boldsymbol{x}_j}{\|\boldsymbol{x} - \boldsymbol{x}_j\|_2^3}. \qquad (3.5.1)$$

An upper bound of the higher-order derivatives with respect to $\boldsymbol{x}_j$ is given by [68]

$$\|\partial_i^{n+1}\boldsymbol{f}(\boldsymbol{x},\cdot)\|_{\infty,Q_S} \leq \frac{\text{const}}{\text{dist}(x,S)^2}\frac{(n+1)!}{\text{dist}(\boldsymbol{x},S)^{n+1}}, \quad i\in\{x,y,z\}$$

and the substitution into (3.4.6) yields

$$\|\boldsymbol{f}(\boldsymbol{x},\cdot)-\tilde{\boldsymbol{f}}(\boldsymbol{x},\cdot)\|_{\infty,Q_S} \leq \frac{\text{const}}{\text{dist}(\boldsymbol{x},S)^2}\left(\frac{\text{diam}(S)}{\text{dist}(\boldsymbol{x},S)}\right)^{n+1}. \tag{3.5.2}$$

Here, we use the definitions

$$\text{diam}(S) := \max_{\boldsymbol{x}_i,\boldsymbol{x}_j\in S}\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2 \quad \text{and} \quad \text{dist}(\boldsymbol{x},S) := \min_{\boldsymbol{x}_j\in S}\|\boldsymbol{x}-\boldsymbol{x}_j\|_2 \tag{3.5.3}$$

to denote the diameter of S and the distance from $\boldsymbol{x}$ to S.

From (3.5.2), we can define an admissibility condition (also called η-admissibility [68]) for the electrostatic kernel by

$$\frac{\text{diam}(S)}{\text{dist}(\boldsymbol{x},S)} < \eta \tag{3.5.4}$$

with some admissibility parameter $\eta\in\mathbb{R}_{>0}$.

However, the admissibility condition of the electrostatic kernel (called conventional admissibility condition in this study) may not be used for the relativistic kernel (3.4.4). Thus, we will derive an upper bound of the interpolation error for the relativistic kernel and derive a corresponding admissibility condition. We first introduce a theorem which is useful for estimating the upper bound of the higher-order derivatives of a specific type of kernel function.

Theorem 3.5.1 ([68, Theorem E.4])**.** *Let $s(\boldsymbol{x},\boldsymbol{x}_j)=1/\|\boldsymbol{x}-\boldsymbol{x}_j\|_2^a$ with $a\in\mathbb{R}_{>0}$, we have*

$$|\partial_{\boldsymbol{x}}^{\boldsymbol{\nu}}s(\boldsymbol{x},\boldsymbol{x}_j)| \leq \boldsymbol{\nu}!w^{a/2+|\boldsymbol{\nu}|}\|\boldsymbol{x}-\boldsymbol{x}_j\|_2^{-|\boldsymbol{\nu}|-a} \quad \forall\boldsymbol{x},\boldsymbol{x}_j\in\mathbb{R}^d\wedge\boldsymbol{x}\neq\boldsymbol{x}_j,$$

with a suitable constant w. Here, $\boldsymbol{\nu}=(\nu_1,\ldots,\nu_d)\in\mathbb{N}_0^d$ and the following conventions are used [68]:

$$|\boldsymbol{\nu}|=\sum_{i=1}^d\nu_i, \quad \boldsymbol{\nu}!=\prod_{i=1}^d\nu_i!, \quad \partial_{\boldsymbol{x}}^{\boldsymbol{\nu}}=\prod_{i=1}^d\Big(\frac{\partial}{\partial x_i}\Big)^{\nu_i}.$$

Since $s(\boldsymbol{x},\boldsymbol{x}_j)$ is symmetric with respect to $\boldsymbol{x}$ and $\boldsymbol{x}_j$, the conclusion above also holds for the derivatives with respect to $\boldsymbol{x}_j$.

The force kernel for the relativistic space-charge field is (3.4.4)

$$\boldsymbol{g}(\boldsymbol{x},\boldsymbol{x}_j)=\frac{\boldsymbol{x}-\boldsymbol{x}_j}{\Big((x-x_j)^2+(y-y_j)^2+\overline{\gamma}^2(z-z_j)^2\Big)^{3/2}}.$$

By introducing a change of variables

$$\begin{cases}\overline{x}=x,\\ \overline{y}=y,\\ \overline{z}=\overline{\gamma}z,\end{cases} \quad \text{and} \quad \begin{cases}\overline{x_j}=x_j,\\ \overline{y_j}=y_j,\\ \overline{z_j}=\overline{\gamma}z_j,\end{cases}$$

the kernel function $\boldsymbol{g}$ can expressed component-wise as

$$
\begin{aligned}
(g_x, g_y, g_z) &= \frac{\left(\overline{x} - \overline{x_j}, \overline{y} - \overline{y_j}, \frac{1}{\overline{\gamma}}(\overline{z} - \overline{z_j})\right)}{\left((\overline{x} - \overline{x_j})^2 + (\overline{y} - \overline{y_j})^2 + (\overline{z} - \overline{z_j})^2\right)^{3/2}} \\
&= (\partial_{\overline{x_j}}, \partial_{\overline{y_j}}, \tfrac{1}{\overline{\gamma}}\partial_{\overline{z_j}})\phi \quad \text{with} \quad \phi := \frac{1}{\|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_2}.
\end{aligned}
$$

Therefore, the norm of the derivative of $\boldsymbol{g}$ along each coordinate direction is

$$
\begin{aligned}
\|\partial_{x_j}^{n+1}\boldsymbol{g}\|_\infty &= \max(|\partial_{\overline{x_j}}^{n+1}\partial_{\overline{x_j}}\phi|, |\partial_{\overline{x_j}}^{n+1}\partial_{\overline{y_j}}\phi|, |\partial_{\overline{x_j}}^{n+1}\partial_{\overline{z_j}}\phi|), \\
\|\partial_{y_j}^{n+1}\boldsymbol{g}\|_\infty &= \max(|\partial_{\overline{y_j}}^{n+1}\partial_{\overline{x_j}}\phi|, |\partial_{\overline{y_j}}^{n+1}\partial_{\overline{y_j}}\phi|, |\partial_{\overline{y_j}}^{n+1}\partial_{\overline{z_j}}\phi|), \\
\|\partial_{z_j}^{n+1}\boldsymbol{g}\|_\infty &= \overline{\gamma}^{n+1} \cdot \max(|\partial_{\overline{z}}^{n+1}\partial_{\overline{x_j}}\phi|, |\partial_{\overline{z_j}}^{n+1}\partial_{\overline{y_j}}\phi|, \tfrac{1}{\overline{\gamma}}|\partial_{\overline{z_j}}^{n+1}\partial_{\overline{z_j}}\phi|).
\end{aligned}
$$

Using Theorem 3.5.1 with $a = 1$ and $|\boldsymbol{\nu}| = n + 1$, the upper bounds of the norms of the derivatives of $\boldsymbol{g}$ are

$$
\begin{aligned}
\|\partial_{x_j}^{n+1}\boldsymbol{g}\|_\infty &\leq \text{const} \cdot (n+1)! \cdot \|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_2^{-(n+3)}, \\
\|\partial_{y_j}^{n+1}\boldsymbol{g}\|_\infty &\leq \text{const} \cdot (n+1)! \cdot \|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_2^{-(n+3)}, \\
\|\partial_{z_j}^{n+1}\boldsymbol{g}\|_\infty &\leq \text{const} \cdot (n+1)! \cdot \overline{\gamma}^{n+1} \cdot \|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_2^{-(n+3)}.
\end{aligned}
$$

After the substitution of the upper bounds above into (3.4.6), the interpolation error becomes

$$
\begin{aligned}
\|\boldsymbol{g}(\boldsymbol{x}, \cdot) - \widetilde{\boldsymbol{g}}(\boldsymbol{x}, \cdot)\|_{\infty, Q_S} &\leq \text{const} \cdot \sum_{k \in \{x,y,z\}} s_k^{n+1}(b_k - a_k)^{n+1} \cdot \frac{1}{\|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_\infty^{n+3}} \\
&= \text{const} \cdot \sum_{k \in \{x,y,z\}} (\bar{b}_k - \overline{a}_k)^{n+1} \cdot \frac{1}{\|\overline{\boldsymbol{x}} - \overline{\boldsymbol{x}_j}\|_\infty^{n+3}} \\
&\leq \frac{\text{const}}{\text{dist}(\overline{x}, \overline{S})^2} \frac{\text{diam}(\overline{S})^{n+1}}{\text{dist}(\overline{\boldsymbol{x}}, \overline{S})^{n+1}}
\end{aligned}
$$

with a three dimensional stretch factor $\boldsymbol{s} = (1, 1, \gamma)$ and $\overline{S} := \{\boldsymbol{s} \circ \boldsymbol{x}_j \,|\, \forall \boldsymbol{x}_j \in S\}$ the set of stretched positions of particles from S. Here, the symbol

$$
\circ : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n \tag{3.5.5}
$$

denotes the component-wise product of two vectors. In the derivation above, the equivalence of the two-norm and the infinity norm has been used [85].

To express the admissibility condition in the original variables, we define the stretched diameter and the stretched distance

$$
\begin{aligned}
\overset{(s_x, s_y, s_z)}{\text{diam}}(S) &:= \max_{\boldsymbol{x}_i, \boldsymbol{x}_j \in S} \|(s_x, s_y, s_z) \circ (\boldsymbol{x}_i - \boldsymbol{x}_j)\|_2, \\
\overset{(s_x, s_y, s_z)}{\text{dist}}(\boldsymbol{x}, S) &:= \min_{\boldsymbol{x}_j \in S} \|(s_x, s_y, s_z) \circ (\boldsymbol{x} - \boldsymbol{x}_j)\|_2,
\end{aligned}
$$

where $\boldsymbol{s} = (s_x, s_y, s_z)$ is a three-dimensional stretch factor such that this holds:

$$
\overset{(1,1,\overline{\gamma})}{\text{diam}}(S) = \text{diam}(\overline{S}) \quad \text{and} \quad \overset{(1,1,\overline{\gamma})}{\text{dist}}(\boldsymbol{x}, S) = \text{dist}(\overline{\boldsymbol{x}}, \overline{S}).
$$

Therefore, the stretched admissibility condition for the relativistic kernel can be defined with

$$
\frac{\overset{(1,1,\overline{\gamma})}{\text{diam}}(S)}{\overset{(1,1,\overline{\gamma})}{\text{dist}}(\boldsymbol{x}, S)} \leq \eta. \tag{3.5.6}
$$

3.6 An Interpolation-Based Treecode for Evaluating the Relativistic Space-Charge Field

When a particle-cluster pair satisfies an admissibility condition, the separable approximation to the kernel function through the Lagrangian interpolation can be applied and the space-charge field of the particle cluster S can be computed approximately by

$$\begin{aligned}\sum_{j\in\widehat{S}}\boldsymbol{E}(\boldsymbol{x},\boldsymbol{x}_j) &\approx \frac{q}{4\pi\epsilon_0}\sum_{\boldsymbol{\nu}}\sum_{j\in\widehat{S}}\ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\gamma_j\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\xi}_{\boldsymbol{\nu}}) = \frac{q}{4\pi\epsilon_0}\sum_{\boldsymbol{\nu}}\gamma_{S,\boldsymbol{\nu}}\cdot\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\xi}_{\boldsymbol{\nu}}),\\ \sum_{j\in\widehat{S}}\boldsymbol{B}(\boldsymbol{x},\boldsymbol{x}_j) &\approx \frac{q}{4\pi\epsilon_0}\sum_{\boldsymbol{\nu}}\sum_{j\in\widehat{S}}\ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\boldsymbol{p}_j\times\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\xi}_{\boldsymbol{\nu}}) = \frac{q}{4\pi\epsilon_0}\sum_{\boldsymbol{\nu}}\boldsymbol{p}_{S,\boldsymbol{\nu}}\times\boldsymbol{g}(\boldsymbol{x},\boldsymbol{\xi}_{\boldsymbol{\nu}}).\end{aligned} \tag{3.6.1}$$

Here, the quantities

$$\gamma_{S,\boldsymbol{\nu}} := \sum_{j\in\widehat{S}}\ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\gamma_j \quad\text{and}\quad \boldsymbol{p}_{S,\boldsymbol{\nu}} := \sum_{j\in\widehat{S}}\ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\boldsymbol{p}_j \tag{3.6.2}$$

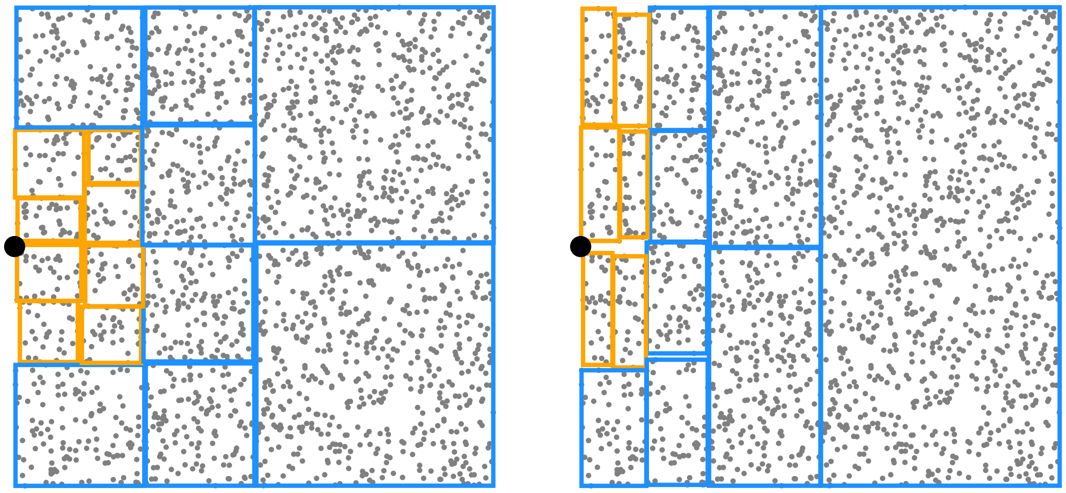
are the effective Lorentz factor and the effective momentum of the macro particle with a position $\boldsymbol{\xi}_{\boldsymbol{\nu}}$ in the cluster S.

Based on (3.6.1), we can formulate a treecode for computing the relativistic space-charge field. When the treecode is applied with the stretched admissibility condition, it is advantageous to adjust the procedure to determine the splitting direction (k_{split} defined in Section 3.3). As implied by the stretched diameter in (3.5.6), the splitting coordinate direction should be determined by the stretched bounding box. Therefore, in the implementation of the treecode with the stretched admissibility condition, a modified strategy to determine the splitting direction is used (Algorithm 3.3 with $\boldsymbol{s} = (1, 1, \overline{\gamma})$).

The formulated treecode (Algorithm 3.1) consists of the two main procedures:

1. A given root particle cluster is subdivided into a hierarchy of smaller clusters (Algorithm 3.2). The splitting direction is determined by the biggest length of the bounding box (Algorithm 3.3 with $\boldsymbol{s} = (1, 1, 1)$) or the stretched bounding box (Algorithm 3.3 with $\boldsymbol{s} = (1, 1, \overline{\gamma})$).
2. In the evaluation of the field on each target point (Algorithm 3.4), an independent traversal of the cluster tree is performed to determine the candidates of the interaction cluster. When a target point to a non-leaf cluster fulfills the conventional admissibility condition (Algorithm 3.5 with $\boldsymbol{s} = (1, 1, 1)$) or the stretched admissibility condition (Algorithm 3.5 with $\boldsymbol{s} = (1, 1, \overline{\gamma})$), the effective quantities are used to compute the space-charge field (3.6.1). Note that, instead of (3.5.3), we adapt a different definition to compute the diameter and the distance for the admissibility condition in Algorithm 3.5 because of its simplicity in the practical implementation.

To illustrate the effect of the stretch on the particle-clustering of treecode, we performed 2D simulations for particles uniformly randomly distributed over the unit square $[0, 1]^2$. The result is demonstrated in Figure 3.3.



(a) non-stretched

(b) stretched

Figure 3.3: Clustering of 2D particles (gray dots) with respect to a target point (black dot) by using (a) treecode without stretch and (b) treecode with a stretch by a factor of 2.5 in the horizontal direction (*i.e.*, $\boldsymbol{s} = (2.5, 1)$). The bounding box of the leaf clusters and the admissible clusters (from different levels) are marked with orange and blue, respectively. The simulation is performed with admissibility parameter $\eta = 0.5$, total number of particles $N = 2048$ and maximum number of particles in the leaf clusters $N_0 = 32$.

Algorithm 3.1: Treecode with Stretch

```
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$N_0$: maximum number of particles in the leaf node
$\eta$: admissibility parameter
$\boldsymbol{E}_i, \boldsymbol{B}_i$: the space-charge field experienced by the $i$-th particle
Function TreecodeStretch($S$, $\boldsymbol{s}$, $N_0$, $\eta$)
    subdivide($S$, $\boldsymbol{s}$, $N_0$) (Algorithm 3.2)
    $\boldsymbol{E}_i = \boldsymbol{0}, \boldsymbol{B}_i = \boldsymbol{0} \quad \forall i \in \widehat{S}$
    for $i \in \widehat{S}$ do
        $\boldsymbol{E}_i, \boldsymbol{B}_i$ = cluster2p($\boldsymbol{x}_i$, $S$, $\boldsymbol{s}$, $\eta$) (Algorithm 3.4)
    end
    return $\{\boldsymbol{E}_i \mid i \in \widehat{S}\}$, $\{\boldsymbol{B}_i \mid i \in \widehat{S}\}$
end
```

3.7 An Alternative Method Based on Relativity Transformation

In the previous section, we derived an upper bound of the interpolation error for the relativistic interaction kernel and introduced a stretched admissibility condition.

Algorithm 3.2: Subdivision of Particle Cluster

```
$\gamma_{S,\boldsymbol{\nu}}$: effective Lorenz factor for cluster (global variable) (3.6.2)
$\boldsymbol{p}_{S,\boldsymbol{\nu}}$: effective momentum for cluster (global variable) (3.6.2)
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$N_0$: maximum number of particles in the leaf node
Function subdivide($S$, $\boldsymbol{s}$, $N_0$)
    if $|S| > N_0$ then
        compute $\gamma_{S,\boldsymbol{\nu}}$ and $\boldsymbol{p}_{S,\boldsymbol{\nu}}$ using (3.6.2)
        $k$ = direction4split($S, \boldsymbol{s}$) (Algorithm 3.3)
        $k_{\text{split}} = \lfloor |S|/2 \rfloor$-th largest element of $\{k_i \mid i \in \widehat{S}\}$
        $S_1 = \{\boldsymbol{x}_i \mid k_i \leq k_{\text{split}}, i \in \widehat{S}\}$
        $S_2 = \{\boldsymbol{x}_i \mid k_i > k_{\text{split}}, i \in \widehat{S}\}$
        children(S) = $\{S_1, S_2\}$
        subdivide($S_1$, $\boldsymbol{s}$, $N_0$)
        subdivide($S_2$, $\boldsymbol{s}$, $N_0$)
    else
        children(S) = $\emptyset$
    end
end
```

Algorithm 3.3: Direction for the Split of Particle Cluster with Stretch

```
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
Function direction4split($S$, $\boldsymbol{s}$)
    $(s_x, s_y, s_z) = \boldsymbol{s}$
    $[a_x, b_x] \times [a_y, b_y] \times [a_z, b_z] = \text{bbox}(S)$
    $k = \underset{i \in \{x,y,z\}}{\text{argmax}}\ s_i(b_i - a_i)$
    return $k$
end
```

Algorithm 3.4: Cluster Field Evaluation

```
$\gamma_{S,\boldsymbol{\nu}}$: effective Lorenz factor for cluster (global variable) (3.6.2)
$\boldsymbol{p}_{S,\boldsymbol{\nu}}$: effective momentum for cluster (global variable) (3.6.2)
$\boldsymbol{x}$: position of the target point
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
Function cluster2p($\boldsymbol{x}$, $S$, $\boldsymbol{s}$, $\eta$)
    $\boldsymbol{E} = \mathbf{0}$, $\boldsymbol{B} = \mathbf{0}$
    isAdmissible = admissible($\boldsymbol{x}, S, \boldsymbol{s}, \eta$) (Algorithm 3.5)
    if children($S$) == $\emptyset$ then
        for $j \in \widehat{S}$ do
            $\boldsymbol{E} = \boldsymbol{E} + \gamma_j \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j)$
            $\boldsymbol{B} = \boldsymbol{E} + \boldsymbol{p}_j \times \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{x}_j, \boldsymbol{p}_j)$
        end
    else if isAdmissible then
        $\boldsymbol{E} = \boldsymbol{E} + \sum_{\boldsymbol{\nu}} \gamma_{S,\boldsymbol{\nu}} \cdot \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\xi}_{\boldsymbol{\nu}})$
        $\boldsymbol{B} = \boldsymbol{B} + \sum_{\boldsymbol{\nu}} \boldsymbol{p}_{S,\boldsymbol{\nu}} \times \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\xi}_{\boldsymbol{\nu}})$
    else
        for $s \in$ children($S$) do
            $\boldsymbol{E}_{\text{temp}}$, $\boldsymbol{B}_{\text{temp}}$ = cluster2p($\boldsymbol{x}$, $S$, $\boldsymbol{s}$, $\eta$)
            $\boldsymbol{E} = \boldsymbol{E} + \boldsymbol{E}_{\text{temp}}$
            $\boldsymbol{B} = \boldsymbol{B} + \boldsymbol{B}_{\text{temp}}$
        end
    end
    return $\boldsymbol{E}$, $\boldsymbol{B}$
end
```

Algorithm 3.5: Stretched Admissibility Condition

```
$\boldsymbol{x}$: position of the target point
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
Function admissible($\boldsymbol{x}$, $S$, $\boldsymbol{s}$, $\eta$)
    $[a_x, b_x] \times [a_y, b_y] \times [a_z, b_z] = \mathrm{bbox}(S)$
    $\boldsymbol{c} = (\boldsymbol{a} + \boldsymbol{b})/2$
    diam $= \|\boldsymbol{s} \circ (\boldsymbol{b} - \boldsymbol{c})\|_2$
    dist $= \|\boldsymbol{s} \circ (\boldsymbol{x} - \boldsymbol{c})\|_2$
    return diam/dist $\leq \eta$
end
```

In this section, we introduce an alternative approach based on the relativity transformation. With this approach, the treecode can be performed without switching to a stretched admissibility condition. We know that the necessity of the stretched admissibility condition comes from the average particle momentum term $\overline{\boldsymbol{p}}$ in the denominator of the relativistic kernel (3.4.4). If some transformation can be applied to eliminate the average momentum term, we can just use treecode with the conventional admissibility condition. The idea is to transform every physical quantity to the average rest-frame (AVGRF, the inertial frame moving with particle beam's average momentum) through the Lorentz transformation and evaluate the particle interaction in the frame. In the average rest-frame, the interaction kernel is approximately equal to the electrostatic kernel and treecode can be directly performed with the conventional admissibility condition.

3.7.1 Particle interaction in the average rest-frame

Let $\overline{\boldsymbol{p}}$ be the average momentum of particles in the system in the lab-frame $\mathcal{K}$ and $\mathcal{K}'$ be the average rest-frame in which the average particle momentum is $\overline{\boldsymbol{p}}' = 0$. According to (B.2.1) and (B.2.2) with $\boldsymbol{p}_u = \overline{\boldsymbol{p}}$, the position and the momentum in $\mathcal{K}$ and $\mathcal{K}'$ have the relations

$$x'_{\parallel} = \overline{\gamma} x_{\parallel} - \overline{p}(c_0 t), \qquad x'_{\perp} = x_{\perp}, \tag{3.7.1}$$

$$p'_{\parallel} = \overline{\gamma} p_{\parallel} - \overline{p}\gamma, \qquad p'_{\perp} = p_{\perp}, \tag{3.7.2}$$

where $\parallel$ and $\perp$ denote the components parallel and perpendicular to $\overline{\boldsymbol{p}}$. Here, we use $\overline{p}$ to denote the magnitude of $\overline{\boldsymbol{p}}$ ($i.e$, $\overline{p} := \|\overline{\boldsymbol{p}}\|_2$).

