TECHNISCHE
UNIVERSITÄT
DRESDEN

Electromagnetic Radiation from Relativistic Electrons as Characteristic Signature of their Dynamics

Diploma Thesis
for the acquisition of the academic degree
Diplom-Physiker

submitted by

Richard Pausch
born on September 29th 1986 in Dresden

Institute of Nuclear and Particle Physics
Technische Universität Dresden
2012

submitted on December 13, 2012

Primary Reviewer:	Prof. Dr. Ulrich Schramm
Secondary Reviewer:	Dr. Michael Bussmann

Errata 2017-08-15:

Equation (4.18) should state:

$$\omega < \frac{\pi}{\Delta t \cdot \left(1 - \vec{\beta} \cdot \vec{n}\right)}$$

DOI: 10.5281/zenodo.843510

Abstract

This Diploma thesis summarizes the development, verification and application of three different parallel codes for computing electromagnetic fields generated by particles moving at relativistic velocities. The codes are based on the classical Liénard-Wiechert potential formalism. Highly efficient numerical solutions for modeling the radiation emitted by millions to billions of particles are introduced and implemented for parallel compute architectures.
The first code allows to simulate the electromagnetic near and far field at arbitrary spatial and temporal resolution. The numerical solution of the Liénard-Wiechert potentials in the time domain used in this code is especially suited to compute strongly varying electromagnetic fields in space and time as they occur in coherent synchrotron radiation.
The second code computes the angularly resolved radiation spectrum emitted in the far field. It simply requires electron trajectories as provided by many particle simulations as an input. Due to its open structure its applications ranges from determining the beam emittance of an electron beam via Thomson scattering to predicting radiation produced in laser-plasma interactions. It is however limited by file system size and bandwidth and cannot be applied to large-scale plasma simulations.
Large-scale plasma simulations are the domain of the third code. Similarly to the second code it computes the radiation intensity per unit solid angle and unit frequency. The code is implemented for use on graphics processing units (GPUs) and integrated into the particle in cell code PIConGPU. It thus provides a highly-efficient, strongly-scalable method to compute the complete radiation spectrum emitted over the full solid angle by all particles in a laser plasma simulation. The range of frequencies spans from the infrared to the X-ray region. This allows to directly links spectral signatures to specific plasma dynamics such as electron injection and betatron oscillations occurring in laser wakefield acceleration. Such spectra can be compared to experimental measurements and can thus help to better understand the femtosecond particle dynamics in laser plasma interactions.

Zusammenfassung

Diese Diplomarbeit beschreibt die Entwicklung, Verifizierung und Anwendung von drei unterschiedlichen Programmen zur Berechnung von elektromagnetischen Feldern relativistisch bewegter Ladungen. Die entwickelten Algorithmen basieren auf den klassischen Liénard-Wiechert Potentialen. Hoch effiziente numerische Lösungen zur Simulation der Abstrahlung von Millionen bis Milliarden Teilchen wurden entwickelt und für parallele Rechnerarchitekturen implementiert.
Das erste Programm ermöglicht es, elektromagnetische Nah- und Fernfelder mit beliebiger örtlicher und zeitlicher Auflösung zu bestimmen. Das auf Lösungen der Liénard-Wiechert Potentiale basierende numerische Verfahren ist speziell dazu geeignet, örtlich und zeitlich stark veränderliche Felder zu bestimmen, wie sie beispielsweise bei kohärenter Synchrotronstrahlung auftreten.
Das zweite Programm gestattet es, Abstrahlungsspektren im Fernfeld in beliebige Richtungen zu berechnen. Dazu werden Elektronentrajektorien benötigt, die von gängigen Teilchensimulationen zur Verfügung gestellt werden können. Aufgrund der Unabhängigkeit von speziellen Simulationsprogrammen reicht der Anwendungsbereich von der Bestimmung der Emittanz von Elektronenstrahlen mittels Thomson-Streuung bis zur Vorhersage der elektromagnetischen Emissionen in Laser-Plasma-Interaktionen. Speicherplatz- und Bandbreitenlimitierungen erlauben es diesem Programm aber nicht, die Abstrahlung großskaliger Plasmasimulationen zu berechnen.
Dies ist aber mit dem dritten entwickelten Programm möglich. Ähnlich dem zweiten berechnet es die Strahlungsintensität pro Raumwinkel und Frequenz. Es wurde für die Anwendung auf Graphikkarten (GPUs) konzipiert und in die Particle-in-Cell Simulation PIConGPU integriert. Dies erlaubt es, die Abstrahlungsspektren auf einer kompletten Raumkugel für alle Teilchen einer Laser-Plasma-Simulation höchst effizient zu bestimmen. Dabei kann ein Frequenzbereich von Infrarot- bis zur Röntgenstrahlung gleichzeitig abgetastet werden. Ein Vergleich solcher Spektren mit experimentellen Beobachtungen hilft, das Verständnis der in Laser-Plasma-Interaktionen auftretenden Teilchenbewegung im Femtosekundenbereich zu erweitern.