The momentum of a particle can be written as $\boldsymbol{p} = \overline{\boldsymbol{p}} + \Delta\boldsymbol{p}$ and can also be component-wise expressed in the plane spanned by the average momentum $\overline{\boldsymbol{p}}$ and the momentum $\boldsymbol{p}$ of the particle as

$$p_{\parallel} = \overline{p} + \Delta p_{\parallel} \quad \text{and} \quad p_{\perp} = \Delta p_{\perp}, \tag{3.7.3}$$

where $\Delta\boldsymbol{p}$ is the deviation of the particle momentum from $\overline{\boldsymbol{p}}$ with the scalar components $\Delta p_{\parallel}$, $\Delta p_{\perp}$. Throughout this study, we assume that $\overline{p} \gg \Delta p_{\parallel}, \Delta p_{\perp}$, which is valid in the physics of particle beams.

To eliminate the momentum appearing in the denominator of the relativistic kernel, we can evaluate the particle field in $\mathcal{K}'$,

$$\boldsymbol{E}' = \frac{1}{4\pi\epsilon_0} \frac{\gamma' \boldsymbol{R}'}{\left|\|\boldsymbol{R}'\|_2^2 + (\boldsymbol{p}' \cdot \boldsymbol{R}')^2\right|^{3/2}}, \qquad \boldsymbol{B}' = \frac{1}{4\pi\epsilon_0} \frac{\boldsymbol{p}_j \times \boldsymbol{R}'}{\left|\|\boldsymbol{R}'\|_2^2 + (\boldsymbol{p}' \cdot \boldsymbol{R}')^2\right|^{3/2}}, \tag{3.7.4}$$

where $\boldsymbol{R}'$ is the distance between the source and target particles, γ' and $\boldsymbol{p}_j$ are the Lorentz factor and the momentum of the source particle, respectively. Let $\Delta\gamma$ denote the deviation to the average Lorentz factor $\overline{\gamma}$ of the particle, $\gamma = \overline{\gamma} + \Delta\gamma$. By (3.7.2) and (3.7.3), the particle momentum in the average rest-frame can be approximated by

$$p'_{\parallel} = \overline{\gamma}(\overline{p} + \Delta p_{\parallel}) - \overline{p}(\overline{\gamma} + \Delta\gamma) = \overline{\gamma}\Delta p_{\parallel} - \overline{p}\Delta\gamma \approx \frac{\overline{\gamma}^2 - \overline{p}^2}{\overline{\gamma}} \Delta p_{\parallel} = \frac{1}{\overline{\gamma}} \Delta p_{\parallel}, \quad p'_{\perp} = \Delta p_{\perp}, \tag{3.7.5}$$

where $\Delta\gamma$ is estimated using the identites $\gamma^2 = 1 + \boldsymbol{p}\cdot\boldsymbol{p}$ and $\overline{\gamma}^2 = 1 + \overline{p}^2$:

$$\begin{aligned} (\overline{\gamma} + \Delta\gamma)^2 &= 1 + (p_{\parallel} + \Delta p_{\parallel})^2 + (\Delta p_{\perp})^2 \\ \implies \quad \overline{\gamma}^2 + 2\overline{\gamma}\Delta\gamma + (\Delta\gamma)^2 &= 1 + \overline{p}^2 + 2\overline{p}\Delta p_{\parallel} + (\Delta p_{\parallel})^2 + (\Delta p_{\perp})^2 \\ \implies \quad \overline{\gamma}^2 + 2\overline{\gamma}\Delta\gamma &\approx 1 + \overline{p}^2 + 2\overline{p}\Delta p_{\parallel} \\ \implies \quad \Delta\gamma &\approx \frac{\overline{p}}{\overline{\gamma}}\Delta p_{\parallel}. \end{aligned}$$

Substituting (3.7.5) into (3.7.4), the denominator in (3.7.4) can be approximated by

$$\left|\|\boldsymbol{R}'\|_2^2 + (\boldsymbol{p}'\cdot\boldsymbol{R}')^2\right|^{3/2} \approx \left|R_{\parallel}'^2 + R_{\perp}'^2 + (\frac{1}{\overline{\gamma}}\Delta p_{\parallel} R_{\parallel}' + \Delta p_{\perp} R_{\perp}')^2\right|^{3/2} \approx \|\boldsymbol{R}'\|_2^3,$$

where the assumption $\frac{\Delta p_{\parallel}}{\overline{\gamma}}$, $\Delta p_{\perp} \ll 1$ was used. Therefore, the particle field in $\mathcal{K}'$ is approximately equal to

$$\boldsymbol{E}' \approx \frac{1}{4\pi\epsilon_0}\frac{\gamma'\boldsymbol{R}'}{\|\boldsymbol{R}'\|_2^3}, \quad \boldsymbol{B}' \approx \frac{1}{4\pi\epsilon_0}\frac{\boldsymbol{p}_j \times \boldsymbol{R}'}{\|\boldsymbol{R}'\|_2^3}. \tag{3.7.6}$$

After $\boldsymbol{E}'$ and $\boldsymbol{B}'$ are computed by the treecode, the corresponding particle field in $\mathcal{K}$ can be calculated with the field transformation [42]

$$\boldsymbol{E} = \overline{\gamma}\boldsymbol{E}' - \overline{\boldsymbol{p}} \times \boldsymbol{B}' - \frac{1}{\overline{\gamma}+1}(\overline{\boldsymbol{p}}\cdot\boldsymbol{E}')\overline{\boldsymbol{p}}, \quad \boldsymbol{B} = \overline{\gamma}\boldsymbol{B}' + \overline{\boldsymbol{p}} \times \boldsymbol{E}' - \frac{1}{\overline{\gamma}+1}(\overline{\boldsymbol{p}}\cdot\boldsymbol{B}')\overline{\boldsymbol{p}}. \tag{3.7.7}$$

The procedure discussed above is summarized in Algorithm 3.6 and a schematic of the procedure is illustrated in Figure 3.4. In the practical implementation, instead of (3.7.1), we use

$$x_{\parallel}' = \overline{\gamma}x_{\parallel} \quad \text{and} \quad x_{\perp}' = x_{\perp} \tag{3.7.8}$$

to compute the particle position in the average rest-frame. Because the lab-frame position of particles is given at a common simulation time t, both (3.7.1) and (3.7.8) lead to the same pair-wise distance between particles at the average-rest frame. Therefore, equations (3.7.8) can be applied in the computation of particle field by treecode where only the relative position of particles is of importance.

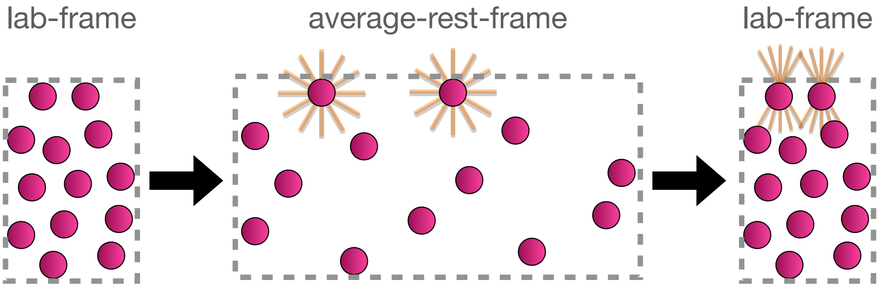


Figure 3.4: A schematic of the treecode based on the transformation to the average rest-frame. The particle's position and momentum are transformed to the average rest-frame and the particle field evaluated in the average rest-frame is then transformed back to the lab-frame.

Algorithm 3.6: Treecode with Average Rest-Frame Technique

```
S: particle cluster
$\boldsymbol{s}$: stretch factor
$N_0$: maximum number of particles in the leaf node
$\eta$: admissibility parameter
$\boldsymbol{E}_i, \boldsymbol{B}_i$: the space-charge field experienced by the $i$-th particle
Function TreecodeAVGRF($S$, $N_0$, $\eta$)
    $\boldsymbol{s} = (1, 1, 1)$
    $S' = \text{lab2avgframe}(S)$ (Algorithm 3.7)
    subdivide($S'$, $\boldsymbol{s}$, $N_0$) (Algorithm 3.2)
    $\boldsymbol{E}_i = \boldsymbol{0}, \boldsymbol{B}_i = \boldsymbol{0} \quad \forall i \in \widehat{S}$
    for $i \in \hat{S}'$ do
        $\boldsymbol{E}'_{\text{temp}}, \boldsymbol{B}'_{\text{temp}} = \text{cluster2p}(\boldsymbol{x}'_i,\ S'\ \boldsymbol{s},\ \eta)$ (Algorithm 3.4)
        compute $\boldsymbol{E}_i, \boldsymbol{B}_i$ using $\boldsymbol{E}'_{\text{temp}}, \boldsymbol{B}'_{\text{temp}}$ and (3.7.7)
    end
    return $\{\boldsymbol{E}_i \mid i \in \widehat{S}\}$, $\{\boldsymbol{B}_i \mid i \in \widehat{S}\}$
end
```

Algorithm 3.7: Transformation of Particle Beam to the Average Rest-Frame

```
S: particle cluster
Function lab2avgframe($S$)
    $\overline{\boldsymbol{p}} = \frac{1}{|S|}(\sum_{i \in \widehat{S}} \boldsymbol{p}_i)$
    $\overline{\gamma} = (1 + \overline{\boldsymbol{p}} \cdot \overline{\boldsymbol{p}})^{1/2}$
    for $i \in \widehat{S}$ do
        $\boldsymbol{x}'_i = \boldsymbol{x}_i + \frac{1}{\overline{\gamma}+1}(\boldsymbol{x}_i \cdot \overline{\boldsymbol{p}})\overline{\boldsymbol{p}}$ (vector form of (3.7.8))
        $\boldsymbol{p}'_i = \boldsymbol{p}_i + \frac{1}{\overline{\gamma}+1}(\boldsymbol{p}_i \cdot \overline{\boldsymbol{p}})\overline{\boldsymbol{p}} - \gamma_i \overline{\boldsymbol{p}}$ (vector form of (3.7.2))
        $\gamma'_i = (1 + \boldsymbol{p}_i \cdot \boldsymbol{p}_i)^{1/2}$
    end
    return $S' = \{\boldsymbol{x}'_i \,|\, \forall i \in \widehat{S}\}$
end
```

3.8 Numerical Results and Discussion

To understand the performance of the different treecodes discussed in this study, we performed numerical simulations for particle beams with different configurations. All the treecodes and algorithms discussed in this study have been implemented in a package called NChargedBodyTreecode.jl using the Julia programming language [46]. We refer to the procedure proposed by Wang *et al.* [83] to compute the effective Lorentz factor and the effective momentum of the macro particle: the Lagrangian polynomials are implemented based on the second form of the barycentric formula [86], and the interpolation points (*i.e.*, the positions of the macro particles) are generated by Chebyshev points of the second kind [83].

3.8.1 Particle beam with single momentum

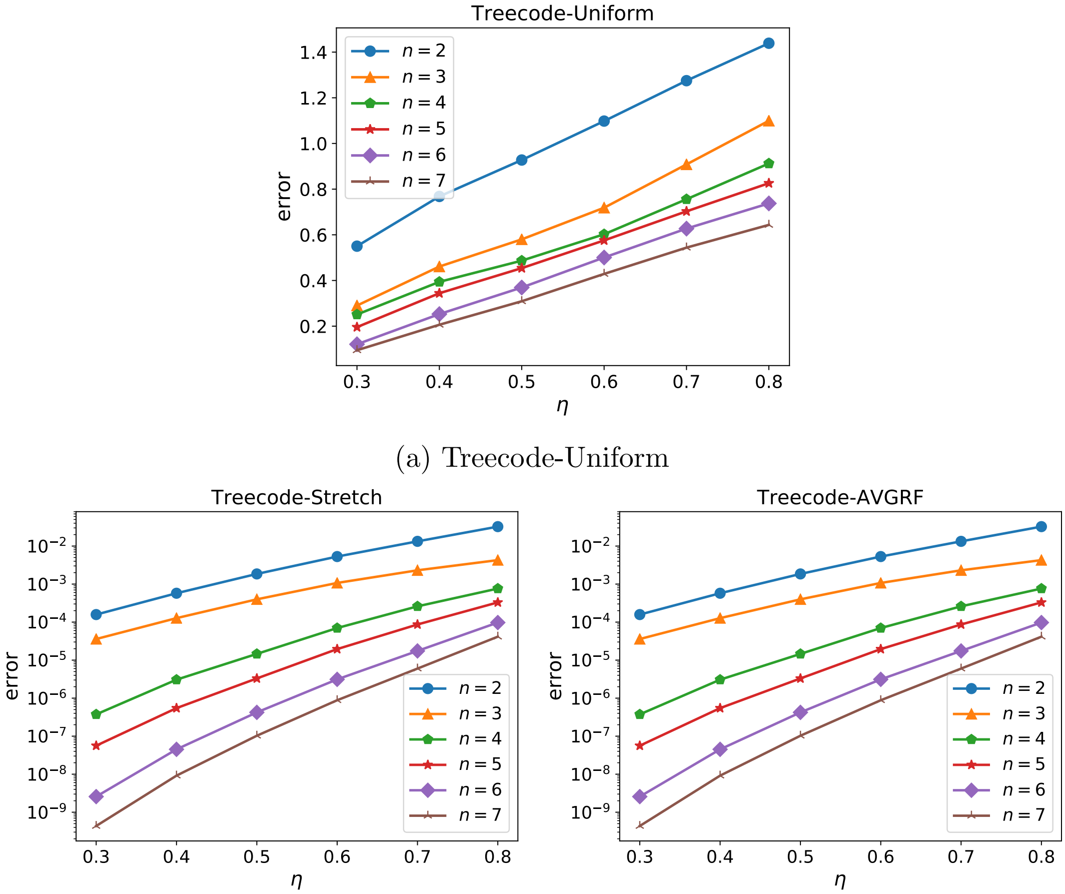
We first consider particles randomly uniformly distributed in the unit cube $[0,1]^3$ where each particle has the same momentum $\boldsymbol{p} = (0, 0, p_0)$ with $p_0 = (\gamma^2 - 1)^{1/2}$. The space-charge field (both E- and B-field) experienced by each particle in the system is computed by the following methods:

1. brute-force method,
2. a treecode called "Treecode-Uniform" using the conventional admissibility condition (Algorithm 3.5 with $\boldsymbol{s} = (1, 1, 1)$),
3. a treecode called "Treecode-Stretch" using the stretched admissibility condition and the subdivision strategy with stretch (Algorithm 3.5 with $\boldsymbol{s} = (1, 1, \overline{\gamma})$),
4. a treecode called "Treecode-AVGRF" using the average rest-frame approach (Algorithm 3.6).

The result from the brute-force method serves as a reference to evaluate the speedup and the relative error of different treecodes. The measured error is the maximal relative error in the electrical and magnetic field,

$$\text{error} := \max_{\boldsymbol{f} \in \{\boldsymbol{E}, \boldsymbol{B}\}} \left(\sum_{i=1}^{N} \|\boldsymbol{f}_i^t - \boldsymbol{f}_i^b\|_2^2 \Big/ \sum_{i=i}^{N} \|\boldsymbol{f}_i^b\|_2^2 \right)^{1/2},$$

where N is the number of particles in the system. The space-charge fields $\boldsymbol{f}_i^t$ and $\boldsymbol{f}_i^b$ experienced by the i-th particle are computed by the treecode and the brute-force method, respectively. The result is shown in Figure 3.5. As illustrated by Figure 3.5a, Treecode-Uniform has large errors and this supports our argument in the beginning that the conventional treecode cannot be used directly for relativistic particle beams. On the other hand, as demonstrated in Figure 3.5b and Figure 3.5c, the two proposed treecodes can effectively treat the problem of a relativistic particle beam. Also, we can observe that the errors computed by Treecode-Stretch and Trecode-AVGRF look the same. The reason is that both methods result in equivalent ways to subdivide the particle clusters and to determine the admissible clusters. In Treecode-AVGRF, the particles' spatial distribution gets boosted along the longitudinal direction after the relativity transformation depicted in Figure 3.4. In Treecode-Stretch, the particles' spatial distribution stays unchanged in the lab-frame but the subdivision and the admissibility condition involve a stretch factor related to the Lorentz boost.



(a) Treecode-Uniform

(b) Treecode-Stretch

(c) Treecode-AVGRF

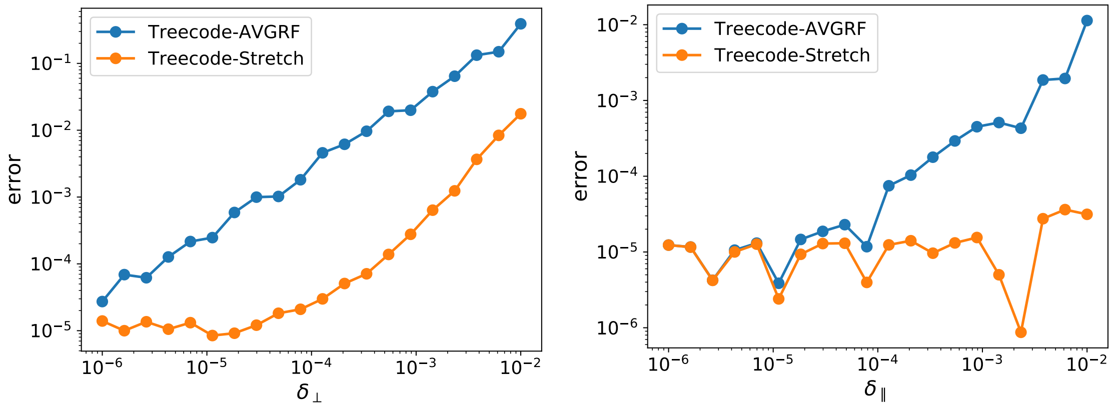
Figure 3.5: The accuracy of different treecodes for the particle beam with mono-momentum. The number of particles is 8×10^4 and $\gamma = 50$. Here, n and η are the interpolation degree (3.4.5) and the admissibility parameter (3.5.4), respectively. The interpolation degree is $n = 2, 3, 4, 5, 6, 7$ and the maximum number of the particles in the leaf cluster is chosen as $N_0 = (n + 1)^3$.

3.8.2 Particle beam with momentum spread

In practical scenarios, beam divergence and energy spread are typically present in particle beams. These two quantities are related to the momentum distribution of a particle beam in transverse and longitudinal directions, respectively. Therefore it is also important to know how the momentum distribution influences the accuracy of treecodes. We consider a particle beam randomly distributed over a momentum distribution defined as

$$f_{\boldsymbol{p}}(p_x, p_y, p_z) \propto \exp\left(-\frac{p_x^2 + p_y^2}{2(\delta_\perp p_0)^2}\right) \exp\left(-\frac{(p_z - p_0)^2}{2(\delta_\parallel p_0)^2}\right) \text{ with } p_0 = (\gamma^2 - 1)^{1/2}.$$

Here, $\delta_\perp$ and $\delta_\parallel$ are equivalent to RMS (root mean square) divergence and RMS energy spread for a relativistic particle beam with the paraxial approximation (*i.e.*, $p_{\{x,y\}} \ll p_0$). The positions of the particles are uniformly randomly distributed over the unit cube $[0, 1]^3$. The result is shown in Figure 3.6. Our numerical results reveal that treecode-AVGRF is less accurate than Treecode-Stretch when a momentum spread is considered in the particle beam.



(a) transverse

(b) longitudinal

Figure 3.6: The accuracy of Treecode-AVGRF and Treecode-Stretch for particle beams with (a) transverse momentum spread and (b) longitudinal momentum spread. The number of particles is 8×10^4 and $\gamma = 50$. Each simulation is performed with admissibility parameter $\eta = 0.5$, interpolation degree $n = 4$, and maximum number of particles in the leaf clusters $N_0 = 256$.

3.8.3 The problem of Treecode-AVGRF

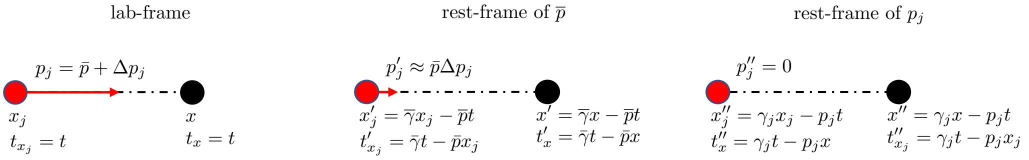
Our numerical results show that Treecode-AVGRF is less accurate compared to Treecode-Stretch when a momentum spread is considered in the particle beam. The reason is due to the inaccurate computation of the target-source distance in the average rest-frame. To explain this, we consider a particle moving with a momentum p_j in the lab-frame $\mathcal{K}$ (Figure 3.7a). If the positions of the particle x_j and the target point x are both measured at the same time $t_{x_j} = t_x = t$, the distance between these two points can be calculated directly using $x - x_j$. If we transform the system to a reference frame $\mathcal{K}'$ moving with a momentum $\overline{p}$ (Figure 3.7b), the events observed in $\mathcal{K}'$ will become

$$\begin{cases} x' = \overline{\gamma}x - \overline{p}t, \\ t'_x = \overline{\gamma}t - \overline{p}x, \end{cases} \qquad \begin{cases} x'_j = \overline{\gamma}x_j - \overline{p}t, \\ t'_{x_j} = \overline{\gamma}t - \overline{p}x_j, \end{cases} \qquad \text{and} \qquad p'_j \approx \overline{p}\Delta p_j.$$

Here, we assume $p_j = \overline{p} + \Delta p_j$ with $\overline{p} \gg \Delta p_j$ and the space-time quantities are normalized such that the speed of light is $c_0 = 1$. It is problematic if we compute the distance directly by $x' - x'_j$ because these two events are measured at different times (*i.e.*, $t'_x \neq t'_{x_j}$) and the particle keeps moving during the measurement. To get a correct distance, as demonstrated in Figure 3.7c, we need to transform the system to the particle's rest-frame $\mathcal{K}''$ that satisfies

$$\begin{cases} x'' = \gamma_j x - p_j t, \\ t''_x = \gamma_j t - p_j x, \end{cases} \qquad \begin{cases} x''_j = \gamma_j x_j - p_j t, \\ t''_{x_j} = \gamma_j t - p_j x_j, \end{cases} \qquad \text{and} \qquad p''_j = 0.$$

In the reference frame $\mathcal{K}''$, although two events are still measured at different times (*i.e.*, $t''_x \neq t''_{x_j}$), we can still compute the distance with $x'' - x''_j$ because the particle is stationary during the measurement. However, this approach is not applicable to treecode because the transformations of the whole particle beam to each particle's rest-frame have a complexity of $\mathcal{O}(N^2)$.


lab-frame
$p_j = \bar{p} + \Delta p_j$
x_j
$t_{x_j} = t$
x
$t_x = t$
rest-frame of $\overline{p}$
$p'_j \approx \bar{p}\Delta p_j$
$x'_j = \overline{\gamma} x_j - \overline{p} t$
$t'_{x_j} = \bar{\gamma} t - \bar{p} x_j$
$x' = \overline{\gamma} x - \overline{p} t$
$t'_x = \bar{\gamma} t - \bar{p} x$
rest-frame of p_j
$p''_j = 0$
$x''_j = \gamma_j x_j - p_j t$
$t''_x = \gamma_j t - p_j x$
$x'' = \gamma_j x - p_j t$
$t''_{x_j} = \gamma_j t - p_j x_j$

(a) lab-frame $\mathcal{K}$ (b) rest-frame $\mathcal{K}'$ (c) rest-frame $\mathcal{K}''$

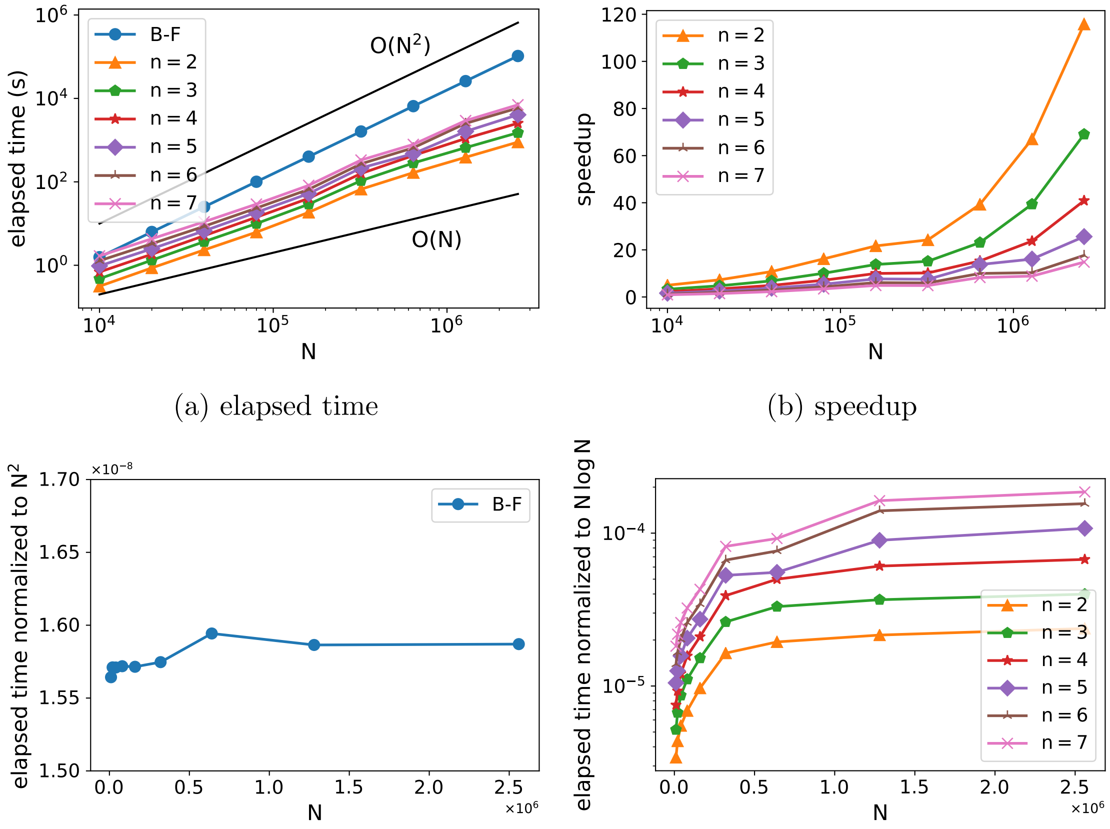
Figure 3.7: A 1D illustration of the distance between one point (black dot) and one particle (red dot) moving with $p_j = \overline{p} + \Delta p_j$. The system is viewed in (a) the lab-frame ($\mathcal{K}$), (b) the rest-frame of $\overline{p}$ ($\mathcal{K}'$) and (c) the rest-frame of p_j ($\mathcal{K}''$).

3.8.4 Performance

To understand the speedup achieved with our treecode, we compare the performance of the brute-force method and the treecode in Figure 3.8. Here, Treeecode-Stretch is selected because it has the best accuracy in all the scenarios discussed before. As shown in Figure 3.8a and Figure 3.8c, the brute-force method scales with $\mathcal{O}(N^2)$. The treecode scales between $\mathcal{O}(N^2)$ and $\mathcal{O}(N \log N)$ (Figure 3.8a) and approaches to its theoretical complexity $\mathcal{O}(N \log N)$ (Figure 3.8d). Also, we can observe that the speedup of the treecode increases with the number of particles (Figure 3.8b). Here, the speedup refers to the ratio of the elapsed time of the brute-force method to the elapsed time of the treecode.

3.9 Summary

In this chapter, based on the Lagrangian interpolation, we formulate a treecode for computing the relativistic space-charge field. We propose two approaches to control the interpolation error: Treecode-Stretch and Treecode-AVGRF. Our numerical result shows that Treecode-Stretch has better accuracy than Treecode-AVGRF for particle beams with momentum spread. Also, the performance of our treecode scales like $O(N \log N)$ as expected by the theory. Compared to FMM, which usually relies on a sophisticated data structure, the proposed treecode is easy to implement and intrinsically parallelizable; it may be especially suitable for building in-house solvers for the study of space-charge effects from relativistic beams.



(a) elapsed time

(b) speedup

(c) normalized elapsed time

(d) normalized elapsed time

Figure 3.8: The performance of brute-force and treecode methods. Figure 3.8a shows the elapsed time used by brute-force method (B-F) and Treecode-Stretch to evaluate the space-charge field of increasing numbers of particles N. Figure 3.8b shows the speedup of Treecode-Stretch relative to the brute-force method. Figure 3.8c shows the elapsed time of brute-force method normalized to N^2. Figure 3.8d shows the elapsed times of treecode method normalized $N \log N$. The treecode is performed with the fixed admissibility parameter $\eta = 0.5$ and different interpolation degrees $n = 2, 3, 4, 5, 6, 7$. The maximum number of particles in the leaf cluster is chosen as $N_0 = (n+1)^3$.

Chapter 4

Relativistic Space-Charge Force Calculation by Interpolation-Based Fast-Multipole Method

In the previous chapter, we formulated an interpolation-based treecode for the efficient computation of the relativistic space-charge field. The treecode considers the particle-cluster interaction and reduces the complexity for the space-charge field calculation from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ with the total number of particles N. In the fast summation methods, there also exists a more sophisticated approach, called fast-multipole method (FMM). The FMM considers the cluster-cluster interaction and can further reduce the complexity to $\mathcal{O}(N)$. In this chapter, based on the formulation of the treecode discussed in Chapter 3, we discuss an interpolation-based FMM for the efficient computation of the relativistic space-charge force field.

4.1 The Idea of FMM

In this section, we give a short overview of the interpolation-based FMM [87, 88]. Consider two particle-clusters S_t and S_s. The total force-field f from the source particles in the cluster S_s acting on a target particle with index i and position $\boldsymbol{x}_i \in S_t$ through an interaction kernel $g(\boldsymbol{\cdot}, \boldsymbol{\cdot})$ can be modeled as

$$f(\boldsymbol{x}_i) = \sum_{j \in \widehat{S}_s} g(\boldsymbol{x}_i, \boldsymbol{x}_j) m_j. \tag{4.1.1}$$

Here, $\widehat{S}_t$ and $\widehat{S}_s$ represent the index sets of the particles in S_t and S_s, respectively. The symbol m_j is the physical quantity of the source particle with index j. Although the actual meaning of m_j depends on the physics problem we investigate, without loss of generality, we call it mass throughout this section.

The idea of using FMM for a fast evaluation of the summation in (4.1.1) is based on the approximation of the kernel function by interpolating both the target variable $\boldsymbol{x}_i$ and the source variable $\boldsymbol{x}_j$

$$\boldsymbol{g}(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx \sum_{\boldsymbol{\mu}} \sum_{\boldsymbol{\nu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\nu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j), \tag{4.1.2}$$

where the Lagrange basis polynomials (3.4.5) over the bounding boxes of S_t and S_s are defined as

$$\ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) := \ell^x_{S_t,\mu_1}(x_i) \cdot \ell^y_{S_t,\mu_2}(y_i) \cdot \ell^z_{S_t,\mu_3}(z_i),$$
$$\ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j) := \ell^x_{S_s,\nu_1}(x_j) \cdot \ell^y_{S_s,\nu_2}(y_j) \cdot \ell^z_{S_s,\nu_3}(z_j),$$

with the corresponding interpolation points

$$\boldsymbol{\xi}_{S_t\,\boldsymbol{\mu}} := (\xi_{S_t,\mu_1}, \xi_{S_t,\mu_2}, \xi_{S_t,\mu_3}) \quad \text{and} \quad \boldsymbol{\xi}_{S_s\,\boldsymbol{\nu}} := (\xi_{S_s,\nu_1}, \xi_{S_s,\nu_2}, \xi_{S_t,\nu_3}).$$

Substituting (4.1.2) into (4.1.1), we have

$$\begin{aligned} f(\boldsymbol{x}_i) &= \sum_{j\in\widehat{S}_s} g(\boldsymbol{x}_i, \boldsymbol{x}_j) m_j \\ &\approx \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \sum_{\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) \underbrace{\sum_{j\in\widehat{S}_s} \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j) m_j}_{=:M_{S_s,\boldsymbol{\nu}}} \\ &= \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \underbrace{\sum_{\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) M_{S_s,\boldsymbol{\nu}}}_{=:L_{S_t,\boldsymbol{\mu}}} \\ &= \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S_t,\boldsymbol{\mu}}. \end{aligned} \tag{4.1.3}$$

By observing (4.1.3), we can identify the four of FMM kernels:

- P2M (point to multipole): The micro particles in the cluster S_s are aggregated into a few macro particles and the mass of each macro particle ($M_{S_s,\boldsymbol{\nu}}$, also called multipole) can be computed by

$$M_{S_s,\boldsymbol{\nu}} := \sum_{j\in\widehat{S}_s} \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j) m_j. \tag{4.1.4}$$

- M2L (multipole to local): The multipoles of the source cluster are used to evaluate the force-field acting on the macro particles ($L_{S_t,\boldsymbol{\mu}}$, also called local field) in the target cluster S_t

$$L_{S_t,\mu} := \sum_{\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) M_{S_s,\boldsymbol{\nu}}. \tag{4.1.5}$$

- L2P (local to point): In the target cluster, the effective force-field acting on the macro particles are transferred to the micro particle at $\boldsymbol{x}_i$ by

$$f(\boldsymbol{x}_{\boldsymbol{i}}) = \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S_t,\boldsymbol{\mu}}. \tag{4.1.6}$$

- P2P (point to point): If S_t and S_s do not fulfill an admissibility condition (ADMC, also called multipole acceptance criteria MAC in some literature [83, 89]) so that (4.1.2) is not applicable, the force-field can be calculated directly by

$$f(\boldsymbol{x}_i) = \sum_{j\in\widehat{S}_s} g(\boldsymbol{x}_i, \boldsymbol{x}_j) m_j \quad \forall i \in \widehat{S}_t. \tag{4.1.7}$$

This also applies for the case $S_t = S_s = S$ where $i, j \in \widehat{S}$ and $i \neq j$.

One main feature of the FMM is the consideration of the cluster-cluster interaction (M2L) through macro particles; and hence, the total number of operations for the evaluation of force-fields can be drastically reduced. In the FMM, we first partition all particles in the system into a hierarchy of clusters (cluster tree). If we directly use (4.1.4) to compute the multipole of each cluster, the number of operations for computing the multipoles of the whole cluster tree is

$$\text{const} \cdot N \cdot N_r \cdot \log\left(\frac{N}{N_r}\right) \tag{4.1.8}$$

with the assumption that the number of macro particles used for the approximation and the number of micro particles contained in the leaf cluster are both N_r. To reduce the total number of operations for computing the multipoles, we can make use of the following property of polynomial interpolation stated in Theorem 4.1.1.

Theorem 4.1.1. *If $P(x)$ is a polynomial function of degree n, we have*

$$P(x) = \sum_{k=0}^{n} P(\xi_{S,k})\ell_{S,k}(x) \quad \forall x \in \mathbb{R} \tag{4.1.9}$$

with $\ell_{S,k}(x)$ being Lagrange basis for polynomials of degree $\leq n$ for the interpolation nodes $\xi_{S,k} \in S$.

This equality can be seen by the fundamental theorem of algebra since both LHS and RHS have the same values at the $n+1$ points $\{\xi_{S,k} \mid k = 0, \ldots, n\}$ and the RHS is a polynomial of degree n. By using Theorem 4.1.1, we can introduce two further procedures and kernels of FMM:

- Upward Pass: A source cluster S is subdivided into a cluster tree of the depth κ. The multipoles of each cluster can be computed by the multipoles of its children clusters because

$$\begin{aligned}
M_{S,\boldsymbol{\nu}} &= \sum_{j\in\widehat{S}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)m_j \\
&= \sum_{s'\in\text{children}(S)} \sum_{j\in\widehat{s'}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)m_j \\
&\overset{(4.1.9)}{=} \sum_{s'\in\text{children}(S)} \sum_{j\in\widehat{s'}} \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{s',\boldsymbol{\nu}'})\ell_{s',\boldsymbol{\nu}'}(\boldsymbol{x}_j)m_j \\
&= \sum_{s'\in\text{children}(S)} \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{s',\boldsymbol{\nu}'}) \sum_{j\in\widehat{s'}} \ell_{s',\boldsymbol{\nu}'}(\boldsymbol{x}_j)m_j \\
&= \sum_{s'\in\text{children}(S)} \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{s',\boldsymbol{\nu}'})M_{s',\boldsymbol{\nu}'}.
\end{aligned} \tag{4.1.10}$$

  Equation (4.1.10) is the formula of the M2M (multipole to multipole) kernel. In FMM, the multipoles of the source cluster tree can be updated by a procedure called upward pass. In the upward pass, the multipoles of the leaf clusters are first evaluated with P2M (4.1.4); and then, we start from the second deepest level of the cluster tree (*i.e.*, level $\kappa - 1$) and apply M2M (4.1.10) to compute the multipoles of each cluster in each level (level by level). If the upward pass is used for the computation of the macroparticles of the whole cluster tree, the number of operations shown in (4.1.8) can be reduced to

$$\text{const} \cdot N_r \cdot N.$$

  The reason is because a cluster tree contains N/N_r clusters and the number of operations for computing the macroparticles of each cluster is proportional to N_r^2.

- Downward Pass: A target cluster S is subdivided into a hierarchy of clusters of the depth κ and each target particle (say particle i) will be contained in a sequence of clusters $\{S^l \mid l = 0, \ldots, \kappa\}$ of each level l with $S^{l+1} \subset S^l$ and $S^0 = S$. The force-field on the target particle i can be calculated by

$$f(\boldsymbol{x}_i) = \sum_{l=0}^{\kappa} \sum_{\boldsymbol{\mu}} \ell_{S^l,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^l,\boldsymbol{\mu}} = \sum_{\boldsymbol{\mu}} \mathcal{L}_{S^\kappa,\boldsymbol{\mu}} \ell_{S^\kappa,\boldsymbol{\mu}}(\boldsymbol{x}_i). \tag{4.1.11}$$

Here, we define the "cumulative local field" $\mathcal{L}_{S^l,\boldsymbol{\mu}}$ which follows from the recursive relation

$$\mathcal{L}_{S^l,\boldsymbol{\mu}} := L_{S^l,\boldsymbol{\mu}} + \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S^{l-1},\boldsymbol{\mu}'} \cdot \ell_{S^{l-1},\boldsymbol{\mu}'}(\boldsymbol{\xi}_{S^l,\boldsymbol{\mu}}) \quad \text{with} \quad \mathcal{L}_{S^0,\boldsymbol{\mu}} := L_{S^0,\boldsymbol{\mu}}. \tag{4.1.12}$$

Equation (4.1.11) can be proved by using (4.1.9), (4.1.12) and the mathematical induction (cf. Section C.1). Therefore, during the downward pass of FMM, we first perform L2L (4.1.12) to calculate the cumulative local fields of the deepest-level cluster S^κ; afterwards, we transfer $\mathcal{L}_{S^\kappa,\mu}$ to the target particles contained in S^κ via L2P (4.1.11).

4.2 FMM Formulation for the Efficient Computation of the Relativistic Space-Charge Field

Consider a relativistic charged particle beam moving in z-direction. Inside this particle beam, the space-charge field from a source particle with the position $\boldsymbol{x}_j$ exerting to a target particle with the position $\boldsymbol{x}_i$ can be approximately written as

$$\boldsymbol{E}(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx \frac{q}{4\pi\epsilon_0} \gamma_j \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{x}_j) \quad \text{and} \quad \boldsymbol{B}(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx \frac{q}{4\pi\epsilon_0 c_0} \boldsymbol{p}_j \times \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{x}_j), \tag{4.2.1}$$

with the kernel function called "relativistic kernel" (3.4.4)

$$\boldsymbol{g}(\boldsymbol{x}_i, \boldsymbol{x}_j) := \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\Big((x_i - x_j)^2 + (y_i - y_j)^2 + \overline{\gamma}^2 (z_i - z_j)^2 \Big)^{3/2}},$$

where $\gamma_j = 1/\sqrt{1 - \|\boldsymbol{\beta}_j\|_2^2}$ and $\boldsymbol{p}_j = \gamma_j \boldsymbol{\beta}_j$ are the Lorentz factor and normalized momentum of the particle, respectively, with $\boldsymbol{\beta}_j$ the particle velocity normalized to the speed of light c_0. Here, $\overline{\gamma}$ is the average Lorentz factor and is defined as $\overline{\gamma}^2 = 1 + \overline{\boldsymbol{p}} \cdot \overline{\boldsymbol{p}}$ with the average momentum of the particle beam $\overline{\boldsymbol{p}}$. Throughout this study, we assume all particles in the particle beam are the same type with charge q.

Given a target particle with the position $\boldsymbol{x}_i$ contained in a target cluster S_t, the space-charge field from all the particles in the cluster S_s experienced by this target particle can be computed approximately by applying (4.1.2) to (4.2.1)

$$\begin{aligned}
\sum_{j \in \widehat{S}_s} \boldsymbol{E}(\boldsymbol{x}_i, \boldsymbol{x}_j) &\approx \frac{q}{4\pi\epsilon_0} \sum_{\boldsymbol{\mu}} \sum_{\boldsymbol{\nu}} \sum_{j \in \widehat{S}_s} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j) \gamma_j \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) \\
&= \frac{q}{4\pi\epsilon_0} \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \boldsymbol{E}_{S_t,\boldsymbol{\mu}}, \\
\sum_{j \in \widehat{S}_s} \boldsymbol{B}(\boldsymbol{x}_i, \boldsymbol{x}_j) &\approx \frac{q}{4\pi\epsilon_0 c_0} \sum_{\boldsymbol{\mu}} \sum_{\boldsymbol{\nu}} \sum_{j \in \widehat{S}_s} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j) \boldsymbol{p}_j \times \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) \\
&= \frac{q}{4\pi\epsilon_0 c_0} \sum_{\boldsymbol{\mu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i) \boldsymbol{B}_{S_t,\boldsymbol{\mu}}.
\end{aligned} \tag{4.2.2}$$

Here, we introduce the effective Lorentz factor and the effective momentum for a macro particle with the position $\boldsymbol{\xi}_{\boldsymbol{\nu}}$ and index $\boldsymbol{\nu}$ in the cluster S_s as

$$\gamma_{S_s,\boldsymbol{\nu}} := \sum_{j\in\widehat{S}_s} \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j)\gamma_j \quad \text{and} \quad \boldsymbol{p}_{S_s,\boldsymbol{\nu}} := \sum_{j\in\widehat{S}_s} \ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j)\boldsymbol{p}_j.$$

Similarly, the effective electric and magnetic field experienced by a macro particle with the position $\boldsymbol{\xi}_{\boldsymbol{\mu}}$ and index $\boldsymbol{\mu}$ in the target cluster S_t are defined as

$$\boldsymbol{E}_{S_t,\boldsymbol{\mu}} := \sum_{\boldsymbol{\nu}} \gamma_{S_s,\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}) \quad \text{and} \quad \boldsymbol{B}_{S_t,\boldsymbol{\mu}} := \sum_{\boldsymbol{\nu}} \boldsymbol{p}_{S_s,\boldsymbol{\nu}} \times \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}}).$$

4.3 Admissibility Condition for Cluster-Cluster Interaction of the Relativistic Kernel

In the previous section, we used Lagrangian interpolation to approximate the space-charge field on a target particle in a relativistic particle beam. It is also of importance to know when this approximation can be applied. To answer this question, we investigate the interpolation error bound of the relativistic kernel

$$\|\boldsymbol{g}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \widetilde{\boldsymbol{g}}(\boldsymbol{x}_i, \boldsymbol{x}_j)\|_{\infty, Q_t\times Q_s} \le B_t + B_s. \tag{4.3.1}$$

Here, B_t and B_s are the interpolation error bounds with respect to the target variable $\boldsymbol{x}_i := (x_i, y_i, z_i)$ and the source variable $\boldsymbol{x}_j := (x_j, y_j, z_j)$:

$$\begin{aligned} B_t :=& \text{const} \cdot \sum_{k\in\{x_i,y_i,z_i\}} (b_k - a_k)^{n+1} \cdot \frac{\|\partial_k^{n+1}\boldsymbol{g}(\cdot, \boldsymbol{x}_j)\|_{\infty,Q_t\times Q_s}}{(n+1)!}, \\ B_s :=& \text{const} \cdot \sum_{k\in\{x_j,y_j,z_j\}} (b_k - a_k)^{n+1} \cdot \frac{\|\partial_k^{n+1}\boldsymbol{g}(\boldsymbol{x}_i, \cdot)\|_{\infty,Q_t\times Q_s}}{(n+1)!}, \end{aligned}$$

where the bounding boxes of S_t and S_s are

$$\begin{aligned} Q_t &= [a^t_{x_i}, b^t_{x_i}] \times [a^t_{y_i}, b^t_{y_i}] \times [a^t_{z_i}, b^t_{z_i}] \subset \mathbb{R}^3, \\ Q_s &= [a^s_{x_j}, b^s_{x_j}] \times [a^s_{y_j}, b^s_{y_j}] \times [a^s_{z_j}, b^s_{z_j}] \subset \mathbb{R}^3. \end{aligned}$$

Following the similar analysis as in Section 3.5, we can derive the error bounds of B_t and B_s respectively as

$$\begin{aligned} B_t \le& \text{const} \cdot \sum_{k\in\{x_i,y_i,z_i\}} s_k^{n+1}(b_k - a_k)^{n+1} \cdot \frac{1}{\|\boldsymbol{s}\circ(\boldsymbol{x}_i - \boldsymbol{x}_j)\|_\infty^{n+3}} \le \frac{\text{const}}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t, S_s)^2} \frac{\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_t)^{n+1}}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t, S_s)^{n+1}}, \\ B_s \le& \text{const} \cdot \sum_{k\in\{x_j,y_j,z_j\}} s_k^{n+1}(b_k - a_k)^{n+1} \cdot \frac{1}{\|\boldsymbol{s}\circ(\boldsymbol{x}_i - \boldsymbol{x}_j)\|_\infty^{n+3}} \le \frac{\text{const}}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t, S_s)^2} \frac{\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_s)^{n+1}}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t, S_s)^{n+1}}, \end{aligned}$$

where we define a stretched vector $\boldsymbol{s} := (s_x, s_y, s_z) = (1, 1, \overline{\gamma})$ and the symbol

$$\circ : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$$

denotes the component-wise product of two vectors (3.5.5). Here, the stretched diameter of a cluster (same as defined in Section 3.5) and the stretched distance between two clusters S_t and S_s are

$$\begin{aligned} \overset{(s_x,s_y,s_z)}{\operatorname{diam}}(S) &:= \max_{\boldsymbol{x}_i,\boldsymbol{x}_j\in S} \|(s_x,s_y,s_z)\circ(\boldsymbol{x}_i-\boldsymbol{x}_j)\|_2, \\ \overset{(s_x,s_y,s_z)}{\operatorname{dist}}(S_t,S_s) &:= \min_{\substack{\boldsymbol{x}_i\in S_t\\ \boldsymbol{x}_j\in S_s}} \|(s_x,s_y,s_z)\circ(\boldsymbol{x}_i-\boldsymbol{x}_j)\|_2. \end{aligned} \tag{4.3.2}$$

Therefore, the interpolation error (4.3.1) can be bounded by

$$\begin{aligned} \|\boldsymbol{g}(\boldsymbol{x}_i,\boldsymbol{x}_j)-\widetilde{\boldsymbol{g}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\|_{\infty,Q_t\times Q_s} &\le \text{const}\cdot\frac{\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_t)^{n+1}+\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_s)^{n+1}}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t,S_s)^{n+1}} \\ &\le \text{const}\cdot\left(\frac{\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_t)+\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_s)}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t,S_s)}\right)^{n+1} \\ &\le \text{const}\cdot\left(\frac{\max\left(\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_t),\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_s)\right)}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t,S_s)}\right)^{n+1}, \end{aligned}$$

and we can define an admissibility condition for the cluster-cluster interaction by

$$\frac{\max\left(\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_t),\overset{(1,1,\overline{\gamma})}{\operatorname{diam}}(S_s)\right)}{\overset{(1,1,\overline{\gamma})}{\operatorname{dist}}(S_t,S_s)} < \eta$$

with some admissibility parameter $\eta\in\mathbb{R}_{>0}$. This is an extension of the stretched admissibility condition for the particle-cluster interaction proposed in Chapter 3 from the particle-cluster to the cluster-cluster type of interaction.