Contents

List of Figures

1 | Introduction

Particle accelerators have started as small-scale devices accelerating electrons in vacuum tubes. Most prominently known today are the large-scale accelerators, like LHC at CERN, used as research tools in high energy physics. But accelerators also have wide industrial and medical applications. Their use ranges from particle physics to material science, from cancer treatment to ion implantation for semiconductor production and metal finishing.
The first particle accelerators used static electric fields to accelerate charged particles. This concept allows to use the accelerator structure only once, limiting the effectiveness of such accelerators.
The next generation of accelerators used oscillating fields to enable the reuse of the provided voltage, allowing to build these new type of machines more compactly.
Both concepts are limited since the structures providing the voltage can only generate field strengths of up to $\sim 100\,\mathrm{MV/m}$ [1, 2]. Otherwise, electrons from the acceleration cavities itself would leave the structure creating field emissions. This strongly limits the energy gained per length even if reusing the accelerator structure. In order to exploit the same accelerating cavity several times, circular accelerators and storage rings bend the path of the particles, guiding them on a circular orbit. Bending the path causes the particle to emit energy by radiation. In order to cope with the energy lost in this way, large bending radii are required. Independent of using static or dynamic fields in accelerators, it is necessary to build large accelerator structures to reach high particle energies.
Today, there are around a total of 26.000 accelerators of which 44% are used for radiotherapy [3]. However, hospitals as well as factories and research institutes cannot always provide the infrastructure to house a large particle accelerator. Alternatives for small-scaled accelerators are needed. Those could be provided by laser plasma accelerators. Here, the laser field causes charge imbalances in the plasma leading to high field gradients which can accelerate charged particles. These high field strength can be achieved without the risk of vacuum breakdown, because the plasma is completely ionized already.
The research field of high-intensity laser interaction with matter is fast grow-

ing. It promises applications not only for compact accelerator structures but also as ignitors for nuclear fusion and drivers for novel light sources.
Since the development of chirped pulse amplification, the laser intensity reached in experiments has grown rapidly. Nowadays, about $10^{21}\,\frac{\mathrm{W}}{\mathrm{cm}^3}$ are achieved [4]. Such high laser intensities force electrons to move at relativistic velocities. Describing their non-linear dynamics is quite difficult, thus making simulations the only feasible way to predict all details of the complex plasma dynamics.
Today's laser-plasma simulations are capable of modeling processes ranging from low energetic wakefield acceleration to the highly non-linear blow-out regime [5]. However, particle dynamics within a plasma are usually not directly accessible to experiments and it proves difficult to compare theoretical predictions with experiments.
One way to solve this problem is to observe the radiation emanating from the interaction zone. This provides an indirect look into the plasma since information about the dynamics of the laser particle interaction are contained in the radiation.
There exist several analytical solutions for radiation emitted by particles in strong laser fields. However, they only cover very special cases and are only valid for single particles. A complete analytical description taking into account ponderomotive and collective effects does not yet exist.
Thus, simulating the electromagnetic emissions from both relativistic and sub-relativistic plasma electrons is essential to better understand the imprint of electron dynamics on this radiation. [5]. This might provide new means of diagnostics if it could be achieved to link the particle dynamics in the plasma to a characteristic photon emission which would be easily observable in experiments.
Another application of such a code would be simulating novel light sources. Using Thomson scattered radiation from a laser interacting with an electron beam, one can obtain a brilliant high-energetic source of light. The mechanisms are understood quite well in analytical theory. However, when considering real laser pulses and realistic electron beams, beam emittance and pulse shapes need to be taken into account which can only be done by simulations. Modeling the radiation is essential for optimizing such light sources with respect to brightness.
Finally, such a code would have applications in simulations themselves, allowing to take electromagnetic fields into account that might not have been considered so far.

For this Diploma thesis three codes were developed and extensively tested. One is able to compute the electric and magnetic fields at arbitrary positions and times based on given electron dynamics. It allows to resolve fine field structures precisely and outperforms for such cases standard mesh-based field solvers. This program was applied to simulate influences of the electromag-

netic fields occurring in coherent synchrotron radiation.
The other two codes, one CPU the other GPU based, allow to calculate the emitted electromagnetic intensity in arbitrary directions, resolved with respect to the emitted frequencies. Computing the radiation based on simulated electron dynamics provides a way to compare particle simulations with electromagnetic emissions observed in experiments. The CPU based program *CLARA 2.0* is used to simulate Thomson scattering for beam emittance scans and laser plasma interactions. The GPU based radiation code is combined with a particle-in-cell code in order to speed up computing the radiation. This approach allows, for the first time, to simulate electromagnetic emission for laser plasma simulations at full scale.

2 | Theoretical Prerequisites

2.1 Radiation from accelerated charges

The fact that particles carrying an electric charge emit electromagnetic radiation when accelerated is one of the fundamental results of Maxwell's theory of electrodynamics. The emission can be described accurately by the Liénard-Wiechert Potentials [6]. Their derivation is rather complicated with respect to the needed mathematics, but the basic concept of radiation can be understood without deriving the formulae.
This section will introduce an example to illustrate, without too many equations, the origin of emitted radiation, while the next sections will concentrate on the exact derivation of all formulae used in the thesis.
To demonstrate the origin of electromagnetic radiation, a single particle with an electric charge moving at constant speed in one direction is assumed. This is equivalent to the situation at time t_1 illustrated in figure 2.1. The particle generates an electric field filling the space around it. Assuming the particle moves at a speed close the speed of light, the electric field surrounding the particle, which would be spherical in its rest frame, becomes Lorentz contracted and has an oblate shape in the lab frame. For simplicity, only fields perpendicular to the particle path will be considered. For this perpendicular plane, the electric field strength simply follows Coulombs law without the need to consider the Lorentz contraction.

$$\vec{E} = \frac{q}{4\pi\varepsilon_0}\frac{\vec{e}_r}{r^2} \tag{2.1}$$

In Fig. 2.1, the electric field is represented by an orange curve. For a charge moving along a straight line, the surrounding field keeps its oblate shape and sticks to the charge.

A more interesting case is a charge following a curved trajectory as it could be caused by an external magnetic field. Assuming the electric field to have the same shape as at t_1 leads to a problem: If the particle moves along a circle with radius r at a speed v, the electric field would have to travel at an angular speed $\omega = v/r$ to keep its prior oblate shape. But this would mean that the electric field outside of $r \leq c/\omega$ travels faster than light to keep up

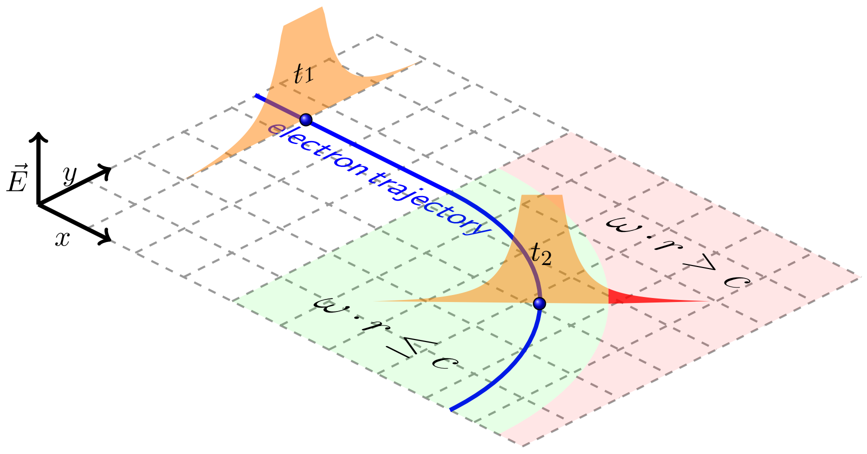


Figure 2.1: Illustrating the origin of radiation from an accelerated charge: At time t_1 a single electron moves at a constant speed in x-direction. The perpendicular field is shown. It does not differ from the static case (Coulomb field). Following that, the electron starts moving along a circular path. At time t_2 the electron's path is already bent. Assuming a field similar to t_1 here would require a field moving faster than the speed of light. This is impossible and the electric field needs to detach from the charge. This results in radiation. (As the fields diverge at the position of the charge, they are cut off.)

with the charge. This, of course, is impossible and causes the electric field to be distorted. The electric field is no longer able to stick to the particle and "radiates" away. If the exchange of information would not be limited by the finite speed of light, radiation effects could not occur. However, this is just an illustration; in the formulae field and charge are still connected as we will see in the next paragraph.
Similar explanations apply to magnetic fields. A moving charge causes a current which induces a magnetic field around its path as described by Ampère's circuital law. The magnetic field outside the $r \cdot \omega \leq c$ zone will apparently be distorted.