Besides deriving the stretched admissibility condition for the relativistic kernel, it is possible to bypass this mathematical analysis by using a relativity transformation. In the rest frame of the particle beam, the kernel function $\boldsymbol{g}$ is approximately equal to the electrostatic kernel and the conventional admissibility condition can be used for controlling the interpolation error. However, as discussed in Chapter 3, this approach can result in a larger error when a particle beam with a larger momentum spread is considered because the distance between each target-source pair in the rest frame is not correctly evaluated. Therefore, this approach will not be discussed in this chapter.

4.4 The Procedure of FMM

With the FMM kernels introduced in Section 4.1 and the stretched admissibility condition for the cluster-cluster type of interaction derived in Section 4.3, we can extend our previous interpolation-based treecode proposed in Chapter 3 to FMM for the calculation of the relativistic space-charge field. The proposed FMM (Algorithm 4.11) for the calculation of the space-charge field can be summarized by the following procedures:

1. A cluster tree is constructed from the particles in the system (Algorithm 4.1)

2. In the upward pass (Algorithm 4.4), the multipoles of each leaf cluster are computed by P2M (Algorithm 4.2) and then are transferred to the multipoles of their ascendants by M2M (Algorithm 4.3).
3. A list of interaction pairs is determined dynamically by the dual tree traversal (Algorithm 4.10) where the pointers of the target and source are initially pointed to the root cluster. For the interaction pair fulfilling the admissibility condition (Algorithm 4.7), the local field of the target cluster is computed by M2L (Algorithm 4.8). For the interaction pair of leaf clusters, the force-fields on the particles of the target cluster are computed by P2P (Algorithm 4.9). In our implementation, instead of using (4.3.2), we adapt a different definition to compute the stretched diameter and the stretched distance (Algorithm 4.5 and Algorithm 4.6) because of their simplicity in the practical implementation.
4. In the downward pass (Algorithm 4.14), the cumulative local fields of each cluster are transferred to their descendants by L2L (Algorithm 4.12) and the cumulative local fields of each leaf cluster are transferred to the force-field of its member particles by L2P (Algorithm 4.13).

A schematic comparison of treecode and FMM is illustrated in Figure 4.4. In the treecode (Figure 4.4a), we interpolate the source variable $\boldsymbol{x}_j$ of the kernel function so that we can cluster the source particles and build up the effective masses of each cluster. The force-field of each particle in the target cluster is evaluated by each independent traversal of the source cluster tree and by investigating the particle-cluster interaction. In the FMM (Figure 4.4b), we interpolate both the target variable $\boldsymbol{x}_i$ and the source variable $\boldsymbol{x}_j$ of the kernel function so that the target particles and source particles can be clustered and the corresponding effective force-fields and masses of each cluster can be built up. The force-field on a target particle is transferred from the effective force-fields of the clusters which are computed by cluster-cluster interaction through the traversal of the target cluster tree and the source cluster tree simultaneously.

The pseudocodes for the algorithms are presented with the following global variables:

$\boldsymbol{E}_i$ the electric field experienced by the i-th particle,

$\boldsymbol{B}_i$ the magnetic field experienced by the i-th particle,

$\gamma_{S,\boldsymbol{\nu}}$ the effective Lorenz factor of a macro particle with index $\boldsymbol{\nu}$ in the cluster S,

$\boldsymbol{p}_{S,\boldsymbol{\nu}}$ the effective momentum of a macro particle with the index $\boldsymbol{\nu}$ in the cluster S,

$\boldsymbol{E}_{S,\boldsymbol{\nu}}$ the total electric field acting on a macro particle with the index $\boldsymbol{\nu}$ in the cluster S,

$\boldsymbol{B}_{S,\boldsymbol{\nu}}$ the total magnetic field acting on a macro particle with the index $\boldsymbol{\nu}$ in the cluster S.

The algorithms described above are implemented as a solver in the Julia programming language [46].

4.5 Results

To understand the performance of our proposed FMM, we first demonstrate a plot of elapsed time against error for the simulations with different FMM parameters in Figure 4.1. In each simulation, 1.28×10^6 particles are randomly uniformly distributed in the unit cube $[0,1]^3$ and each particle has the same momentum $\boldsymbol{p} =$

Algorithm 4.1: Subdivision of Particle Cluster

```
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$N_0$: maximum number of particles in the leaf node
Function subdivide($S$, $\boldsymbol{s}$, $N_0$)
    if $|S| > N_0$ then
        $k$ = direction4split($S, \boldsymbol{s}$) (Algorithm 3.3)
        $k_{\text{split}} = \lfloor |S|/2 \rfloor$-th largest element of $\{k_i \mid i \in \widehat{S}\}$
        $S_1 = \{\boldsymbol{x}_i \mid k_i \leq k_{\text{split}}, i \in \widehat{S}\}$
        $S_2 = \{\boldsymbol{x}_i \mid k_i > k_{\text{split}}, i \in \widehat{S}\}$
        children(S) = $\{S_1, S_2\}$
        subdivide($S_1$, $\boldsymbol{s}$, $N_0$)
        subdivide($S_2$, $\boldsymbol{s}$, $N_0$)
    else
        children(S) = $\emptyset$
    end
end
```

Algorithm 4.2: P2M

```
$S$: particle cluster
Function P2M($S$)
    $\gamma_{S,\boldsymbol{\nu}} = \sum_{j \in \widehat{S}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\gamma_j$
    $\boldsymbol{p}_{S,\boldsymbol{\nu}} = \sum_{j \in \widehat{S}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)\boldsymbol{p}_j$
end
```

Algorithm 4.3: M2M

```
$S$: parent particle cluster
$S'$: child particle cluster
Function M2M($S, S'$)
    $\gamma_{S,\boldsymbol{\nu}} = \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{S',\boldsymbol{\nu}'})\gamma_{S',\boldsymbol{\nu}'}$
    $\boldsymbol{p}_{S,\boldsymbol{\nu}} = \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{S',\boldsymbol{\nu}'})\gamma_{S',\boldsymbol{\nu}'}$
end
```

Algorithm 4.4: Upward Pass

```
$S$: particle cluster
Function upwardpass($S$)
    if children($S$) == $\emptyset$ then
        P2M($S$) (Algorithm 4.2)
    else
        for $s \in$ children($S$) do
            upwardpass($s$)
        end
        for $s \in$ children($S$) do
            M2M($S, s$) (Algorithm 4.3)
        end
    end
end
```

Algorithm 4.5: Stretched Diameter of Cluster

```
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
Function diam($S, \boldsymbol{s}$)
    $[a_x, b_x] \times [a_y, b_y] \times [a_z, b_z] = \mathrm{bbox}(S)$
    return $\|\boldsymbol{s} \circ (\boldsymbol{a} - \boldsymbol{b})/2\|_2$
end
```

Algorithm 4.6: Stretched Distance between two Clusters

```
$S_1$: particle cluster 1
$S_2$: particle cluster 2
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
Function dist($S_1, S_2, \boldsymbol{s}$)
    $[a_{1,x}, b_{1,x}] \times [a_{1,y}, b_{1,y}] \times [a_{1,z}, b_{1,z}] = \mathrm{bbox}(S_1)$
    $[a_{2,x}, b_{2,x}] \times [a_{2,y}, b_{2,y}] \times [a_{2,z}, b_{2,z}] = \mathrm{bbox}(S_2)$
    $\boldsymbol{c}_1 = (\boldsymbol{a}_1 + \boldsymbol{b}_1)/2$
    $\boldsymbol{c}_2 = (\boldsymbol{a}_2 + \boldsymbol{b}_2)/2$
    return $\|\boldsymbol{s} \circ (\boldsymbol{c}_1 - \boldsymbol{c}_2)\|_2$
end
```

Algorithm 4.7: Stretched Admissibility Condition for Cluster-Cluster Interaction

```
$S_1$: particle cluster 1
$S_2$: particle cluster 2
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
Function admissible($S_1$, $S_2, \boldsymbol{s}, \eta$)
    $r_1 = \mathrm{diam}(S_1, \boldsymbol{s})$ (Algorithm 4.5)
    $r_2 = \mathrm{diam}(S_2, \boldsymbol{s})$ (Algorithm 4.5)
    $d = \mathrm{dist}(S_1, S_2, \boldsymbol{s})$ (Algorithm 4.6)
    return $\max(r_1, r_2)/d < \eta$
end
```

Algorithm 4.8: M2L

```
$S_t$: target particle cluster
$S_s$: source particle cluster
Function M2L($S_t, S_s$)
    $\boldsymbol{E}_{S_t,\boldsymbol{\mu}} = \sum_{\boldsymbol{\nu}} \gamma_{S_s,\boldsymbol{\nu}} \cdot \boldsymbol{g}(\xi_{S_t,\boldsymbol{\mu}}, \xi_{S_s,\boldsymbol{\nu}})$
    $\boldsymbol{B}_{S_t,\boldsymbol{\mu}} = \sum_{\boldsymbol{\nu}} \boldsymbol{p}_{S_s,\boldsymbol{\nu}} \times \boldsymbol{g}(\xi_{S_t,\boldsymbol{\mu}}, \xi_{S_s,\boldsymbol{\nu}})$
end
```

Algorithm 4.9: P2P

```
$S_t$: target particle cluster
$S_s$: source particle cluster
Function P2P($S_t, S_s$)
    for $i \in \hat{S}_t$ do
        for $j \in \hat{S}_s$ do
            $\boldsymbol{E}_i = \boldsymbol{E}_i + \gamma_j \boldsymbol{g}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{p}_j)$
            $\boldsymbol{B}_i = \boldsymbol{B}_i + \boldsymbol{p}_j \times \boldsymbol{g}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{p}_j)$
        end
    end
end
```

Algorithm 4.10: Cluster-Cluster Interaction by Dual Tree Traversal

```
$S_t$: target particle cluster
$S_s$: source particle cluster
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
Function dualtraverseinteract($S_t, S_s, \boldsymbol{s}, \eta$)
    if children($S_t$) $== \emptyset \wedge$ children($S_s$) $== \emptyset$ then
        P2P($S_t, S_s$) (Algorithm 4.9)
    else
        isAdmissible = admissible($S_t$, $S_s, \boldsymbol{s}, \eta$) (Algorithm 4.7)
        if isAdmissible then
            M2L($S_t$, $S_s$) (Algorithm 4.8)
        else if children($S_t$) $== \emptyset$ then
            for $s \in$ children($S_s$) do
                dualtraverseinteract($S_t, s, \boldsymbol{s}, \eta$)
            end
        else if children($S_s$) $== \emptyset$ then
            for $t \in$ children($S_t$) do
                dualtraverseinteract($t, S_s, \boldsymbol{s}, \eta$)
            end
        else
            if diam($S_t, \boldsymbol{s}$) $>$ diam($S_s, \boldsymbol{s}$) then
                for $t \in$ children($S_t$) do
                    dualtraverseinteract($t, S_s, \boldsymbol{s}, \eta$)
                end
            else
                for $s \in$ children($S_s$) do
                    dualtraverseinteract($S_t, s, \boldsymbol{s}, \eta$)
                end
            end
        end
    end
end
```

Algorithm 4.11: FMM with Stretch

```
$S$: particle cluster
$\boldsymbol{s}$: stretch factor
$N_0$: maximum number of particles in the leaf node
$\eta$: admissibility parameter
Function FMM($S$, $\boldsymbol{s}$, $N_0$, $\eta$)
    subdivide($S$, $\boldsymbol{s}$, $N_0$) (Algorithm 4.1)
    $\boldsymbol{E}_i = \mathbf{0}, \boldsymbol{B}_i = \mathbf{0} \quad \forall i \in \widehat{S}$
    upwardpass($S$) (Algorithm 4.4)
    dualtraverseinteract($S, S, \boldsymbol{s}, \eta$) (Algorithm 4.10)
    downwardpass($S$) (Algorithm 4.14)
end
```

Algorithm 4.12: L2L

```
$S$: parent particle cluster
$S'$: child particle cluster
Function L2L($S', S$)
    $\boldsymbol{E}_{S',\boldsymbol{\mu}'} = \boldsymbol{E}_{S',\boldsymbol{\mu}'} + \sum_{\boldsymbol{\mu}} \ell_{S,\boldsymbol{\mu}}(\xi_{S',\boldsymbol{\mu}'})\boldsymbol{E}_{S,\boldsymbol{\mu}}$
    $\boldsymbol{B}_{S',\boldsymbol{\mu}'} = \boldsymbol{B}_{S',\boldsymbol{\mu}'} + \sum_{\boldsymbol{\mu}} \ell_{S,\boldsymbol{\mu}}(\xi_{S',\boldsymbol{\mu}'})\boldsymbol{B}_{S,\boldsymbol{\mu}}$
end
```

Algorithm 4.13: L2P

```
$S$: particle cluster
Function L2P($S$)
    for $i \in \hat{S}$ do
        $\boldsymbol{E}_i = \boldsymbol{E}_i + \sum_{\boldsymbol{\mu}} \ell_{S,\boldsymbol{\mu}}(\boldsymbol{x}_i)\boldsymbol{E}_{S,\boldsymbol{\mu}}$
        $\boldsymbol{B}_i = \boldsymbol{B}_i + \sum_{\boldsymbol{\mu}} \ell_{S,\boldsymbol{\mu}}(\boldsymbol{x}_i)\boldsymbol{B}_{S,\boldsymbol{\mu}}$
    end
end
```

Algorithm 4.14: Downward Pass

```
$S$: particle cluster
Function downwardpass($S$)
    if children($S$) $==\emptyset$ then
        L2P($S$) (Algorithm 4.13)
    else
        for $s \in$ children($S$) do
            L2L($s, S$) (Algorithm 4.12)
        end
        for $s \in$ children($S$) do
            downwardpass($s$)
        end
    end
end
```

$(0, 0, p_0)$ with $p_0 = (\gamma^2 - 1)^{1/2}$ and $\gamma = 50$. The measured error is the maximal relative error in the electrical and magnetic field,

$$\text{error} := \max_{\boldsymbol{f} \in \{\boldsymbol{E}, \boldsymbol{B}\}} \left(\sum_{i=1}^{N} \|\boldsymbol{f}_i^t - \boldsymbol{f}_i^b\|_2^2 / \sum_{i=i}^{N} \|\boldsymbol{f}_i^b\|_2^2 \right)^{1/2},$$

where N is the number of particles in the system. The space-charge fields $\boldsymbol{f}_i^t$ and $\boldsymbol{f}_i^b$ experienced by the i-th particle are computed by FMM and the brute-force method, respectively. We can observe that a smaller admissibility parameter η leads to higher accuracy (smaller error) but costs more elapsed time. This is because fewer M2Ls in the coarse level are considered and each non-admissible cluster in the coarse level can result in more M2Ls in the finer level or P2Ps in the leaf level. Besides, the usage of higher interpolation degree n leads to a result with higher accuracy and higher elapsed time because more macro particles are used in the calculation of M2L.

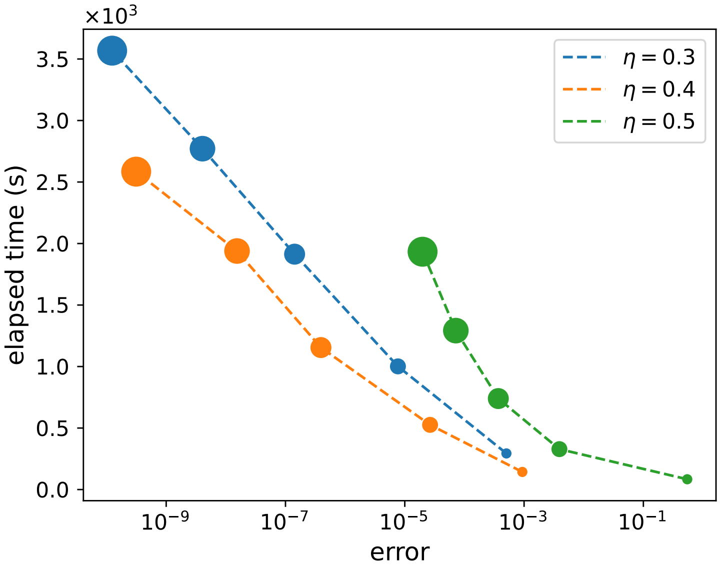
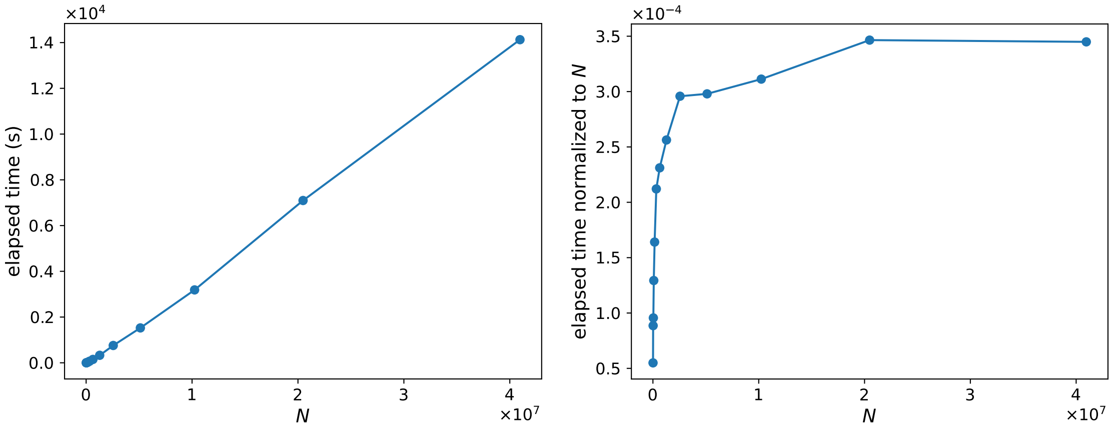


Figure 4.1: A plot of elapsed time against the error for the proposed FMM. Each line represents the result computed with an admissibility parameter $\eta = 0.3, 0.4, 0.5$. Each point in a line represents a simulation with an interpolation degree $n = 2, 4, 6, 8, 10$ and the maximum number of particles in the leaf cluster $N_0 = (n+1)^3$. The marker size is associated with the value of n; data with bigger n is expressed with a bigger point size.

In Figure 4.2, the elapsed time of FMM against the number of particles N is presented. We can see that our FMM approaches the theoretical complexity $\mathcal{O}(N)$ (Figure 4.2b).

We also perform code profiling on our solver and demonstrate the cumulative elapsed time of the six FMM kernels in Figure 4.3. One can observe that the total elapsed time is mostly dominated by P2P and M2L; this indicates that the routines of these two kernels will be the focus when any further optimizations for the solver are considered. Besides, one can also observe a sudden jump in the value of the elapsed time for P2P and M2L at a specific number of particles N. To understand this phenomenon, we consider a case where the total particle number is equal to a transition value $N = N_t^\kappa := 2^\kappa N_0$ with κ the depth of the cluster tree. If each leaf cluster is going to interact with at most a constant number of clusters via P2P, the



(a) elapsed time

(b) normalized elapsed time

Figure 4.2: The performance of the FMM method. Figure 4.2a shows the elapsed time used by FMM to evaluate the space-charge field of increasing numbers of particles N with $n = 4$, $N_0 = (n+1)^3$ and $\eta = 0.5$. Figure 4.2b shows the elapsed time normalized by N.

total number of operation counts to perform P2P can be written as

$$W_{\mathrm{P2P}}(N) = \mathrm{const} \cdot 2^{\kappa} \cdot N_0^2 \quad \text{for } N = N_t^{\kappa}. \tag{4.5.1}$$

When the number of particles N slightly increases with a value $\delta N \to 0$ so that $N > N_t$, the number of particles in each leaf cluster N_{f} will slightly increase with $\delta N_{\mathrm{f}} \to 0$ so that $N_{\mathrm{f}} > N_0$. In this case, each leaf cluster will be subdivided into two clusters and the cluster tree will gain one more level $\kappa + 1$. Therefore, the number of leaf clusters will increase from 2^{κ} to $2^{\kappa+1}$ and the value of N_{f} reduces from N_0 to $N_0/2$. Thus, the total number of operation counts for P2P can be written as

$$W_{\mathrm{P2P}}(N) = \mathrm{const} \cdot 2^{\kappa+1} \cdot (\tfrac{N_0}{2} + \delta N_{\mathrm{f}})^2 \quad \text{for } N_t^{\kappa+1} \geq N > N_t^{\kappa}, \tag{4.5.2}$$

with $N_0/2 \geq \delta N_{\mathrm{f}}(N, N_0) > 0$. The ratio between (4.5.1) and (4.5.2) for different δN_{f} is

$$\frac{W_{\mathrm{P2P}}(N_t^{\kappa} + \delta N)}{W_{\mathrm{P2P}}(N_t^{\kappa})} = \begin{cases} 1 & \text{for } \delta N_{\mathrm{f}} = 0, \\ \frac{1}{2} & \text{for } \delta N_{\mathrm{f}} \to 0, \\ 2 & \text{for } \delta N_{\mathrm{f}} = \frac{N_0}{2}. \end{cases}$$

Together with (4.5.2), we can see that W_{P2P} suddenly decreases to one half of $W_{\mathrm{P2P}}(N_t^{\kappa})$ as N increases from $N = N_t^{\kappa}$ and then grows quadratically until it is two times bigger than $W_{\mathrm{P2P}}(N_t^{\kappa})$ at $N = N_t^{\kappa+1}$. This performance model can qualitatively describe the trend of elapsed time for P2P. Likewise, we can also apply a similar analysis to M2L and write down the corresponding performance model as

$$W_{\mathrm{M2L}}(N) = \begin{cases} \mathrm{const} \cdot 2^{\kappa+1} \cdot (n+1)^6 & \text{for } N = N_t^{\kappa+1}, \\ \mathrm{const} \cdot 2^{\kappa+2} \cdot (n+1)^6 & \text{for } N_t^{\kappa+1} \geq N > N_t^{\kappa}. \end{cases} \tag{4.5.3}$$

Here, we use the fact that a balanced cluster tree with the depth l contains 2^{l+1} total clusters and the assumption that each cluster interacts with at most a constant value of clusters through M2L. Equation 4.5.3 shows that W_{M2L} suddenly increases to two times of $W_{\mathrm{M2L}}(N_t^{\kappa})$ as N slightly increases with $\delta N \to 0$ from $N = N_t^{\kappa}$; and then it remains constant whenever $N_t^{\kappa+1} \geq N > N_t^{\kappa}$. This performance model can successfully explain the behavior of the elapsed time for M2L.