2.1.1 Maxwell's Equations and the generation of electromagnetic radiation

The fundamental achievement of Maxwell's work on electrodynamics was the synthesis of optics with electromagnetism, which were two completely separate topics prior to his work. Based on his theory, Helmholtz and Hertz could show that electric and magnetic fields can form waves behaving like light. More importantly, they also showed that one could generate such waves by accelerating electric charges.
There is a variety of ways to solve Maxwell's equations for the most general

case of charges moving at relativistic speeds. The mathematically more elegant solutions use the four-vector notation (see for example [6]) and derive the solution quickly but without much insight into the physics behind it. The more lengthy approaches (for example found in [7] and [8]) do not use relativistic notations but thereby clarify the physical origin of the solutions's mathematical structure.
This section will outline how electromagnetic radiation is generated by electric charges and concentrate on the relationship between the motion of the particles and the resulting radiation. It follows the basic concept presented in [7]. Not all formulae will be derived in detail, this can be found in the works mentioned above, but steps that will give physical insight to this problem will be presented.
When investigating Maxwell's equations

$$\vec{\nabla} \cdot \vec{E} = \frac{\rho}{\varepsilon_0} \tag{2.2}$$

$$\vec{\nabla} \cdot \vec{B} = 0 \tag{2.3}$$

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{2.4}$$

$$\vec{\nabla} \times \vec{B} = \frac{1}{c^2 \varepsilon_0} \vec{j} + \frac{1}{c^2} \frac{\partial \vec{E}}{\partial t} \tag{2.5}$$

in vacuum with charge density ρ and current density $\vec{j}$ equalling 0, combining the curl of Eq. 2.4 with Eq. 2.5 results in a differential equation describing a wave equation for the electric field $\vec{E}$.

$$\vec{\nabla}^2 \vec{E} - \frac{1}{c^2} \frac{\partial^2 \vec{E}}{\partial t^2} = 0 \tag{2.6}$$

A similar wave equation can be derived for the magnetic field $\vec{B}$. These equations describe the well known electromagnetic waves explaining everything from radio waves to optics and X-ray scattering.
However, when trying to find a solution to these fields for any non-vacuum case, one encounters a system of coupled partial differential equations. A way to solve these is to introduce potentials describing the electric and magnetic fields. Because the divergence of the magnetic field needs to be zero everywhere (Eq. 2.3), it turns out to be useful to describe the magnetic field $\vec{B}$ as a curl of a differentiable vector field $\vec{A}$, the vector potential. This can be done since $\vec{\nabla} \cdot (\vec{\nabla} \times \vec{\chi})$ is zero for any differentiable vector field $\vec{\chi}$.

$$\vec{B} = \vec{\nabla} \times \vec{A} \tag{2.7}$$

This definition results in a $\vec{B}$ field satisfying Eq. 2.3 automatically. Applying it to Eq. 2.4 leads to

$$\vec{\nabla} \times \left(\vec{E} + \frac{\partial \vec{A}}{\partial t} \right) = 0 \quad . \tag{2.8}$$

Similarly, one usually defines a scalar potential Φ as

$$\vec{E} + \frac{\partial \vec{A}}{\partial t} = -\vec{\nabla} \cdot \Phi \quad . \tag{2.9}$$

This allows to calculate the electric field using both potentials by

$$\vec{E} = -\vec{\nabla}\Phi - \frac{\partial \vec{A}}{\partial t} \quad . \tag{2.10}$$

Having a second look at the vector potential, the choice of $\vec{A}$ is arbitrary. A transformation $\vec{A'} = \vec{A} + \vec{\nabla}\alpha$ would not change the magnetic field $\vec{B}$ because the curl of a divergence equals zero. However, applying this change requires to adjust the scalar potential as well. Otherwise the electric field would change. Using $\Phi' = \Phi - \frac{\partial \alpha}{\partial t}$ instead of Φ compensates for this. This optional transformation is called a *gauge transform*. Having a gauge freedom in the definition of the potentials allows to simplify the coupled system of partial differential equations. By substituting the definitions of the potentials in Eq. 2.5 and simplifying the formula, one obtains:

$$-c^2\vec{\nabla}^2\vec{A} + c^2\vec{\nabla}\left(\vec{\nabla} \cdot \vec{A}\right) + \frac{\partial}{\partial t}\vec{\nabla}\Phi + \frac{\partial^2 \vec{A}}{\partial t^2} = \frac{\vec{j}}{\varepsilon_0} \quad . \tag{2.11}$$

This does not look like a simplification of the initial set of equations. However, by using the gauge freedom and setting $\vec{\nabla} \cdot A = \vec{\nabla}^2\alpha = -\frac{1}{c^2}\frac{\partial \Phi}{\partial t}$, the following set of equations is derived from Eq. 2.2 and Eq. 2.5:

$$\vec{\nabla}^2\Phi - \frac{1}{c^2}\frac{\partial^2 \Phi}{\partial t^2} = -\frac{\rho}{\varepsilon_0} \tag{2.12}$$

$$\vec{\nabla}^2\vec{A} - \frac{1}{c^2}\frac{\partial^2 \vec{A}}{\partial t^2} = -\mu_0\vec{j} \quad . \tag{2.13}$$