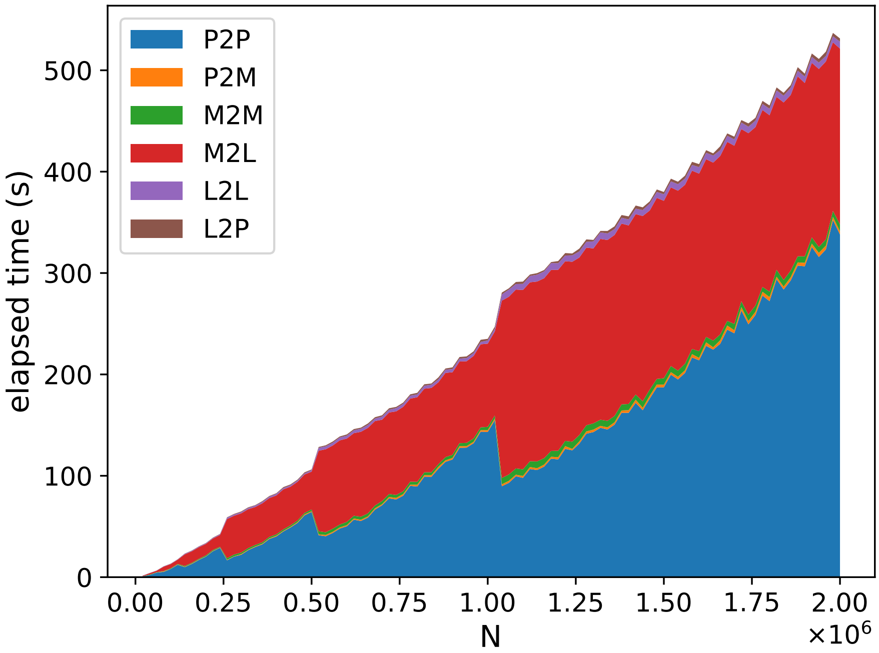
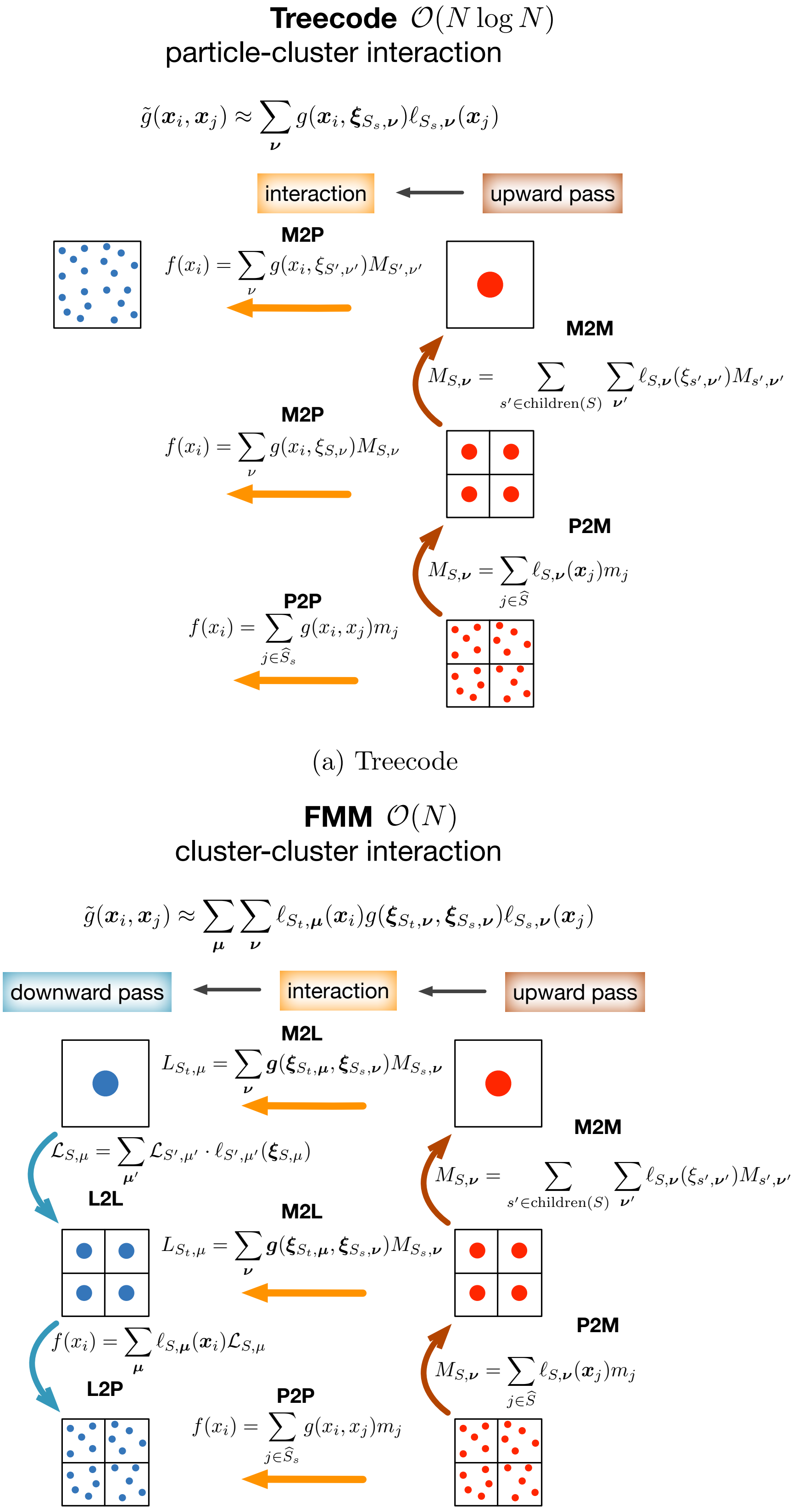


Figure 4.3: Elapsed time of the six FMM kernels against the increasing number of particles N, with polynomial interpolation degree $n = 4$, the maximum number of particles in the leaf cluster $N_0 = (n+1)^3$ and admissibility parameter $\eta = 0.5$.

4.6 Summary

In this chapter, we formulate an interpolation-based FMM for the efficient computation of the relativistic space-charge field from a particle beam. In particular, we derive the stretched ADMC of the cluster-cluster interaction for the relativistic kernel. Our solver shows that the proposed FMM scales like $\mathcal{O}(N)$. Besides, our code profiling result provides an insight that P2P and M2L should be especially emphasized when further optimization or parallelization is considered for the solver.


Treecode $\mathcal{O}(N\log N)$
particle-cluster interaction
$\tilde{g}(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx \sum_{\boldsymbol{\nu}} g(\boldsymbol{x}_i, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}})\ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j)$
interaction
upward pass
M2P
$f(x_i) = \sum_{\nu} g(x_i, \xi_{S',\nu'})M_{S',\nu'}$
M2M
$M_{S,\boldsymbol{\nu}} = \sum_{s'\in\mathrm{children}(S)} \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{s',\boldsymbol{\nu}'})M_{s',\boldsymbol{\nu}'}$
M2P
$f(x_i) = \sum_{\nu} g(x_i, \xi_{S,\nu})M_{S,\nu}$
P2M
$M_{S,\boldsymbol{\nu}} = \sum_{j\in\widehat{S}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)m_j$
P2P
$f(x_i) = \sum_{j\in\widehat{S}_s} g(x_i, x_j)m_j$

(a) Treecode

FMM $\mathcal{O}(N)$
cluster-cluster interaction
$\tilde{g}(\boldsymbol{x}_i, \boldsymbol{x}_j) \approx \sum_{\boldsymbol{\mu}} \sum_{\boldsymbol{\nu}} \ell_{S_t,\boldsymbol{\mu}}(\boldsymbol{x}_i)g(\boldsymbol{\xi}_{S_t,\boldsymbol{\nu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}})\ell_{S_s,\boldsymbol{\nu}}(\boldsymbol{x}_j)$
downward pass
interaction
upward pass
M2L
$L_{S_t,\mu} = \sum_{\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}})M_{S_s,\boldsymbol{\nu}}$
M2M
$\mathcal{L}_{S,\mu} = \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S',\mu'} \cdot \ell_{S',\mu'}(\boldsymbol{\xi}_{S,\mu})$
$M_{S,\boldsymbol{\nu}} = \sum_{s'\in\mathrm{children}(S)} \sum_{\boldsymbol{\nu}'} \ell_{S,\boldsymbol{\nu}}(\xi_{s',\boldsymbol{\nu}'})M_{s',\boldsymbol{\nu}'}$
L2L
M2L
$L_{S_t,\mu} = \sum_{\boldsymbol{\nu}} \boldsymbol{g}(\boldsymbol{\xi}_{S_t,\boldsymbol{\mu}}, \boldsymbol{\xi}_{S_s,\boldsymbol{\nu}})M_{S_s,\boldsymbol{\nu}}$
P2M
$f(x_i) = \sum_{\boldsymbol{\mu}} \ell_{S,\boldsymbol{\mu}}(\boldsymbol{x}_i)\mathcal{L}_{S,\mu}$
$M_{S,\boldsymbol{\nu}} = \sum_{j\in\widehat{S}} \ell_{S,\boldsymbol{\nu}}(\boldsymbol{x}_j)m_j$
L2P
P2P
$f(x_i) = \sum_{j\in\widehat{S}_s} g(x_i, x_j)m_j$

(b) FMM

Figure 4.4: A schematic comparison of treecode and FMM. The blue and red dots indicate the target and source particles, respectively.

Chapter 5

GPU Parallelization of Dual-Tree-Traversal based FMM

In the previous chapter, we introduced an FMM based on the dual tree traversal for the efficient computation of the relativistic space-charge field. We also demonstrated that the proposed FMM successfully brings the complexity down to $\mathcal{O}(N)$. A typical beam dynamics simulation requires hundreds or thousands of time steps. The number of space-charge field evaluations is also proportional to the number of time steps (the exact number depends on the choice of the numerical method for integrating the particle's equation of motion (2.3.1)). It is important to speed up and parallelize the FMM solver so that the overall simulation time can be further reduced. In this chapter, we discuss a GPU parallelization for our proposed FMM.

5.1 Overview

As illustrated in Section 4.1, the treecode computes the force-field for each single target particle through the particle-cluster interaction. The dual-tree-traversal-based FMM computes the force-field for a batch of particles through the cluster-cluster interactions and the total computational complexity can be thus reduced. This improvement also changes the tree walk scheme from multiple independent single tree traversals to a single-time dual tree traversal. A treecode can be parallelized in the data-parallelism way that each thread processes an independent tree traversal for each target particle; thus, the parallelization of a treecode can benefit directly from the power of GPUs which is based on the execution of multiple simple tasks through multiple threads in the SIMD (single instruction, multiple data) model. On the other hand, FMM might underuse the power of GPU because the dual tree traversal could be only executed by one thread. Because a single GPU core has weaker computing power compared to a single CPU core, the execution of the dual tree traversal should be done on the CPU instead of GPU. Therefore, hybrid CPU-GPU approaches based on the creation of the interaction lists by CPU were investigated by some previous works [90, 88]. In this approach, the CPU first performs a dual tree traversal to generate interaction lists; and then, the GPU uses these interaction lists to evaluate the force-field. In this chapter, we refer to the work proposed by Wilson *et al.* [88, 89] and discuss the GPU parallelization for our proposed FMM. Different to their approach which is based on OpenACC (a directive-based programming model) [91], our parallelization is based on the CUDA programming model. The execution of our GPU-parallelized FMM can be summarized in 10 steps in Algorithm 10. The H2D and D2H denote the "host to device" and "device to host", respectively. As shown in Figure 4.3, the execution of FMM spends most of the time on the interaction phases (P2P and M2L). Although the parallelization of each

FMM kernel is implemented in our application, we will only focus on the implementations of P2P and M2L (*i.e.*, step 6 and step 8 in Algorithm 10) in later discussions.

Algorithm 5.1: An Outline for the Execution of the Proposed GPU-Parallelized FMM

```
1  CPU: generate particles information $\boldsymbol{x}_i$, $\boldsymbol{p}_i$ and allocate $\boldsymbol{E}_i$, $\boldsymbol{B}_i$
2  CPU: create cluster tree $S$ with $\boldsymbol{x}_i$, $\boldsymbol{p}_i$
3  H2D: copy $\boldsymbol{x}_i$, $\boldsymbol{p}_i$ and $S$ to device
4  GPU: allocate $\gamma_{S,\boldsymbol{\nu}}$, $\boldsymbol{p}_{S,\boldsymbol{\nu}}$, $\boldsymbol{E}_{S,\boldsymbol{\nu}}$, $\boldsymbol{B}_{S,\boldsymbol{\nu}}$, $\boldsymbol{E}_i$, $\boldsymbol{B}_i$
5  GPU: perform upward pass with $S$, $\boldsymbol{x}_i$, $\boldsymbol{p}_i$ to compute $\gamma_{S,\boldsymbol{\nu}}$, $\boldsymbol{p}_{S,\boldsymbol{\nu}}$
6  CPU: perform dual-tree-traversal on $S$ to build interaction lists (ITLs)
7  H2D: copy ITLs to device
8  GPU: perform P2P, M2L with ITLs to compute $\boldsymbol{E}_{S,\boldsymbol{\nu}}$, $\boldsymbol{B}_{S,\boldsymbol{\nu}}$
9  GPU: perform downward pass with $\boldsymbol{E}_{S,\boldsymbol{\nu}}$, $\boldsymbol{B}_{S,\boldsymbol{\nu}}$ to compute $\boldsymbol{E}_i$, $\boldsymbol{B}_i$
10 D2H: copy $\boldsymbol{E}_i$, $\boldsymbol{B}_i$ to host
```

5.2 Array-based Tree Data Structure

In the implementation, it might be straightforward to express the cluster tree with a pointer-based data structure; that is, each node object (particle cluster in our case) contains data fields and a pointer, and this pointer is used to allocate the objects of children nodes. One major disadvantage of using a pointer-based tree is that the node objects are not stored at contiguous locations in the memory, and this makes the data transfer between host and device difficult. Hence, it might be beneficial to consider an array-based tree in the GPU application. Following the approach in Ref. [92], we use multiple arrays to store node objects with multiple members, one array for one member. A member of a node object with index i is located in the i-th element of the corresponding array. Besides, two additional arrays are respectively used to specify the parent index and children indices of nodes. Because our cluster tree is constructed through a k-d tree with cardinality-balanced subdivision of particles, it will be a balanced binary tree. Hence, we will narrow our following discussions to balanced binary tree.

Although there may exist several possibilities, we adapt the breadth-first scheme to assign the node index of a tree. With this index assignment scheme, the nodes in the level l have the indices $\{2^l, \ldots, 2^{(l+1)-1}\}$; and similarly, a node with the index i belongs to a level $\lfloor \log i / \log 2 \rfloor$. Here, we define that the level of the cluster tree starts from 0 and the node index starts from 1. The breadth-first scheme can ensure that the member data of nodes from the same level stays contiguously in an array. This data arrangement is cache-friendly for both the upward pass (P2M and M2M) and the downward pass (L2L and L2P) where the whole member data of nodes from the same level will be accessed for the calculation. Therefore, the parent index and the children pair of indices for a node with index i are defined as

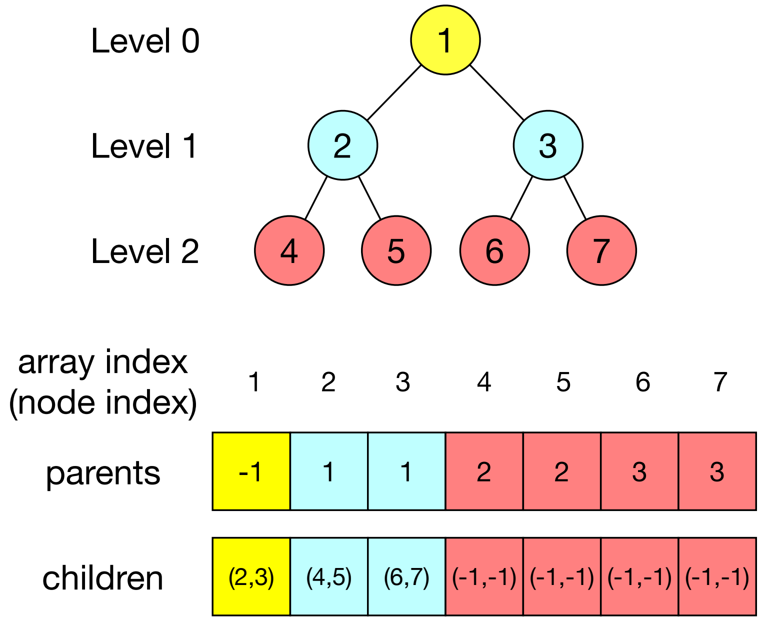
$$\mathrm{iparent}(i) = \begin{cases} -1 & i = 1, \\ \lfloor \frac{i}{2} \rfloor & i \neq 1, \end{cases} \quad \text{and} \quad \mathrm{ichildren}(i) = \begin{cases} (-1, -1) & \text{if leaf node}, \\ (2 \cdot i, 2 \cdot i + 1) & \text{else}. \end{cases}$$

A schematic representation of our array-based tree is provided in Figure 5.1. Because our tree is balanced (due to the cardinality-balanced subdivision scheme), we can preallocate a fixed-size array with the knowledge that the total number of nodes is

$2^{(\kappa+1)} - 1$ with the tree-depth

$$\kappa = \begin{cases} 0 & N \leq N_0, \\ \lceil \log(N/N_0)/\log 2 \rceil & N > N_0. \end{cases}$$

It is worth noting that the resulting tree might not be balanced if other space-subdivision schemes are adapted. In this case, one may preallocate a big enough array that each node contains the maximum possible number of children nodes. However, this causes large memory of unused nodes and leads to poor load-balancing across multiple ranks when MPI parallelization is considered [93]. One possible way to work around this issue is first creating a pointer-based tree, and an array-based tree can be allocated based on the information from that. This approach is adapted by some solvers, *e.g.* BaryTree [94].



array index (node index)	1	2	3	4	5	6	7
parents	-1	1	1	2	2	3	3
children	(2,3)	(4,5)	(6,7)	(-1,-1)	(-1,-1)	(-1,-1)	(-1,-1)

Figure 5.1: An array-representation of a balanced binary tree. The node index is assigned with a breadth-first scheme.

5.3 Parallelization of P2P and M2L Kernels

Two lists of interaction pairs respectively for P2P and M2L, called interaction lists (ITLs), are generated by the execution of the dual tree traversal (Algorithm 5.2) in the CPU. After that, the ITLs are copied into the GPU and the GPU kernels respectively of P2P and M2L are launched and each interaction pair is handled by a thread block. As P2P and M2L are both similar to a direct summation algorithm, their GPU implementations are straightforward; one thread in the threads block handles the evaluation of the force-field of one target micro/macro particle (P2P/M2L). The GPU-parallelization of P2P is illustrated in Figure 5.3. In our implementation, we use an additional array to store the indices of all particles in the system and the indices of particles from a cluster will always stay in a contiguous memory block in the array during the subdivision (cf. Section B.3). However, each member data of particles (*e.g.,* positions and momenta) from a cluster accessed through this particle-indices array does not necessarily stay contiguous in its array

(Figure 5.3a). Thus, a member data of source particles accessed by threads is non-contiguously distributed in an array. This can slow down the application because the data access is not cache-friendly and requires frequent access from the global memory. One way to remedy this problem is using the shared memory (Figure 5.2) provided in CUDA-capable GPUs: we first load each member data of particles from a source cluster to shared memory so that the data can be accessed much faster by threads (Figure 5.3b). For one thing, each member data of source particles stays in a contiguous block in the shared memory. For another, the shared memory is on-chip memory and has much lower latency than the global memory. For the M2L implementation, it is not necessary to apply the shared memory because the member data of macro particles from a cluster is originally in a contiguous memory block. The data access is already cache-friendly so that the L1 cache in each streaming multiprocessor can be effectively used. A schematic of M2L implementation is provided in Figure 5.4. Because the implementations of the other FMM kernels share large similarities with P2P or M2L, we will not go through the detail of the implementations.

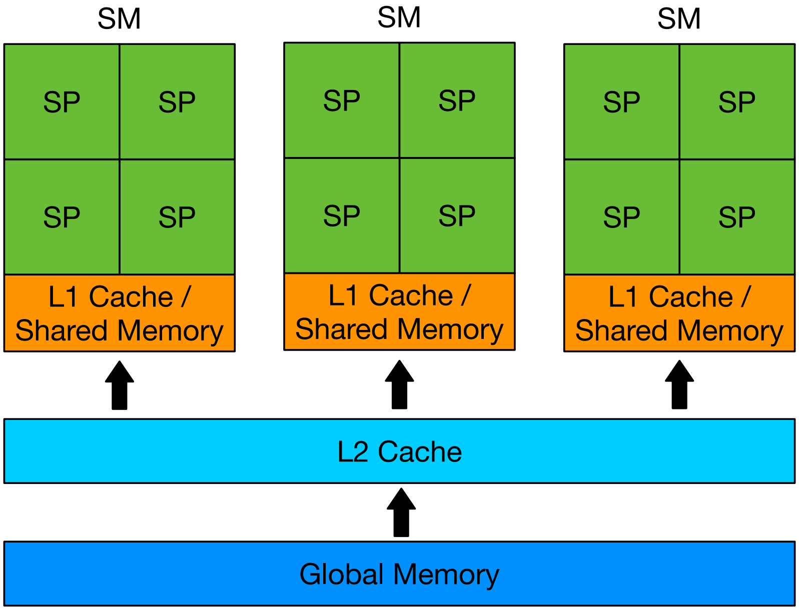
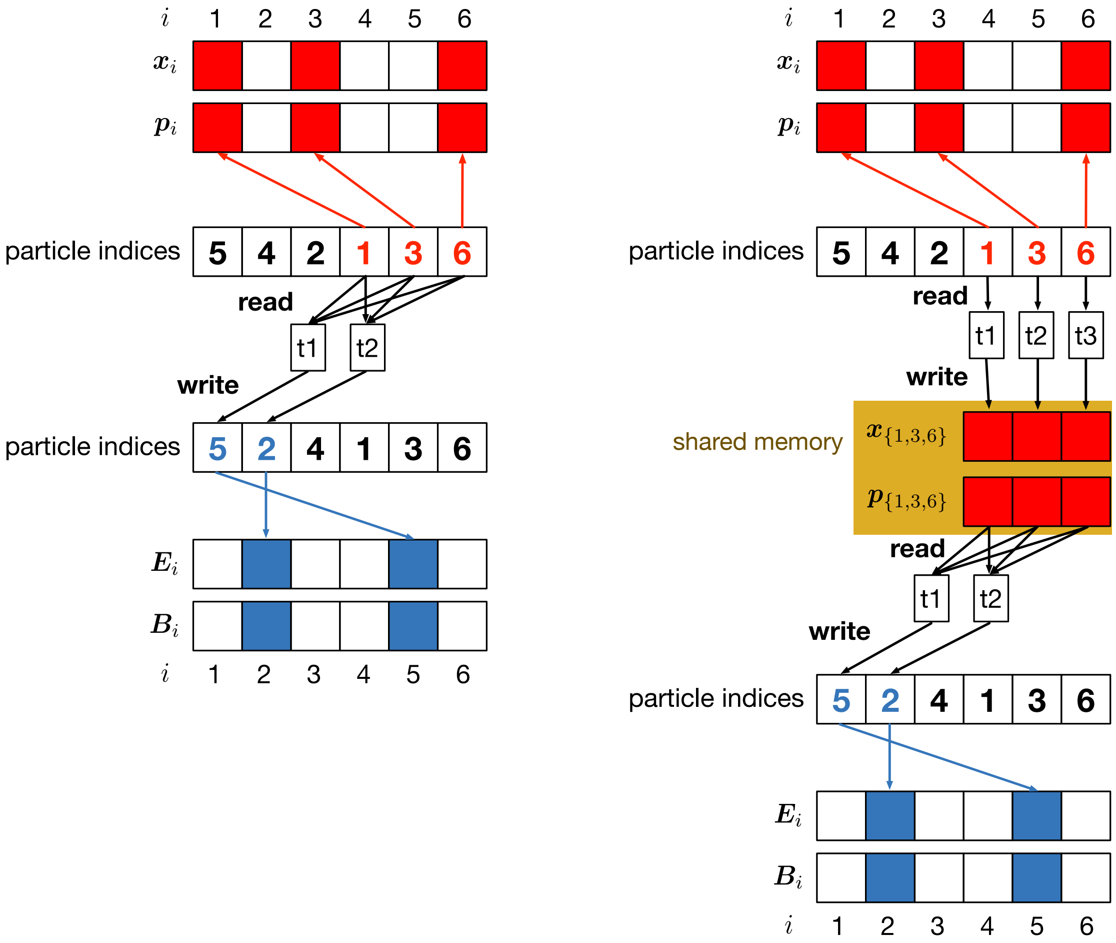


Figure 5.2: Memory hierarchy of CUDA-capable GPUs. SP and SM denote streaming Processor and streaming multiprocessor, respectively.



(a) P2P without shared memory

(b) P2P with shared memory

Figure 5.3: Implementations of P2P kernels (a) without shared memory and (b) with shared memory. The data associated with the target and source particle is colored with blue and red, respectively. The thread is denoted by a shorthand "t".

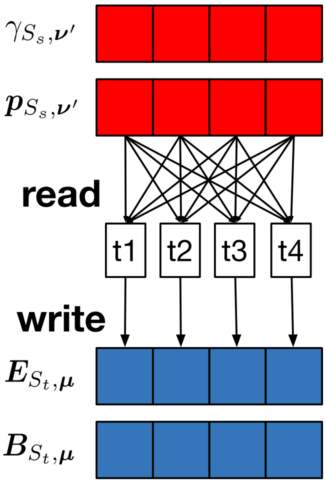

$\gamma_{S_s,\boldsymbol{\nu}'}$
$\boldsymbol{p}_{S_s,\boldsymbol{\nu}'}$
read
t1
t2
t3
t4
write
$\boldsymbol{E}_{S_t,\boldsymbol{\mu}}$
$\boldsymbol{B}_{S_t,\boldsymbol{\mu}}$

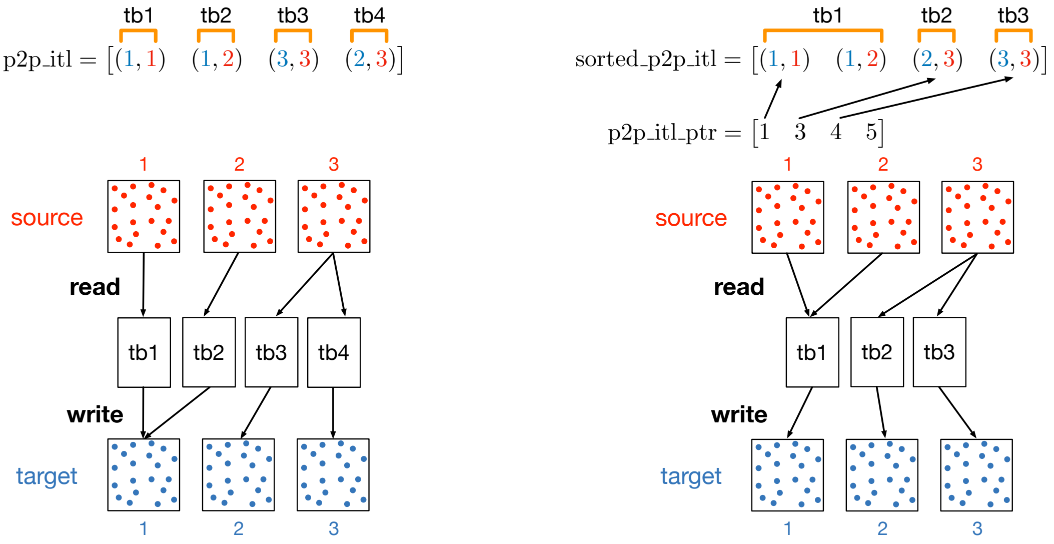
Figure 5.4: An illustration of the implementation of the M2L kernel. The data associated with the macro particle of the target cluster and the source cluster is colored with blue and red, respectively. The thread is denoted by a shorthand "t".

Algorithm 5.2: Generation of Interaction Lists by Dual Tree Traversal

```
$S_i$: particle cluster with node index $i$
$\boldsymbol{s}$: stretch factor
$\eta$: admissibility parameter
p2p_itl: initially empty list of interaction for P2P (global scope)
m2l_itl: initially empty list of interaction for M2L (global scope)
Function dualtraversefillitl($i, j, \boldsymbol{s}, \eta$)
    if ichildren($i$) == $(-1, -1) \wedge$ ichildren($j$) == $(-1, -1)$ then
        push $(i, j)$ to p2p_itl
    else
        isAdmissible = admissible($S_i$, $S_j$, $\boldsymbol{s}, \eta$) (Algorithm 4.7)
        if isAdmissible then
            push $(i, j)$ to m2l_itl
        else if ichildren($i$) == $(-1, -1)$ then
            for $k \in$ ichildren($j$) do
                dualtraversefillitl($i, k, \boldsymbol{s}, \eta$)
            end
        else if ichildren($j$) == $(-1, -1)$ then
            for $k \in$ ichildren($i$) do
                dualtraversefillitl($k, j, \boldsymbol{s}, \eta$)
            end
        else
            if diam($S_i, \boldsymbol{s}$) > diam($S_j, \boldsymbol{s}$) then
                for $k \in$ ichildren($i$) do
                    dualtraversefillitl($k, j, \boldsymbol{s}, \eta$)
                end
            else
                for $k \in$ ichildren($j$) do
                    dualtraversefillitl($i, k, \boldsymbol{s}, \eta$)
                end
            end
        end
    end
end
```

5.4 Race Conditions in P2P and M2L Kernels

In CUDA applications, a GPU kernel can be launched with a grid of thread blocks. Several thread blocks can be executed by a streaming multiprocessor concurrently, depending on the resource required to run a thread block. In the execution of M2L or P2P, each pair of interaction is handled by one thread block and it is possible that several pairs of interaction with the same target index are handled by different thread blocks simultaneously. This can cause a race condition and produce an unexpected result because the corresponding memory data associated with a target cluster can be updated by the threads from the different blocks at the same time (Figure 5.5a). One common way to remedy the race condition is using CUDA's atomic operations [95] which locks a memory location so that only one exclusive thread is allowed to write the value stored in this memory location each time. However, the atomic operations in CUDA are only supported for some primitive types (*e.g.*, `Int32` and `Float32`) and cannot be used in our implementation because each three-dimensional vector in the physical system (*e.g.*, position, momentum and vector field) is represented by a non-primitive and immutable type `SVector{3,T}` [96] with three elements of type `T`. Due to this immutability, we cannot apply atomic operations to change any individual elements of a `SVector{3,T}` object even though `T` is a primitive type (if so we can apply atomic operations to update each element of a `SVector{3,T}` object). Therefore, in our implementation, we divide pairs of interaction into groups such that each group only contains the pairs of interaction with the same target index; and during the kernel execution, each group will be handled by a thread block. To implement this, we first sort the pairs by the value of the target index which can be done efficiently with Quicksort. After that, we generate an additional array to indicate the start position of each group of pairs in the sorted ITL so that this array can be used for the dispatch of thread blocks to each group during the kernel execution (Figure 5.5b).



(a) original problem

(b) a solution

Figure 5.5: A schematic of the race condition problem during the execution of P2P kernel (same for M2L kernel) with ITL. Figure 5.5a illustrates the original problem. Figure 5.5b illustrates a solution by dividing the pairs of interaction to groups with the same target index. The thread block is denoted by a shorthand "tb".

5.5 Results

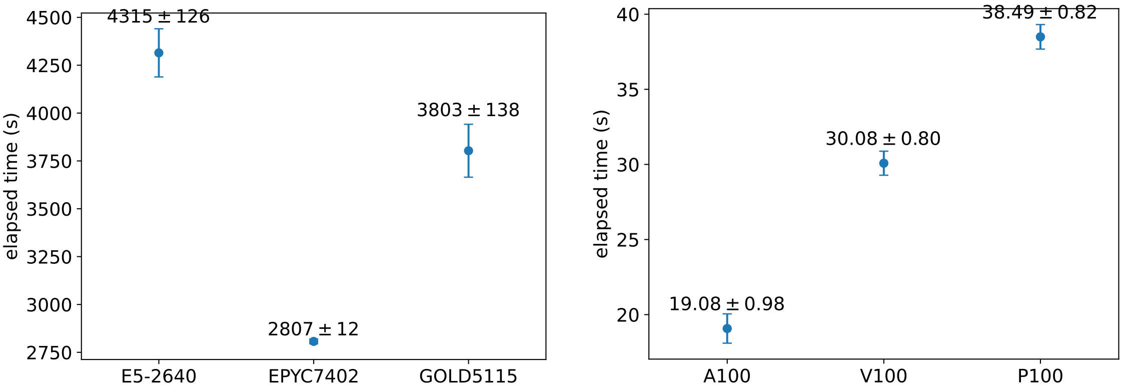
In this study, a package FMM4RBGPU.jl is written in the Julia programming language [46] with CUDA.jl [97, 98]. This package provides CPU (serial) and GPU solvers of the dual-tree-traversal-based FMM for the efficient computation of the relativistic space-charge field. The cluster tree is implemented using the array-based data structure discussed in Section 5.2.

To understand the performance of our GPU parallelization, we run a simulation with $N = 2.56 \times 10^7$ particles on different CPUs and GPUs listed in Table 5.1. The corresponding elapsed times are demonstrated in Figure 5.6. We can see that our

GPU-based solver can achieve a speed up between 57 and 197 (relative to the result of a single CPU).

CPU	GPU
INTEL XEON E5-2640V4	NVIDIA A100
AMD EPYC 7402	NVIDIA V100
INTEL XEON GOLD 5115	NVIDIA P100

Table 5.1: CPUs and GPUs used in the simulations for the performance benchmark.



(a) CPU

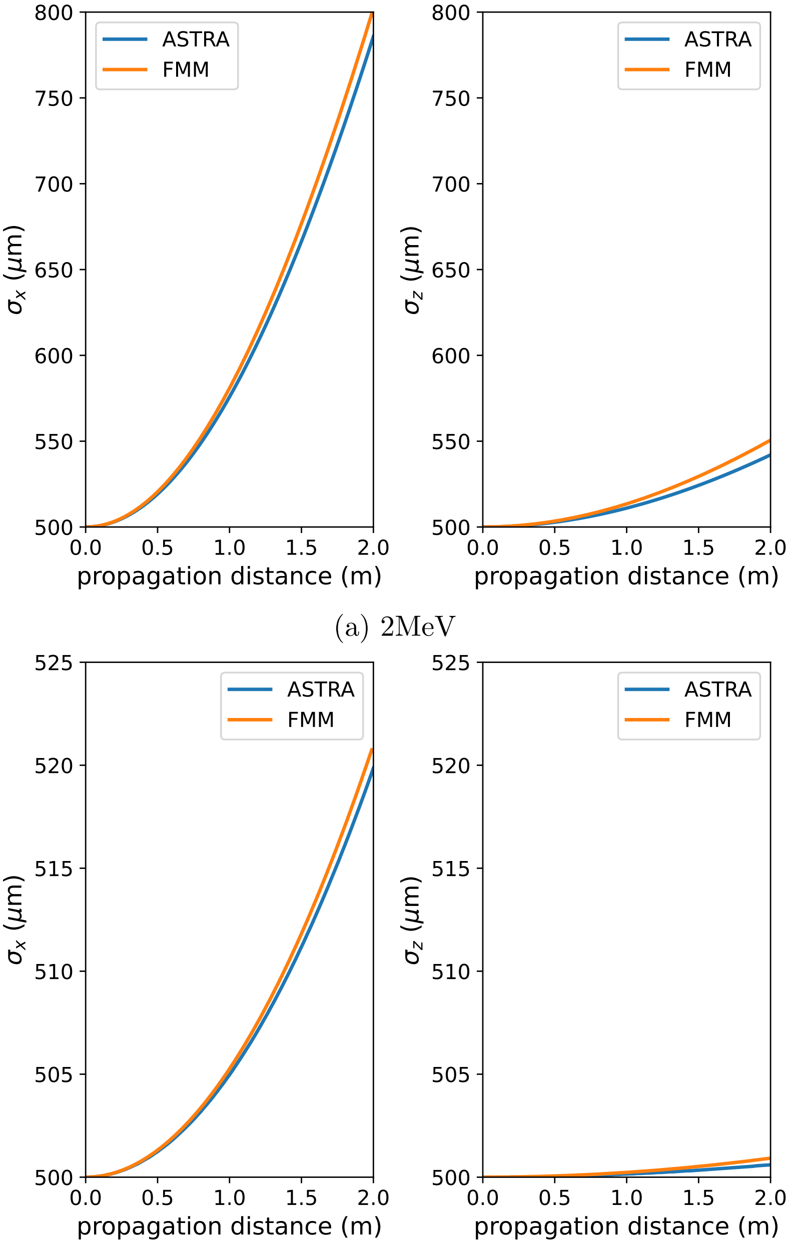
(b) GPU

Figure 5.6: Elapsed times of a simulation with $N = 2.56 \times 10^7$ particles on different (a) CPUs and (b) GPUs. The simulation is performed with $\eta = 0.5$, $n = 4$ and $N_0 = (n+1)^3$. Each data point shows the statistical result of 100 runs.

To validate our GPU-parallelized FMM solver, we simulate bunch broadening of an 1 pC electron beam propagating in the free space with our solver and ASTRA (a beam dynamics solver based on the particle-in-cell method) [34]. The simulation results from both solvers are demonstrated in Figure 5.7. The electron beam has an initial Gaussian distribution with the same RMS beam size 500 µm in each direction. We use 10^4 macro particles and 6.25×10^6 real particles for the simulations of ASTRA and FMM, respectively. The Boris method (2.3.3) is used to push the charged particles in the simulation. One may observe that the results computed from ASTRA and FMM are very close.

5.6 Summary

In this chapter, we discuss a GPU parallelization for our FMM solver. Instead of the commonly used atomic operations, the race conditions appearing in P2P and M2L are solved by dividing the pairs of interaction with the same target index into groups. The result shows that our solver can achieve a speedup of roughly 200 when executed on GPUs. Our solver is also validated by the simulation of bunch broadening and by comparing it with the result of ASTRA.



(a) 2MeV

(b) 5MeV

Figure 5.7: Simulation results of bunch broadening for (a) 2 MeV and (b) 5 MeV beams using ASTRA and FMM. The transverse beam size σ_x and the longitudinal beam size σ_z are plotted against the propagation distance. The FMM parameters are $\eta = 0.5$, $n = 3$ and $N_0 = (n + 1)^3$.

Chapter 6

ICS Simulation Model with Space-Charge Effect

In Chapter 3 and Chapter 4, we introduced numerical methods for the efficient computation of the relativistic space-charge field. Also, in Chapter 5, we discuss a GPU parallelization for our proposed FMM and develop a package for that. In this chapter, we will combine our research efforts made in this thesis and propose a GPU-parallelized ICS simulation solver with the consideration of the space-charge effect.

6.1 Simulation Model

We can now include the space-charge effect in the ICS simulation model, *i.e.*, we solve the equation of motion of a charged particle

$$\begin{aligned} \frac{d\boldsymbol{x}}{dt} &= \frac{1}{\gamma}\boldsymbol{p}, \\ \frac{d\boldsymbol{p}}{dt} &= \frac{q}{m}(\boldsymbol{E}(\boldsymbol{x},t) + \frac{\boldsymbol{p}}{\gamma} \times \boldsymbol{B}_(\boldsymbol{x},t)), \end{aligned}$$

with respect to the force-fields

$$\boldsymbol{E} = \boldsymbol{E}_{\text{ext}} + \boldsymbol{E}_{\text{spc}} \quad \text{and} \quad \boldsymbol{B} = \boldsymbol{B}_{\text{ext}} + \boldsymbol{B}_{\text{spc}}. \tag{6.1.1}$$

Therefore, in this simulation model, the following tasks are done in every simulation time step:

1. computing the radiation spectral-angular distribution,
2. evaluating the relativistic space-charge field,
3. pushing particles using Boris method.

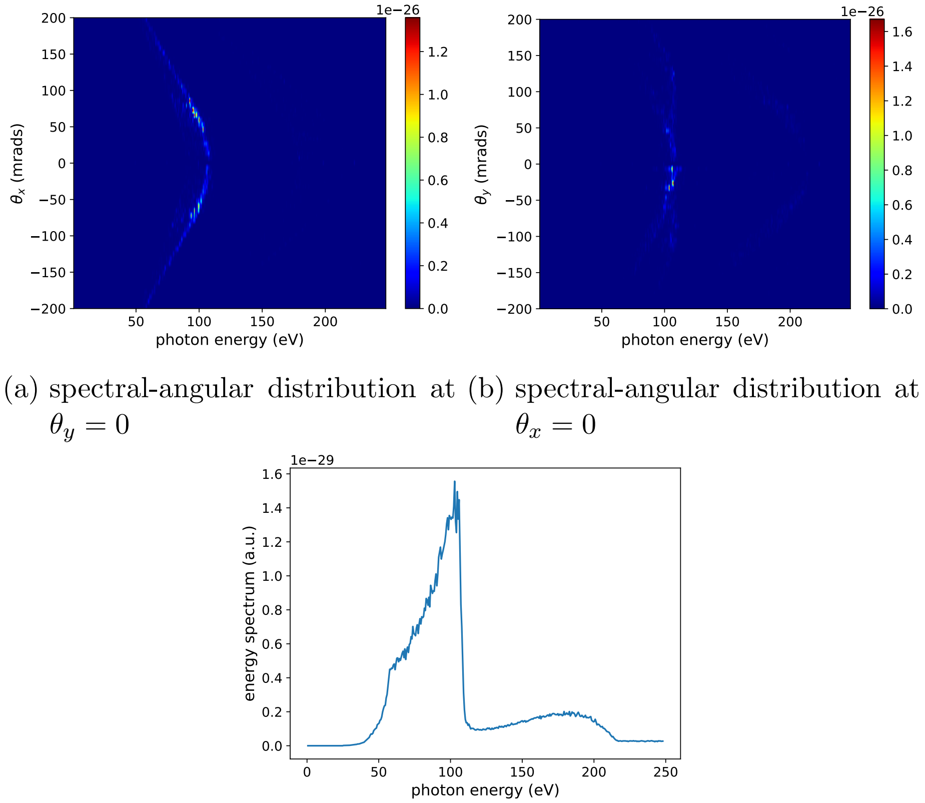
A simulation solver is implemented for the simulation model discussed above. In this solver, the radiation spectral-angular distribution is computed by the frequency-domain method instead of the time-domain method. The time-domain method requires large memory for the storage of the total radiation field over a uniform time grid and the memory requirement might not be manageable for a single GPU. The corresponding routine of the frequency-domain method is parallelized in the way that each GPU thread computes the radiation energy in a frequency and at an observation point (*i.e.*, the detector parallelization introduced in Section 2.5.2). The space-charge field calculation in each time step is performed by the GPU routines from the package FMM4RBGPU.jl proposed in Section 5.5. The Boris method (2.3.3) is used to integrate the particles' equation of motion and the corresponding routine is also parallelized such that each GPU thread updates the trajectory of a particle.

6.2 Results and Summary

To investigate the effect of the space-charge field on ICS, we consider a simulation with the parameters listed in Table 6.1. The simulations both with and without space-charged effect are performed and the radiation profiles for both cases are demonstrated in Figure 6.1 and Figure 6.2, respectively. We can see that the radiation profiles from both simulations have almost the same patterns and magnitudes. This validates our assumption in Chapter 2 that the space-charge effect has negligible impact on the conventional ICS light sources.

Parameter	Value
ICS Laser	
laser strength a_0	0.5
wavelength	1 µm
pulse duration	100 fs
Electron Beam	
beam charge	0.205 pC
mean energy	2.56 MeV
σ_x, σ_y, σ_z	10 µm, 10 µm, 10 µm
energy spread	0
beam divergence (x,y,z)	0, 0, 0

Table 6.1: Parameters for ICS simulations. The laser fields are sinusoidally oscillated in time with a finite duration and a constant amplitude. The particle beam has Gaussian distribution in space with RMS beam sizes σ_x, σ_y and σ_z. The total number of particles is 1.28×10^6 so each particle represents one electron.

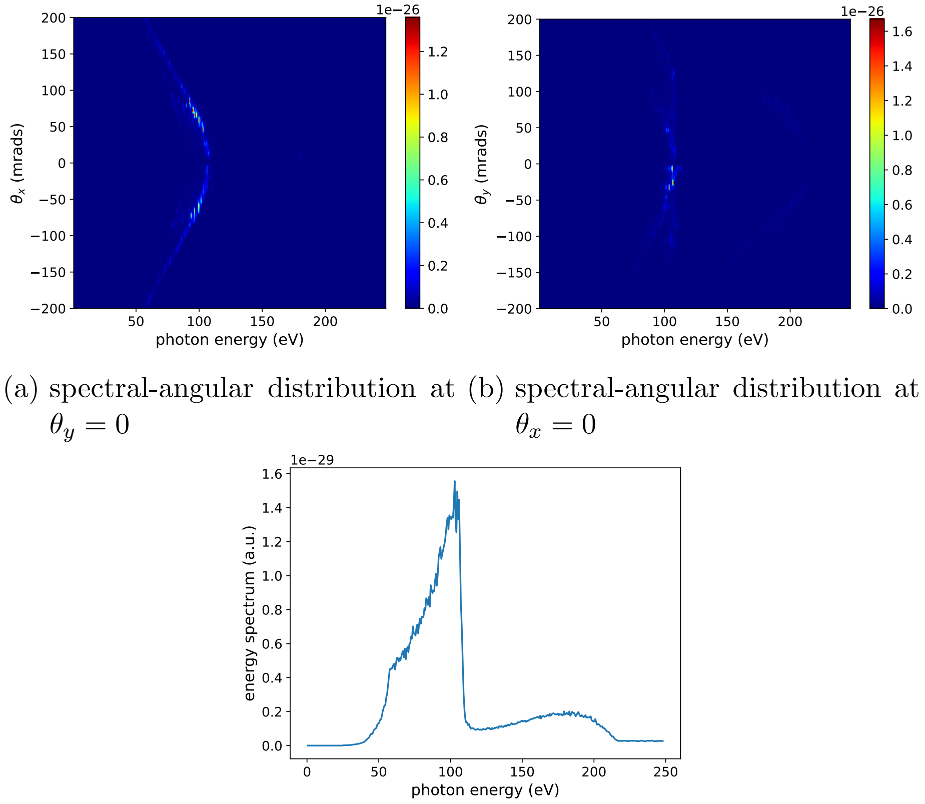


(a) spectral-angular distribution at $\theta_y = 0$

(b) spectral-angular distribution at $\theta_x = 0$

(c) energy spectrum

Figure 6.1: Simulated ICS radiation profiles without space-charge field.