These four independent partial differential equations, three from the vector potential and one from the scalar potential, are separated and have a similar form. Finding a solution to their general form

$$\vec{\nabla}^2\psi - \frac{1}{c^2}\frac{\partial^2 \psi}{\partial t^2} = -s \tag{2.14}$$

with a non-zero source density s allows to solve Maxwell's equation for any charge density ρ and current density $\vec{j}$. There exist several well-known analytical solutions of the wave equation for $s = 0$: plane waves, cylindrical waves and spherical waves. The latter are solutions of Eq. 2.14 everywhere except at their centers. The solution for a spherical wave propagating outwards can be written as:

$$\psi = \frac{1}{r} f(r - ct) \quad , \tag{2.15}$$

with f being a smooth scalar field, r being the distance from the origin of the spherical wave and t being the time. This solution diverges for $r \to 0$. However, by analyzing how fast both second derivatives in the partial differential equation (Eq. 2.14) diverge, one finds:

$$\frac{1}{c^2}\frac{\partial^2 \psi}{\partial t^2} \sim \frac{1}{c^2}\frac{1}{r}\frac{\partial^2 f(r-ct)}{\partial t^2} \tag{2.16}$$

$$\frac{\partial^2 \psi}{\partial r^2} \sim \frac{f(r-ct)}{r^3} + \mathcal{O}\left(\frac{1}{r^2}\right) \tag{2.17}$$

That means that the spatial derivative diverges much faster than the temporal derivative. In the limit $r \to 0$, Eq. 2.14 takes the form:

$$\vec{\nabla}^2 \psi = -s \quad . \tag{2.18}$$

From the derivation of Coulomb's law (Eq. 2.1) from Poisson's equation (Eq. 2.2), it is known that a small charge ($Q/\varepsilon_0 = S$) at $r = 0$ results in an equation equivalent to Eq. 2.18. By following these principles of electrostatics and assuming a fairly localized charge distribution, one obtains the relation between f and the source density s to describe the general field ψ (Eq. 2.15):

$$f = \frac{\int s \,\mathrm{d}V}{4\pi} = \frac{S}{4\pi} \quad . \tag{2.19}$$

Here, S is the total charge equal to the integration of the charge density over the volume V. For $r \to 0$, f also satisfies the full differential equation 2.14. For field values farther away from the source, the only effect caused by the neglected time derivative is a retardation [7]. The source distribution needs to be considered at a past time taking into account that the information about the source's position $\vec{r}_s$ can only propagate with the speed of light c. This former time is called *retarded time* t_{ret} and fulfills the condition $|\vec{r}_s(t_{\mathrm{ret}}) - \vec{r}_f| = c \cdot (t_f - t_{\mathrm{ret}})$ for the field values at $\vec{r}_f$ and time t_f. Therefore, a general form with a time depended source $S(t) = \int s(t)\,\mathrm{d}V$ has the solution:

$$\psi = \frac{1}{4\pi}\frac{S(t - r/c)}{r} \quad . \tag{2.20}$$

So far, no assumption on the potentials was made and it is easy to extend the small charge to a charge density in an arbitrary region. For all cases, a single integration over the volume encompassing all charges at the correct retarded time is required. This leads to the general solution of Maxwells's equations, with $r_{12} = |\vec{r}_1 - \vec{r}_2|$:

$$\Phi(\vec{r}_1, t) = \frac{1}{4\pi\varepsilon_0} \int \frac{\rho(r_2, t - r_{12}/c)}{r_{12}} \,\mathrm{d}\,V_2 \tag{2.21}$$

$$\vec{A}(\vec{r}_1, t) = \frac{\mu_0}{4\pi} \int \frac{\vec{j}(r_2, t - r_{12}/c)}{r_{12}} \,\mathrm{d}\,V_2 \quad . \tag{2.22}$$

These equations describe the potentials $\vec{A}$ and Φ at an arbitrary position $\vec{r}_1$ and time t. The sole contribution to the potentials at $\vec{r} = \vec{r}_1$ are the charge or current densities at $\vec{r}_2$ if they are non-zero at the retarded time $t - r_{12}/c$. By integrating over the entire volume all contributions in space and time are covered.

2.1.2 Liénard-Wiechert Potentials

Applying Eq. 2.21 and Eq. 2.22 to a point charge causes some problems first investigated independently by Liénard and Wiechert. In the above equations, integrating over a point charge has to be carried out carefully because the resulting changes are easily overlooked and hold an important physical interpretation. The correct evaluation of the limit of a point-like charge distribution is presented in the following.

To simplify the derivation, a small cube comprising a homogeneously distributed charge with an edge length x, moving at a speed of v towards the point $\vec{r}_1$ where one wants to evaluate the potential Φ at time t, is assumed. This point $\vec{r}_1$ is called the point of interest or the observation point. The situation is illustrated in Fig. 2.2. In this picture, a blue dot represents the observation point $\vec{r}_1$. Integration over the charge distribution at the retarded time is required when applying Eq. 2.21. To illustrate this integration, one can discretize it and split up the volume surrounding the observation point $\vec{r}_1$ in several spherical shells, each with a thickness $\Delta t'$ in retarded time. The surface of each shell represents an area of equal retarded time. A shell farther away from the observation point represents a prior time compared to the inner shells. If two emitters on the same shell's surface emit an electromagnetic wave at the same time, these waves will arrive simultaneously at the observation point $\vec{r}_1$. As an approximation for further derivations, the volume between two shell surfaces will be attributed to the retarded time associated with the outer of the two shell surfaces. This allows to simplify the Riemann integral over the volume (Eq. 2.21 and Eq. 2.22) to a sum over each shell's volume V_i.

$$\Phi(\vec{r}_1, t) = \frac{1}{4\pi\varepsilon_0} \sum_{i=1}^{N} \frac{\int_{V_i} \rho(t - r'_i/c, \vec{r}) \,\mathrm{d}\,V}{r'_i} = \frac{1}{4\pi\varepsilon_0} \sum_{i=1}^{N} \frac{Q_i}{r'_i} \,. \tag{2.23}$$

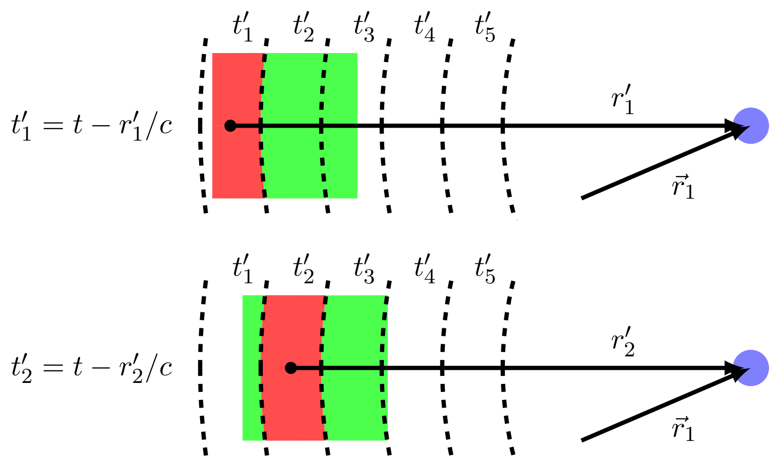


Figure 2.2: Illustrating the correct integration over retarded time: Both pictures show the contribution of a small cubic charge drawn in green or red to the potential Φ at the time t and at the location $\vec{r}_1$ marked by a blue dot. The first picture shows the charge density at t'_1, the second at t'_2. The charge is moving directly towards the observation point ($\vec{r}_1$, blue dot). The charge density at t'_1 inside the shell with the distance r'_1 around the observation point (blue dot) is drawn in red and will contribute to the potentials $\Phi(\vec{r}_1, t)$ at time t. The contributing non-zero charge density is drawn in red while the rest of the non-zero charge density is drawn in green. For the charge density at t'_2, only non-zero values inside the shell with radius r'_2 contribute to the potential of interest and are again drawn in red.

The sum depends only on the total charge $Q_i = \int_{V_i} \rho(t - r'_i/c, \vec{r})\,\mathrm{d}V$ comprised in each shell's volume V_i and the radius r_i of each shell.

In the top of Fig. 2.2, the outermost shell comprising a non-zero charge at its retarded time t'_1 is illustrated. Its contributing charge Q_1 can be calculated by integrating the charge density $\rho(t'_1, \vec{r})$ at the time t'_1 over the entire volume of the shell. Then, one would add the contribution of t'_2, comprised in the next inner shell (illustrated at the bottom of Fig. 2.2). This goes on until the innermost shell is reached. If one compares two consecutive time steps, one sees that the charge has moved and part of the charge density counted at e. g. t'_1 needs to be recounted at t'_2.
By assuming that the charge density is very localized compared to the extent of the shells, any effects by the spherical form of the surfaces can be ignored and a planar integration volume can be assumed. It is possible to approximate the radius of each shell r'_i by a constant radius r' if assuming that the charge is located far away from the observation point $\vec{r}_1$ compared to its extent. These approximations allow to calculate the scalar potential