(a) spectral-angular distribution at $\theta_y = 0$

(b) spectral-angular distribution at $\theta_x = 0$

(c) energy spectrum

Figure 6.2: Simulated ICS radiation profiles with the space-charge field.

Chapter 7

Conclusion and Outlook

This thesis covers the research for the numerical modeling of the ICS process. We first neglect the space-charge effect in the simulation model and investigate the Liénard–Wiechert potential for the computation of ICS radiation spectra. To apply the Liénard–Wiechert potential for the evaluation of radiation spectra, we discuss a time-domain method which preserves the temporal information of the radiation field and the radiation field in the time-domain can be used to compute spectra efficiently by the fast Fourier transform. Through our performance analysis of the computational costs of both the time-domain method and the frequency-domain method, we found that the time-domain method is in general more favorable than the frequency-domain method for the computation of the ICS radiation spectra within the specification of real-world experiment projects. Our developed solver has been used to simulate the radiation profile for the AXSIS project proposed in DESY [24, 99].

For the modeling of the space-charge effect in the ICS, we systematically investigate the interpolation-based fast summation methods. We formulate a treecode and an FMM for the fast evaluation of the relativistic space-charge field and propose two error-control schemes. Our numerical results show that the scheme based on the modified admissibility condition has better accuracy than the scheme based on the relativistic transformation. Besides, GPU parallelization is also included in our numerical solver and the GPU solver can achieve a speedup of roughly 200. Finally, we discuss an ICS simulation model with the consideration of the space-charge effect and a corresponding solver based on the computational techniques discussed in this thesis is developed.

Our simulation result shows that the space-charge effect on the ICS radiation spectrum is insignificant for the parameters which can appear in the conventional compact sources (*i.e*, an electron beam with sizes much larger than the radiation wavelength and laser pulse with a duration of 100 fs). In this case, the space-charge effect is negligible and the simulation can be performed much faster without taking the space-charge into account. However, due to the recent advances in strong-field nano-optics, the generations of attosecond electron pulse trains was experimentally demonstrated [100, 101, 102, 103]. This breakthrough opens up a feasibility of generating coherent ICS X-rays from pre-modulated electron pulse trains in the scale of subnanometers. It might be essential to understand if the structure of microbunches can be distorted by the space-charge effect during the ICS process. Thus, the developed simulation model/solver which takes the space-charge effect into account can be an indispensable tool for investigating this problem in the future. It is also worth noting that our developed methods/solvers for the space-charge field calculation are not limited to the simulation of ICS; they can also be used in other beam dynamics simulations where the consideration of point-to-point Coulomb effect is necessary and the mean-field model cannot be applied [50, 78, 65]. In Ref. [78], the space-charge field was modeled by the computation of Lorentz-boosted Coulomb field (*i.e.*, Equations (3.4.1)) via brute-force method and only 10^4 macro particles were used to approximate 1.6 pC and 16 pC beams due to the high computational complexity of $\mathcal{O}(N^2)$. For this problem, a real-particle simulation (*i.e.*, using 10^7 and 10^8 particles for the 1.6 pC and 16 pC beams, respectively) is manageable with our FMM solver. Because only single-GPU parallelization is implemented in our FMM solver, it might not be enough for real-particle simulations of high bunch charge beams, *e.g.*, a 1 nC beam requires 10^{10} real particles. Therefore, it is also worth it to consider multi-GPU parallelization for the solver in the future.

Appendix A

A.1 Normalization of Physical Quantities

In the theoretical and numerical study, it is beneficial to normalize the physical quantities. For the Inverse Compton Scattering Process, it is natural to use the laser's wave number k_0 and frequency ω_0 to define the characteristic length scale and time scale. Here, the symbols with tilde denote normalized quantities and symbols denote original physical quantities.

- ω_0 = characteristic frequency (laser in this case)
- k_0 = characteristic wave number (laser in this case)
- q_e = charge of a single electron
- m_e = rest mass of a single electron
- $\tilde{q} = q/q_e$
- $\tilde{t} = \omega_0 t$
- $\tilde{\boldsymbol{x}} = k_0 \boldsymbol{x}$
- $\tilde{\boldsymbol{v}} = \boldsymbol{v}/c_0$
- $\tilde{\boldsymbol{p}} = \boldsymbol{p}/m_e c_0$
- $\tilde{\boldsymbol{E}} = e\boldsymbol{E}/m_e \omega_0 c_0$
- $\tilde{\boldsymbol{B}} = e\boldsymbol{B}/m_e \omega_0$

A.2 Property of the Retardation Condition in the Liénard–Wiechert Potential

Lemma A.2.1. *When an observer at the position $\boldsymbol{x}$ and time t receives a radiation pulse from a particle, this pulse can only be emitted from a unique position $\boldsymbol{x}'(t_r)$ in this particle's trajectory at a unique past time t_r. That is only one root t_r fulfills the retardation condition*

$$t = t_r + \frac{\|\boldsymbol{x} - \boldsymbol{x}'(t_r)\|_2}{c_0}. \tag{A.2.1}$$

Proof. We can prove by contradiction. Assume that two positions at the past times t_1, t_2 with $t_2 > t_1$ in a particle's trajectory fulfill the retardation condition for the given observation position $\boldsymbol{x}$ and time t ($t > t_1, t_2$). We have

$$t = t_1 + \frac{\|\boldsymbol{x} - \boldsymbol{x}'(t_1)\|_2}{c_0} \quad \text{and} \quad t = t_2 + \frac{\|\boldsymbol{x} - \boldsymbol{x}'(t_2)\|_2}{c_0}$$

which can be recast into (after the subtraction of them)

$$t_2 - t_1 = \frac{\|\boldsymbol{x} - \boldsymbol{x}'(t_1)\|_2}{c_0} - \frac{\|\boldsymbol{x} - \boldsymbol{x}'(t_2)\|_2}{c_0}. \tag{A.2.2}$$

By triangular inequality, we have

$$\begin{aligned} &\|\boldsymbol{x} - \boldsymbol{x}'(t_1)\|_2 \leq \|\boldsymbol{x} - \boldsymbol{x}'(t_2)\|_2 + \|\boldsymbol{x}'(t_2) - \boldsymbol{x}'(t_1)\|_2 \\ \Longrightarrow &\|\boldsymbol{x} - \boldsymbol{x}'(t_1)\|_2 - \|\boldsymbol{x} - \boldsymbol{x}'(t_2)\|_2 \leq \|\boldsymbol{x}'(t_2) - \boldsymbol{x}'(t_1)\|_2. \end{aligned} \tag{A.2.3}$$

After substituting (A.2.2) into (A.2.3), we get

$$t_2 - t_1 \leq \frac{\|\boldsymbol{x}'(t_2) - \boldsymbol{x}'(t_1)\|_2}{c_0} \quad \Longrightarrow \quad c_0 \leq \frac{\|\boldsymbol{x}'(t_2) - \boldsymbol{x}'(t_1)\|_2}{t_2 - t_1},$$

where $\|\boldsymbol{x}'(t_2) - \boldsymbol{x}'(t_1)\|_2/(t_2 - t_1)$ is the particle's average speed in the time duration $t_2 - t_1$. This violates a postulate of special relativity because no objects can traverse with a speed faster than (or equal to) the speed of light. □

Appendix B

B.1 Approximation of the Denominator of the Relativistic Kernel Function

Lemma B.1.1. *Let $\boldsymbol{p} := p_\rho \boldsymbol{e}_r + p_z \boldsymbol{e}_z$ a vector in the cylindrical coordinate. If $p_z \gg p_r$, we have the approximation*

$$\|\boldsymbol{x}\|_2^2 + (\boldsymbol{p} \cdot \boldsymbol{x})^2 \approx r^2 + (1 + p_z^2)z^2,$$

where $r = x^2 + y^2$.

Proof. In the cylindrical coordinate, the function can be expressed as

$$\|\boldsymbol{x}\|_2^2 + (\boldsymbol{p} \cdot \boldsymbol{x})^2 = r^2 + z^2 + p_r^2 r^2 + p_z^2 z^2 + 2 p_r p_z r z. \tag{B.1.1}$$

To find out the approximation, we analyze the term

$$p_r^2 r^2 + p_z^2 z^2 + 2 p_r p_z r z \tag{B.1.2}$$

in three main different cases.

For the cases of $r \sim z$ and $r \ll z$, we first recast (B.1.2) to

$$p_z^2 z^2 \left(\frac{p_r^2}{p_z^2} \frac{r^2}{z^2} + 1 + 2 \frac{p_r p_z}{p_z^2} \frac{r^2}{z^2} \right)$$

and we can conclude that (B.1.2) can be approximated by $p_z^2 z^2$.

For the case $r \gg z$, we recast (B.1.2) to

$$p_r^2 r^2 \left(1 + \frac{p_z^2}{p_r^2} \frac{z^2}{r^2} + 2 \frac{p_z}{p_r} \frac{z}{r} \right)$$

and consider three further scenarios:

- if $\frac{p_z}{p_r} \frac{z}{r} \gg 1$, (B.1.2) is approximately equal to $p_z^2 z^2$
- if $\frac{p_z}{p_r} \frac{z}{r} \ll 1$, (B.1.2) is approximately equal to $p_r^2 r^2 \approx p_r^2 r^2 + p_z^2 z^2$
- if $\frac{p_z}{p_r} \frac{z}{r} \sim 1$, (B.1.2) is approximately equal to $p_r^2 r^2 (\frac{p_z^2}{p_r^2} \frac{z^2}{r^2} + 3) = p_z^2 z^2 + 3 p_r^2 r^2$

Finally, we can conclude that:

- For $r \sim z$, $r \ll z$ or $r \gg z \wedge \frac{p_z z}{p_r r} \gg 1$, (B.1.1) can be approximated by
$$r^2 + (1 + p_z^2) z^2$$
- For $r \gg z \wedge \frac{p_z z}{p_r r} \ll 1$, (B.1.1) can be approximated by
$$(1 + p_r^2) r^2 + (1 + p_z^2) z^2 \approx r^2 + (1 + p_z^2) z^2$$
- For $r \gg z \wedge \frac{p_z z}{p_r r} \sim 1$
$$(1 + 3p_r^2) r^2 + (1 + p_z^2) z^2 \approx r^2 + (1 + p_z^2) z^2$$

□

B.2 Some Properties of Special Relativity

This section summaries some consequences of special relativity, which are already discussed in the literature, for example in the textbook of classical electrodynamics [42]. Consider one inertial frame $\mathcal{K}'$ moving with a velocity $\boldsymbol{u}$ (corresponding to the momentum $\boldsymbol{p}_u$) relative to another frame $\mathcal{K}$. The space-time coordinates of an event in these two frames follow the transformation

$$(c_0t') = \gamma_u(c_0t) - \boldsymbol{p}_u \cdot \boldsymbol{x}, \quad x'_{\parallel} = \gamma_u x_{\parallel} - p_u(c_0t), \quad x'_{\perp} = x_{\perp} \tag{B.2.1}$$

where $\parallel$ and $\perp$ denote the components parallel and perpendicular to $\boldsymbol{p}_u$. Here, we use p_u to denote the magnitude of $\boldsymbol{p}_u$ (*i.e,* $p_u := \|\boldsymbol{p}_u\|_2$). Similar to the space-time coordinates, the four-momentum (energy and momentum) in $\mathcal{K}'$ and $\mathcal{K}$ follows the transformation

$$\gamma' = \gamma_u\gamma - \boldsymbol{p}_u \cdot \boldsymbol{p}, \quad p'_{\parallel} = \gamma_u p_{\parallel} - p_u\gamma, \quad p'_{\perp} = p_{\perp}. \tag{B.2.2}$$

The transformation formulas for the position and momentum above are expressed in the component-wise form because of its convenience for the theoretical derivation. These transformations can also be expressed in vector form

$$\boldsymbol{x}' = \boldsymbol{x} + \frac{1}{\gamma_u + 1}(\boldsymbol{x} \cdot \boldsymbol{p}_u)\boldsymbol{p}_u - c_0 t \boldsymbol{p}_u, \tag{B.2.3}$$

$$\boldsymbol{p}' = \boldsymbol{p} + \frac{1}{\gamma_u + 1}(\boldsymbol{p} \cdot \boldsymbol{p}_u)\boldsymbol{p}_u - \gamma \boldsymbol{p}_u. \tag{B.2.4}$$

B.3 Data Structure of the Cluster Tree

Although the FMM solvers developed in this work are written in the Julia programming language, we use C-style pseudocode to illustrate the data structure of the cluster tree. The data structure of the cluster tree can be naively designed as follow:

```
struct Cluster {
  size_t npar;
  value_type (*positions)[3]; // array of particle positions
  Cluster* children;
}
```

However, this naive implementation may require a significant amount of memory as the position of particles in each cluster is explicitly stored. For a balanced cluster tree describing an N-particles cluster, the number of particle positions to be stored is $N \log_2 N$. If we have $N = 2 \times 10^6$, a memory of roughly 1 GB will need to be allocated during the construction of the cluster tree and this could cause a performance bottleneck.

Alternatively, one may store the particle positions outside the structure and declare an external array `parindices` to store the indices of all the particles. In such a case, the data structure can be expressed as

```
size_t parindices[N]
value_type positions[N][3]
struct Cluster {
  size_t pindex_lo;
  size_t pindex_hi;
  Cluster* children;
}
```

If the elements of `parindices` are arranged in such a manner that the indices of the particles in the cluster S occupy in `parindices` contiguously from l-th (`pindex_lo`) to h-th (`pindex_hi`) location, their values in `parindices` (*i.e.*, their indices) can be expressed as

$$p_l, p_{l+1}, \dots, p_h.$$

In the subdivision of S, we first determine the splitting coordinate direction $g \in \{x, y, z\}$ from bbox(S) and permute the elements in `parindices` that

$$p'_l, \dots, p'_{\left\lfloor \frac{l+h}{2} \right\rfloor}, p'_{\left\lfloor \frac{l+h}{2} \right\rfloor+1}, \dots p'_h$$

and

$$g_{p'_i} \begin{cases} \leq g_{p'_{\left\lfloor \frac{l+h}{2} \right\rfloor}} & \text{if } \left\lfloor \frac{l+h}{2} \right\rfloor \geq i \geq l, \\ > g_{p'_{\left\lfloor \frac{l+h}{2} \right\rfloor}} & \text{if } \left\lfloor \frac{l+h}{2} \right\rfloor < i \leq h. \end{cases}$$

This permutation enables the objects of the children clusters `S1` and `S2` to access their belonging particle indices by:

`S1.pindex_lo=`l, `S1.pindex_hi=`$\lfloor \frac{l+h}{2} \rfloor$,
`S2.pindex_lo=`$\lfloor \frac{l+h}{2} \rfloor + 1$, `S2.pindex_hi=`$h$,
`parindicies[S1.pindex_lo]`, ..., `parindicies[S1.pindex_hi]`,
`parindicies[S2.pindex_lo]`, ..., `parindicies[S2.pindex_hi]`.

In this study, the permutation is implemented by the Quickselect algorithm with the Lomuto partition scheme [104]. The complexity on average is $\mathcal{O}(N)$ and can be $\mathcal{O}(N^2)$ in the worst case scenario.

Appendix C

C.1 Definition of Cumulative Local Field

Lemma C.1.1. *Assume a target point $\boldsymbol{x}_i$ is contained in a sequence of clusters $\{S^l \mid l = 0, \dots, k\}$ of each level l with $S^{l+1} \subset S^l$ and $S^0 = S$. The total force-field of the macro particles from this sequence of clusters transferred to this target point can be calculated by*

$$f(\boldsymbol{x}_i) = \sum_{l=0}^{k} \sum_{\boldsymbol{\mu}} \ell_{S^l,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^l,\boldsymbol{\mu}} = \sum_{\boldsymbol{\mu}} \mathcal{L}_{S^k,\boldsymbol{\mu}} \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i),$$

where $\mathcal{L}_{S^l}$ is defined as

$$\mathcal{L}_{S^l,\boldsymbol{\mu}} := L_{S^l,\boldsymbol{\mu}} + \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S^{l-1},\boldsymbol{\mu}'} \cdot \ell_{S^{l-1},\boldsymbol{\mu}'}(\boldsymbol{\xi}_{S^l,\boldsymbol{\mu}}) \quad \textit{with} \quad \mathcal{L}_{S^0,\boldsymbol{\mu}} := L_{S^0,\boldsymbol{\mu}}.$$

Proof. We can prove this statement by mathematical induction. By the definition above, the statement automatically holds for the case $l = 0$. We only need to prove the statement holds for the case $l = k$ provided that it is true for $l = k - 1$. Thus, we have

$$\begin{aligned}
&\sum_{l=0}^{k} \sum_{\boldsymbol{\mu}} \ell_{S^l,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^l,\boldsymbol{\mu}} \\
=\; & \sum_{\boldsymbol{\mu}} \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^k,\boldsymbol{\mu}} + \sum_{l=0}^{k-1} \sum_{\boldsymbol{\mu}} \ell_{S^l,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^l,\boldsymbol{\mu}} \\
=\; & \sum_{\boldsymbol{\mu}} \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^k,\boldsymbol{\mu}} + \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S^{k-1},\boldsymbol{\mu}'} \cdot \ell_{S^{k-1},\boldsymbol{\mu}'}(\boldsymbol{x}_i) \quad \text{(by assumption)} \\
\overset{(4.1.9)}{=}\; & \sum_{\boldsymbol{\mu}} \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i) L_{S^k,\boldsymbol{\mu}} + \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S^{k-1},\boldsymbol{\mu}'} \sum_{\boldsymbol{\mu}} \ell_{S^{k-1},\boldsymbol{\mu}'}(\boldsymbol{\xi}_{S^k,\boldsymbol{\mu}}) \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i) \\
=\; & \sum_{\boldsymbol{\mu}} \underbrace{\left(L_{S^k,\boldsymbol{\mu}} + \sum_{\boldsymbol{\mu}'} \mathcal{L}_{S^{k-1},\boldsymbol{\mu}'} \cdot \ell_{S^{k-1},\boldsymbol{\mu}'}(\boldsymbol{\xi}_{S^k,\boldsymbol{\mu}}) \right)}_{=\mathcal{L}_{S^k,\boldsymbol{\mu}}} \ell_{S^k,\boldsymbol{\mu}}(\boldsymbol{x}_i).
\end{aligned}$$

□

Bibliography

[1] Y. Hwu, G. Margaritondo, Synchrotron radiation and X-ray free-electron lasers (X-FELs) explained to all users, active and potential, Journal of Synchrotron Radiation 28 (3) (2021) 1014–1029. doi:10.1107/S1600577521003325.

[2] W. C. Röntgen, Über eine neue Art von Strahlen, Annalen der Physik 300 (1) (1898) 1–11. doi:10.1002/andp.18983000102.

[3] R. Behling, X-ray sources: 125 years of developments of this intriguing technology, Physica Medica 79 (2020) 162–187. doi:10.1016/j.ejmp.2020.07.021.

[4] T.-H. Nguyen, C. J. Lee, R. jun Park, G.-H. Jin, S. Y. Kim, I. Jeon, Note: A simple-structured anode exchangeable X-ray tube, Review of Scientific Instruments 84 (2013) 056108. doi:10.1063/1.4807755.

[5] O. Hemberg, M. Otendal, H. M. Hertz, Liquid-metal-jet anode electron-impact x-ray source, Applied Physics Letters 83 (7) (2003) 1483–1485. doi:10.1063/1.1602157.

[6] M. Otendal, T. Tuohimaa, U. Vogt, H. M. Hertz, A 9keV electron-impact liquid-gallium-jet x-ray source, Review of Scientific Instruments 79 (1) (2008) 016102. doi:10.1063/1.2833838.

[7] B. Hornberger, J. Kasahara, M. Gifford, R. Ruth, R. Loewen, A compact light source providing high-flux, quasi-monochromatic, tunable X-rays in the laboratory, in: A. Murokh, D. Spiga (Eds.), Proceedings Volume Advances in Laboratory-based X-Ray Sources, Optics, and Applications VII, Vol. 11110, International Society for Optics and Photonics, SPIE, 2019, p. 1111003. doi:10.1117/12.2527356.

[8] K.-J. Kim, Z. Huang, R. Lindberg, Synchrotron Radiation and Free-Electron Lasers: Principles of Coherent X-Ray Generation, Cambridge University Press, 2017. doi:10.1017/9781316677377.

[9] O. Grimm, H. Delsim-Hashemi, J. Rossbach, V. Balandin, N. Golubeva, Transverse Electron Beam Size Effect on the Bunch Profile Determination with Coherent Radiation Diagnostics, in: I. Andrian, C. Petit-Jean-Genaz (Eds.), Proceedings of EPAC08, Vol. 0806233, 2008, p. TUPC030.
URL https://accelconf.web.cern.ch/e08/papers/tupc030.pdf

[10] C. Bostedt, S. Boutet, D. M. Fritz, Z. Huang, H. J. Lee, H. T. Lemke, A. Robert, W. F. Schlotter, J. J. Turner, G. J. Williams, Linac Coherent Light Source: The first five years, Reviews of Modern Physics 88 (1) (2016) 015007. doi:10.1103/RevModPhys.88.015007.

[11] C. Pellegrini, The development of XFELs, Nature Reviews Physics 2 (7) (2020) 330–331. doi:10.1038/s42254-020-0197-1.

[12] C.-C. Kuo, C.-T. Chen, J.-Y. Chen, M.-S. Chiu, P. Chou, K.-T. Hsu, Y. Liu, G.-H. Luo, H.-J. Tsai, F.-H. Tseng, Commissioning of the Taiwan Photon Source, in: Proceedings of the 6th International Particle Accelerator Conference, 2015, p. TUXC3. doi:10.18429/JACoW-IPAC2015-TUXC3.

[13] C. G. Schroer, I. Agapov, W. Brefeld, R. Brinkmann, Y.-C. Chae, H.-C. Chao, M. Eriksson, J. Keil, X. Nuel Gavaldà, R. Röhlsberger, O. H. Seeck, M. Sprung, M. Tischer, R. Wanzenberg, E. Weckert, PETRA IV: the ultralow-emittance source project at DESY, Journal of Synchrotron Radiation 25 (5) (2018) 1277–1290. doi:10.1107/S1600577518008858.

[14] H. Tanaka, Current Status of the SPring-8 Upgrade Project, Synchrotron Radiation News 27 (6) (2014) 23–26. doi:10.1080/08940886.2014.970935.

[15] P. F. Tavares, S. C. Leemann, M. Sjöström, Å. Andersson, The MAXIV storage ring project, Journal of Synchrotron Radiation 21 (5) (2014) 862–877. doi:10.1107/S1600577514011503.

[16] W. Decking, F. Brinker, L. Fröhlich, R. Kammering, T. Limberg, S. Liu, D. Nölle, M. Omet, M. Scholz, T. Wamsat, Status of the European XFEL, in: Proceedings of the 10th International Particle Accelerator Conference, 2019, p. TUPRB020. doi:10.18429/JACoW-IPAC2019-TUPRB020.

[17] L. F. DiMauro, J. Arthur, N. Berrah, J. Bozek, J. N. Galayda, J. Hastings, Progress report on the LCLS XFEL at SLAC, Journal of Physics: Conference Series 88 (1) (2007) 012058. doi:10.1088/1742-6596/88/1/012058.

[18] T. Shintake, Review of the worldwide SASE FEL development, in: 2007 IEEE Particle Accelerator Conference (PAC), 2007, pp. 89–93. doi:10.1109/PAC.2007.4440331.