(Eq. 2.23),

$$\Phi(\vec{r}_1, t) = \frac{1}{4\pi\varepsilon_0} N \cdot \frac{\rho \cdot x^2}{r'} \underbrace{\Delta t' \cdot c}_{\Delta x'} = \frac{q}{4\pi\varepsilon_0 r'} \frac{N \Delta x'}{x} = \frac{q}{4\pi\varepsilon_0 r'} \frac{x'}{x} \quad , \tag{2.24}$$

with x the width of the cube charge and N the number of discrete shells that contribute to the scalar potential Φ. The length x' is the total distance one needs to integrate over to derive the scalar potential from the charge distribution and can be calculated by:

$$x' = x + v \cdot N \cdot \Delta t' = x + \frac{v}{c} x' \tag{2.25}$$

$$x' = \frac{x}{1 - \beta} \quad . \tag{2.26}$$

All the assumptions made above remain valid in the limit of a point charge. If a charge does not move directly towards $\vec{r}_1$, the projection of its motion vector towards the point at which one wants to compute the potential needs to bee considered, $\beta_r = \vec{\beta} \cdot \vec{r}_{12} / |\vec{r}_{12}| = \vec{\beta} \cdot \vec{n}$, with $\vec{n} = (\vec{r}_2 - \vec{r}_1) / |\vec{r}_2 - \vec{r}_1|$ being a unit vector pointing from the charge towards this point. Now the solutions to Maxwells's equations (Eq. 2.21 and Eq. 2.22) can be rewritten for a point charge. These are the well known Liénard-Wiechert potentials:

$$\Phi(\vec{r}_1, t) = \frac{1}{4\pi\varepsilon_0} \left[\frac{q}{(1 - \vec{\beta} \cdot \vec{n}) |\vec{r}_{12}|} \right]_{\text{ret}} \tag{2.27}$$

$$\vec{A}(\vec{r}_1, t) = \frac{\mu_0}{4\pi} \left[\frac{q \cdot \vec{\beta}}{(1 - \vec{\beta} \cdot \vec{n}) |\vec{r}_{12}|} \right]_{\text{ret}} \tag{2.28}$$

The brackets with the index "ret" mean that the inner part has to be evaluated at the retarded time $t_{\text{ret}} = t - |\vec{r}_{12}(t_{\text{ret}})|/c$. This evaluation of the inner part is the only effect caused by the limited speed of light. An approaching charge causes the potential to appear stronger by a factor $\frac{1}{1-\beta_r}$. On the other hand, a charge moving away seems to be reduced by a factor $\frac{1}{1+\beta_r}$. These effects are negligible for any motion at velocities far smaller than the speed of light and the known static solutions are reproduced in this limit.

2.1.3 Results of the Liénard-Wiechert Potentials

With the Liénard-Wiechert potentials, the electric and magnetic fields radiated by an arbitrarily moving electric charge can be calculated by applying the conversion Eq. 2.7 and Eq. 2.9 to the scalar and vector potential. This calculation is cumbersome and does not lead to any insight considering the physics. A detailed derivation can be found in [8]. Here, only the results will be presented.

$$\vec{E}\,(\vec{r},t) = \frac{q}{4\pi\varepsilon_0}\left\{\left[\frac{(\vec{n}-\vec{\beta})}{R^2\gamma^2(1-\vec{\beta}\cdot\vec{n})^3}\right]+\left[\frac{\vec{n}\times\left[(\vec{n}-\vec{\beta})\times\dot{\vec{\beta}}\right]}{cR(1-\vec{\beta}\cdot\vec{n})^3}\right]\right\}_{\text{ret}} \tag{2.29}$$

$$\vec{B}(\vec{r},t) = \frac{1}{c}\cdot\left[\vec{n}_{\text{ret}}\times\vec{E}(\vec{r},t)\right] \tag{2.30}$$

The above equations allow to calculate the electromagnetic fields everywhere and at every time for a know particle trajectory. In these equations R is the distance between the point of interest $\vec{r}$ and the position of the charge and needs to be evaluated at the retarded time. The particle's charge is q, while β is its velocity normalized to the speed of light c. The electric field consists of two parts. One decreases inversely with the square of the distance $\sim R^{-2}$ and is called velocity or near field $\vec{E}^{\text{vel}}$, the other one decreases only inversely with the distance and is usually referred to as far or radiation field $\vec{E}^{\text{rad}}$.

$$\vec{E}^{\text{vel}} = \frac{q}{4\pi\varepsilon_0}\cdot\left(\frac{(\vec{n}-\vec{\beta})}{R^2\gamma^2(1-\vec{\beta}\cdot\vec{n})^3}\right)_{\text{ret}} \tag{2.31}$$

$$\vec{E}^{\text{rad}} = \frac{q}{4\pi\varepsilon_0 c}\cdot\left(\frac{\vec{n}\times\left[(\vec{n}-\vec{\beta})\times\dot{\vec{\beta}}\right]}{R(1-\vec{\beta}\cdot\vec{n})^3}\right)_{\text{ret}} \tag{2.32}$$

The far field component will not disappear in the infinite, causing the charged particle to lose energy by electromagnetic radiation. In practice, the effects of this energy loss can often be neglected. A more detailed discussion will be postponed until chapter 7.