[19] E. Esarey, S. K. Ride, P. Sprangle, Nonlinear Thomson scattering of intense laser pulses from beams and plasmas, Physical Review E 48 (4) (1993) 3003–3021. doi:10.1103/PhysRevE.48.3003.

[20] G. A. Krafft, G. Priebe, Compton Sources of Electromagnetic Radiation, Reviews of Accelerator Science and Technology 03 (01) (2010) 147–163. doi:10.1142/S1793626810000440.

[21] K. Phuoc, S. Corde, C. Thaury, V. Malka, A. Tafzi, J.-P. Goddet, R. Shah, S. Sebban, A. Rousse, All-optical Compton gamma-ray source, Nature Photonics 6 (2012) 308–311. doi:10.1038/nphoton.2012.82.

[22] D. H. Bilderback, P. Elleaume, E. Weckert, Review of third and next generation synchrotron light sources, Journal of Physics B: Atomic, Molecular and Optical Physics 38 (9) (2005) S773–S797. doi:10.1088/0953-4075/38/9/022.

[23] W. S. Graves, J. Bessuille, P. Brown, S. Carbajo, V. Dolgashev, K.-H. Hong, E. Ihloff, B. Khaykovich, H. Lin, K. Murari, E. A. Nanni, G. Resta, S. Tantawi, L. E. Zapata, F. X. Kärtner, D. E. Moncton, Compact x-ray source based on burst-mode inverse Compton scattering at 100 kHz, Physical Review Accelerators and Beams 17 (2014) 120701. doi:10.1103/PhysRevSTAB.17.120701.

[24] F. Kärtner, F. Ahr, A.-L. Calendron, H. Çankaya, S. Carbajo, G. Chang, G. Cirmi, K. Dörner, U. Dorda, A. Fallahi, A. Hartin, M. Hemmer, R. Hobbs, Y. Hua, W. Huang, R. Letrun, N. Matlis, V. Mazalova, O. Mücke, E. Nanni, W. Putnam, K. Ravi, F. Reichert, I. Sarrou, X. Wu, A. Yahaghi, H. Ye, L. Zapata, D. Zhang, C. Zhou, R. Miller, K. Berggren, H. Graafsma, A. Meents, R. Assmann, H. Chapman, P. Fromme, AXSIS: Exploring the frontiers in attosecond X-ray science, imaging and spectroscopy, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 829 (2016) 24–29. doi:10.1016/j.nima.2016.02.080.

[25] K. Deitrick, J. Delayen, G. Krafft, High Average Brilliance Compact Inverse Compton Light Source, in: Proceedings of the 8th International Particle Accelerator Conference, no. 8 in International Particle Accelerator Conference, JACoW, Geneva, Switzerland, 2017, pp. 932–935. doi:10.18429/JACoW-IPAC2017-MOPVA036.

[26] K. E. Deitrick, G. A. Krafft, B. Terzić, J. R. Delayen, High-brilliance, high-flux compact inverse Compton light source, Physical Review Accelerators and Beams 21 (2018) 080703. doi:10.1103/PhysRevAccelBeams.21.080703.

[27] W. Graves, et al., ASU Compact XFEL, in: Proceedings of the 38th International Free-Electron Laser Conference, no. 38 in International Free Electron Laser Conference, JACoW, Geneva, Switzerland, 2018, pp. 225–228. doi:10.18429/JACoW-FEL2017-TUB03.

[28] T. Brümmer, A. Debus, R. Pausch, J. Osterhoff, F. Grüner, Design study for a compact laser-driven source for medical x-ray fluorescence imaging, Physical Review Accelerators and Beams 23 (3) (2020) 031601. doi:10.1103/PhysRevAccelBeams.23.031601.

[29] B. Günther, R. Gradl, C. Jud, E. Eggl, J. Huang, S. Kulpe, K. Achterhold, B. Gleich, M. Dierolf, F. Pfeiffer, The versatile X-ray beamline of the Munich Compact Light Source: design, instrumentation and applications, Journal of Synchrotron Radiation 27 (5) (2020) 1395–1414. doi:10.1107/S1600577520008309.

[30] A. Variola, J. Haissinski, A. Loulergue, F. Zomer, et al., ThomX Technical Design Report, Tech. rep., ThomX - Dept. Accelerateurs (2014).

[31] K. Dupraz, M. Alkadi, M. Alves, L. Amoudry, D. Auguste, J.-L. Babigeon, M. Baltazar, A. Benoit, J. Bonis, J. Bonenfant, C. Bruni, K. Cassou, J.-N. Cayla, T. Chabaud, I. Chaikovska, S. Chance, V. Chaumat, R. Chiche, A. Cobessi, P. Cornebise, O. Dalifard, N. Delerue, R. Dorkel, D. Douillet, J.-P. Dugal, N. El Kamchi, M. El Khaldi, E. Ergenlik, P. Favier, M. Fernandez, A. Gamelin, J.-F. Garaut, L. Garolfi, P. Gauron, F. Gauthier, A. Gonnin, D. Grasset, E. Guerard, H. Guler, J. Haissinski, E. Herry, G. Iaquaniello, M. Jacquet, E. Jules, V. Kubytskyi, M. Langlet, T. Le Barillec, J.-F. Ledu, D. Leguidec, B. Leluan, P. Lepercq, F. Letellier-Cohen, R. Marie, J.-C. Marrucho, A. Martens, C. Mageur, G. Mercadier, B. Mercier, E. Mistretta, H. Monard, A. Moutardier, O. Neveu, D. Nutarelli, M. Omeich, Y. Peinaud, Y. Petrilli, M. Pichet, E. Plaige, C. Prévost, P. Rudnicky, V. Soskov, M. Taurigna-Quéré, S. Trochet, C. Vallerand, O. Vitez, F. Wicek, S. Wurth,

F. Zomer, P. Alexandre, R. Ben El Fekih, P. Berteaud, F. Bouvet, R. Cuoq, A. Diaz, Y. Dietrich, M. Diop, D. Pedeau, E. Dupuy, F. Marteau, F. Bouvet, A. Gamelin, D. Helder, N. Hubert, J. Veteran, M. Labat, A. Lestrade, A. Letrésor, R. Lopes, A. Loulergue, M. Louvet, M. Louvet, P. Marchand, M. El Ajjouri, D. Muller, A. Nadji, L. Nadolski, R. Nagaoka, S. Petit, J.-P. Pollina, F. Ribeiro, M. Ros, J. Salvia, S. Bobault, M. Sebdaoui, R. Sreedharan, Y. Bouanai, J.-L. Hazemann, J.-L. Hodeau, E. Roy, P. Jeantet, J. Lacipière, P. Robert, J.-M. Horodynski, H. Bzyl, C. Chapelle, M. Biagini, P. Walter, A. Bravin, W. Del Net, E. Lahéra, O. Proux, H. Elleaume, E. Cormier, The ThomX ICS source, Physics Open 5 (2020) 100051. doi:10.1016/j.physo.2020.100051.

[32] Y.-K. Kan, F. X. Kärtner, S. Le Borne, D. Ruprecht, J.-P. M. Zemke, Parallel computation of Inverse Compton Scattering radiation spectra based on Liénard-Wiechert potentials, Journal of Computational Physics 473 (2023) 111724. doi:10.1016/j.jcp.2022.111724.

[33] Y.-K. Kan, F. X. Kärtner, S. Le Borne, J.-P. M. Zemke, Relativistic space-charge field calculation by interpolation-based treecode, Computer Physics Communications 286 (2023) 108668. doi:10.1016/j.cpc.2023.108668.

[34] K. Flöttmann, ASTRA (2018).
URL https://www.desy.de/~mpyflo

[35] A. G. R. Thomas, Algorithm for calculating spectral intensity due to charged particles in arbitrary motion, Physical Review Accelerators and Beams 13 (2) (2010) 020702. doi:10.1103/PhysRevSTAB.13.020702.

[36] J. T. Frederiksen, T. Haugbølle, M. V. Medvedev, A. Nordlund, RADIATION SPECTRAL SYNTHESIS OF RELATIVISTIC FILAMENTATION, The Astrophysical Journal Letters 722 (1) (2010) L114. doi:10.1088/2041-8205/722/1/L114.

[37] T. Haugbølle, J. T. Frederiksen, A. Nordlund, PHOTON-PLASMA: A modern high-order particle-in-cell code, Physics of Plasmas 20 (6) (2013) 062904. doi:10.1063/1.4811384.

[38] R. Pausch, et al., Computing Angularly-resolved Far Field Emission Spectra in Particle-in-cell Codes using GPUs, in: Proceedings of the 5th International Particle Accelerator Conference, 2014, p. MOPRI069. doi:10.18429/JACoW-IPAC2014-MOPRI069.

[39] R. Pausch, A. Debus, R. Widera, K. Steiniger, A. Huebl, H. Burau, M. Bussmann, U. Schramm, How to test and verify radiation diagnostics simulations within particle-in-cell frameworks, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 740 (2014) 250–256. doi:10.1016/j.nima.2013.10.073.

[40] R. Pausch, Electromagnetic Radiation from Relativistic Electrons as Characteristic Signature of their Dynamics, Diploma Thesis, Technische Universität Dresden (12 2012). doi:10.5281/zenodo.843510.

[41] A. Sell, F. X. Kärtner, Attosecond electron bunches accelerated and compressed by radially polarized laser pulses and soft-x-ray pulses from optical undulators, Journal of Physics B: Atomic, Molecular and Optical Physics 47 (1) (2013) 015601. doi:10.1088/0953-4075/47/1/015601.

[42] J. D. Jackson, Classical Electrodynamics, 3rd Edition, Wiley, New York, NY, 1998.

[43] R. Ryne, C. Mitchell, J. Qiang, B. Carlsten, N. Yampolsky, Using a Lienard-Wiechert Solver to Study Coherent Synchrotron Radiation Effects, in: Proceedings of the 35th International Free-Electron Laser Conference, 2013, pp. 17–23.
URL https://accelconf.web.cern.ch/FEL2013/papers/moocno04.pdf

[44] J. P. Boris, Relativistic plasma simulation-optimization of a hybrid code, in: Proceedings of 4th Conference on Numerical Simulations of Plasmas, 1970, pp. 3–67.

[45] B. Ripperda, F. Bacchini, J. Teunissen, C. Xia, O. Porth, L. Sironi, G. Lapenta, R. Keppens, A Comprehensive Comparison of Relativistic Particle Integrators, The Astrophysical Journal Supplement Series 235 (1) (2018) 21. doi:10.3847/1538-4365/aab114.

[46] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, Julia: A Fresh Approach to Numerical Computing, SIAM Review 59 (1) (2017) 65–98. doi:10.1137/141000671.

[47] K. E. Deitrick, Inverse Compton Light Source: A Compact Design Proposal, Ph.D. thesis, Old Dominion University (2017).
URL https://digitalcommons.odu.edu/physics_etds/7/

[48] K. Deitrick, G. Krafft, B. Terzić, J. Delayen, High-brilliance, high-flux compact inverse Compton light source, Physical Review Accelerators and Beams 21 (8) (2018) 080703. doi:10.1103/PhysRevAccelBeams.21.080703.

[49] J. W. Luginsland, Y. Y. Lau, R. J. Umstattd, J. J. Watrous, Beyond the Child–Langmuir law: A review of recent results on multidimensional space-charge-limited flow, Physics of Plasmas 9 (5) (2002) 2371–2376. doi:10.1063/1.1459453.

[50] B. J. Siwick, J. R. Dwyer, R. E. Jordan, R. J. D. Miller, Ultrafast electron optics: Propagation dynamics of femtosecond electron packets, Journal of Applied Physics 92 (3) (2002) 1643–1648. doi:10.1063/1.1487437.

[51] S. Collin, M. Merano, M. Gatri, S. Sonderegger, P. Renucci, J.-D. Ganière, B. Deveaud, Transverse and longitudinal space-charge-induced broadenings of ultrafast electron packets, Journal of Applied Physics 98 (9) (2005) 094910. doi:10.1063/1.2128494.

[52] J. M. Dawson, Particle simulation of plasmas, Reviews of Modern Physics 55 (1983) 403–447. doi:10.1103/RevModPhys.55.403.

[53] T. Esirkepov, Exact charge conservation scheme for Particle-in-Cell simulation with an arbitrary form-factor, Computer Physics Communications 135 (2) (2001) 144–153. doi:10.1016/S0010-4655(00)00228-9.

[54] T. Umeda, Y. Omura, T. Tominaga, H. Matsumoto, A new charge conservation method in electromagnetic particle-in-cell simulations, Computer Physics Communications 156 (1) (2003) 73–85. doi:10.1016/S0010-4655(03)00437-5.

[55] K. Yee, Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media, IEEE Transactions on Antennas and Propagation 14 (3) (1966) 302–307. doi:10.1109/TAP.1966.1138693.

[56] Q. H. Liu, The pseudospectral time-domain (PSTD) method: a new algorithm for solutions of Maxwell's equations, in: IEEE Antennas and Propagation Society International Symposium 1997. Digest, Vol. 1, 1997, pp. 122–125. doi:10.1109/APS.1997.630102.

[57] C. K. Birdsall, A. B. Langdon, Plasma Physics via Computer Simulation, CRC press, 2018.

[58] K. Flöttmann, S. Lidia, P. Piot, Recent improvements to the ASTRA particle tracking code, Tech. rep., Lawrence Berkeley National Lab (LBNL), USA (2003).

[59] J. Qiang, Symplectic multiparticle tracking model for self-consistent space-charge simulation, Physical Review Accelerators and Beams 20 (2017) 014203. doi:10.1103/PhysRevAccelBeams.20.014203.

[60] A. Taflove, S. C. Hagness, M. Piket-May, Computational Electromagnetics: The Finite-Difference Time-Domain Method, The Electrical Engineering Handbook 3 (2005).

[61] I. Haber, R. Lee, H. Klein, J. Boris, Advances in electromagnetic simulation techniques, in: Proc. Sixth Conf. Num. Sim. Plasmas, Berkeley, CA, 1973, pp. 46–48.

[62] R. Lehé, J.-L. Vay, Review of Spectral Maxwell Solvers for Electromagnetic Particle-in-Cell: Algorithms and Advantages, in: Proceedings of 13th International Computational Accelerator Physics Conference, no. 13 in International Computational Accelerator Physics Conference, JACoW Publishing, Geneva, Switzerland, 2019, pp. 345–349. doi:10.18429/JACoW-ICAP2018-WEPLG05.

[63] H. Zhang, H. Huang, R. Li, J. Chen, L.-S. Luo, Fast multipole method using Cartesian tensor in beam dynamic simulation, AIP Conference Proceedings 1812 (1) (2017) 050001. doi:10.1063/1.4975862.

[64] M. Reiser, Theory and Design of Charged Particle Beams, John Wiley & Sons, 2008.

[65] M. Gordon, S. Van Der Geer, J. Maxson, Y.-K. Kim, Point-to-point Coulomb effects in high brightness photoelectron beam lines for ultrafast electron diffraction, Physical Review Accelerators and Beams 24 (8) (2021) 084202. doi:10.1103/PhysRevAccelBeams.24.084202.

[66] J. Barnes, P. Hut, A hierarchical O(N log N) force-calculation algorithm, Nature 324 (6096) (1986) 446–449. doi:10.1038/324446a0.

[67] S. Börm, Efficient numerical methods for non-local operators: H^2-matrix compression, algorithms and analysis, Vol. 14 of EMS Tracts in Mathematics, European Mathematical Society (EMS), Zürich, 2010. doi:10.4171/091.

[68] W. Hackbusch, Hierarchical Matrices: Algorithms and Analysis, Vol. 49, Springer, 2015. doi:10.1007/978-3-662-47324-5.

[69] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, Journal of Computational Physics 73 (2) (1987) 325–348. doi:10.1016/0021-9991(87)90140-9.

[70] R. Capuzzo-Dolcetta, P. Miocchi, A Comparison between the Fast Multipole Algorithm and the Tree-Code to Evaluate Gravitational Forces in 3-D, Journal of Computational Physics 143 (1) (1998) 29–48. doi:10.1006/jcph.1998.5949.

[71] M. S. Warren, J. K. Salmon, A portable parallel particle program, Computer Physics Communications 87 (1-2) (1995) 266–290. doi:10.1016/0010-4655(94)00177-4.

[72] W. Dehnen, A Hierarchical O(N) Force Calculation Algorithm, Journal of Computational Physics 179 (1) (2002) 27–42. doi:10.1006/jcph.2002.7026.

[73] H. Cheng, L. Greengard, V. Rokhlin, A Fast Adaptive Multipole Algorithm in Three Dimensions, Journal of Computational Physics 155 (2) (1999) 468–498. doi:10.1006/jcph.1999.6355.

[74] S. Pfalzner, P. Gibbon, A 3D hierarchical tree code for dense plasma simulation, Computer physics communications 79 (1) (1994) 24–38. doi:10.1016/0010-4655(94)90227-5.

[75] D. Thomas, J. Holgate, A treecode to simulate dust–plasma interactions, Plasma Physics and Controlled Fusion 59 (2) (2016) 025002. doi:10.1088/1361-6587/59/2/025002.

[76] H. Zhang, J. Portman, Z. Tao, P. Duxbury, C.-Y. Ruan, K. Makino, M. Berz, The Differential Algebra Based Multiple Level Fast Multipole Algorithm for 3D Space Charge Field Calculation and Photoemission Simulation, Microscopy and Microanalysis 21 (S4) (2015) 224–229. doi:10.1017/S1431927615013410.

[77] F. W. Jones, A hybrid fast-multipole technique for space-charge tracking with halos, AIP Conference Proceedings 448 (1) (1998) 359–370. doi:10.1063/1.56759.

[78] L. J. Wong, A. Fallahi, F. X. Kärtner, Compact electron acceleration and bunch compression in THz waveguides, Optics Express 21 (8) (2013) 9792–9806. doi:10.1364/OE.21.009792.

[79] I. V. Kochikov, R. J. Dwayne Miller, A. A. Ischenko, Relativistic Modeling of Ultra-Short Electron Pulse Propagation, Journal of Experimental and Theoretical Physics 128 (3) (2019) 333–340. doi:10.1134/S1063776119020201.

[80] S. A. Schmid, H. D. Gersem, E. Gjonaj, REPTIL - A Relativistic 3D Space Charge Particle Tracking Code Based on the Fast Multipole Method, unpublished (01 2019).

[81] S. Schmid, H. De Gersem, M. Dohlus, E. Gjonaj, Simulating Space Charge Dominated Beam Dynamics Using FMM, in: Proceedings of 3rd North American Particle Accelerator Conference, 2019, p. WEPLE10. doi:10.18429/JACoW-NAPAC2019-WEPLE10.

[82] S. B. van der Geer, M. J. de Loos, General Particle Tracer User Manual (Version 3.43), Pulsar Physics.

[83] L. Wang, A kernel-independent treecode based on barycentric Lagrange interpolation, Communications in Computational Physics 28 (4) (2020) 1415–1436. doi:10.4208/cicp.OA-2019-0177.

[84] J. L. Bentley, Multidimensional binary search trees used for associative searching, Communications of the ACM 18 (9) (1975) 509–517. doi:10.1145/361002.361007.

[85] K. Conrad, Equivalence of norms, Expository Paper, University Of Connecticut, Storrs 17 (2018).
URL https://kconrad.math.uconn.edu/blurbs/gradnumthy/equivnorms.pdf

[86] J.-P. Berrut, L. N. Trefethen, Barycentric Lagrange Interpolation, SIAM Review 46 (3) (2004) 501–517. doi:10.1137/S0036144502417715.

[87] W. Fong, E. Darve, The black-box fast multipole method, Journal of Computational Physics 228 (23) (2009) 8712–8725. doi:10.1016/j.jcp.2009.08.031.

[88] L. Wilson, N. Vaughn, R. Krasny, A GPU-accelerated fast multipole method based on barycentric Lagrange interpolation and dual tree traversal, Computer Physics Communications 265 (2021) 108017. doi:10.1016/j.cpc.2021.108017.

[89] L. Wilson, Development and Application of Numerical Methods in Biomolecular Solvation, Ph.D. thesis, University of Michigan (2021). doi:10.7302/1547.

[90] J. Liu, M. Robson, T. Quinn, M. Kulkarni, Efficient GPU Tree Walks for Effective Distributed N-Body Simulations, in: Proceedings of the ACM International Conference on Supercomputing, ICS '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 24–34. doi:10.1145/3330345.3330348.

[91] S. Wienke, P. Springer, C. Terboven, D. an Mey, Openacc — first experiences with real-world applications, in: C. Kaklamanis, T. Papatheodorou, P. G. Spirakis (Eds.), Euro-Par 2012 Parallel Processing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 859–870.

[92] M. Burtscher, K. Pingali, Chapter 6 - An Efficient CUDA Implementation of the Tree-Based Barnes Hut n-Body Algorithm, in: W. mei W. Hwu (Ed.), GPU Computing Gems Emerald Edition, Applications of GPU Computing Series, Morgan Kaufmann, Boston, 2011, pp. 75–92. doi:10.1016/B978-0-12-384988-5.00006-1.

[93] Leighton Wilson, private communication (2022).

[94] L. Wilson, N. Vaughn, BaryTree (2021).
URL https://github.com/Treecodes/BaryTree

[95] J. Cheng, M. Grossman, T. McKercher, Professional CUDA C Programming, John Wiley & Sons, 2014.

[96] A. Ferris, other contributors, StaticArrays (2016).
URL https://github.com/JuliaArrays/StaticArrays.jl

[97] Julia Computing, other contributors, CUDA.jl (2016).
URL https://github.com/JuliaArrays/StaticArrays.jl

[98] T. Besard, C. Foket, B. D. Sutter, Effective Extensible Programming: Unleashing Julia on GPUs, IEEE Transactions on Parallel and Distributed Systems 30 (4) (2019) 827–841. doi:10.1109/tpds.2018.2872064.

[99] M. Fakhari, Y.-K. Kan, F. Kärtner, N. Matlis, M. Vahdani, Start-to-End Simulations of a THz-Driven ICS Source, in: Proceedings of the 13th International Particle Accelerator Conference, JACoW Publishing, Geneva, Switzerland, 2022, pp. 631–633. doi:10.18429/JACoW-IPAC2022-MOPOMS005.

[100] K. E. Priebe, C. Rathje, S. V. Yalunin, T. Hohage, A. Feist, S. Schäfer, C. Ropers, Attosecond electron pulse trains and quantum state reconstruction in ultrafast transmission electron microscopy, Nature Photonics 11 (12) (2017) 793–797. doi:10.1038/s41566-017-0045-8.

[101] M. Kozák, J. McNeur, K. J. Leedle, H. Deng, N. Schönenberger, A. Ruehl, I. Hartl, J. Harris, R. Byer, P. Hommelhoff, Optical gating and streaking of free electrons with sub-optical cycle precision, Nature communications 8 (1) (2017) 1–7. doi:10.1038/ncomms14342.

[102] Y. Morimoto, P. Baum, Diffraction and microscopy with attosecond electron pulse trains, Nature Physics 14 (3) (2018) 252–256. doi:10.1038/s41567-017-0007-6.

[103] P. Dombi, Z. Pápa, J. Vogelsang, S. V. Yalunin, M. Sivis, G. Herink, S. Schäfer, P. Groß, C. Ropers, C. Lienau, Strong-field nano-optics, Reviews of Modern Physics 92 (2020) 025003. doi:10.1103/RevModPhys.92.025003.

[104] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to algorithms, 4th Edition, MIT press, 2022.

Eidesstattliche Versicherung / Declaration on oath

Hiermit versichere ich an Eides statt, die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt zu haben.

Hamburg, den 07.12.2022

Unterschrift der Doktorandin / des Doktoranden