Using the equations 2.29 and 2.30, the first example (Fig. 2.1) can now be revisited rigorously. The results are shown in Fig 2.3. The field during the constant motion of the electron was described correctly while the field of the electron moving along the bent path differs as predicted. Part of the electric field is no longer able to follow the electron and stays behind.
The field changes over time need to be analyzed to better understand the emitted electromagnetic fields. This can be done by calculating the spectrally resolved electromagnetic intensity far away from the particle. Such spectra are easily understandable and can be compared to experimental results since detectors are usually placed at a much larger distance compared to the spatial extent of the emission region.
In order to calculate the spectrally resolved radiated intensity $\frac{\mathrm{d}\,I^2}{\mathrm{d}\Omega\,\mathrm{d}\omega}$, one can start off from the Poynting vector $\vec{S}$. It describes the instantaneous energy flux at a certain point in space. Its direction shows the direction of the flux,

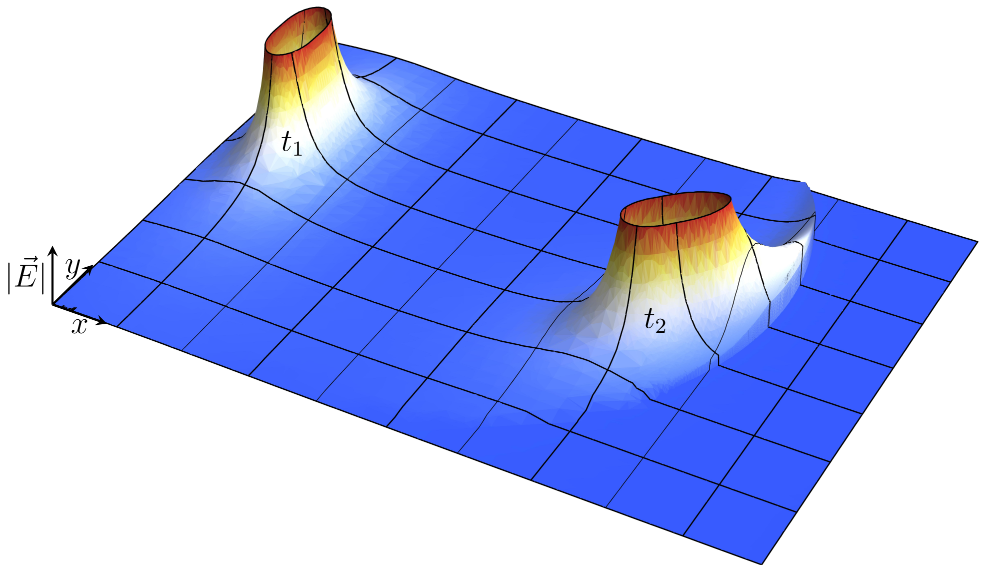


Figure 2.3: This graph illustrates the electric field strength $|\vec{E}|$ occurring in the example discussed in the introduction to this chapter. At t_1 the electron moves with a constant velocity close to the speed of light. The electric field has an oblate shape. At t_2, the electron moves with the same velocity on a circular path and the oblate shape of the electric field is distorted and a wavefront is generated.

its magnitude is the strength of the energy flux [6].

$$\vec{S} = \frac{1}{\mu_0}\vec{E} \times \vec{B} = \frac{1}{c\mu_0}\vec{E} \times \left(\vec{E} \times \vec{n}\right) = \frac{1}{c\mu_0}E^2\vec{n} \tag{2.33}$$

A large sphere containing all charges close to its center is considered to compute the radiated energy per unit solid angle. The power P per unit solid angle $\mathrm{d}\,\Omega$ on the surface of this sphere is then:

$$\frac{\mathrm{d}\,P}{\mathrm{d}\,\Omega} = \vec{S} \cdot |\vec{r}_2 - \vec{r}_1|^2 = \frac{1}{c\mu_0}|R \cdot \vec{E}|^2 \quad . \tag{2.34}$$

For large distances only the part proportional to R^{-1} in Eq. 2.29, the far field $\vec{E}^{\mathrm{rad}}$, does not vanish. Therefore, the calculation can be executed by only considering the far field. In order to calculate the entire energy radiated in a certain direction, one needs to integrate over all times.

$$\frac{\mathrm{d}\,W}{\mathrm{d}\,\Omega} = \int_{-\infty}^{+\infty} \frac{\mathrm{d}\,P}{\mathrm{d}\,\Omega}\,\mathrm{d}\,t = \int_{-\infty}^{+\infty} \frac{1}{c\mu_0}|R \cdot \vec{E}|^2\,\mathrm{d}\,t \tag{2.35}$$

The Fourier transform of $(c\mu_0)^{-1/2}R\vec{E}$ can be calculated by:

$$\vec{\mathcal{E}}(\omega) = \frac{1}{\sqrt{2\pi c\mu_0}} \int_{-\infty}^{+\infty} R \cdot \vec{E}(t) \mathrm{e}^{+\mathrm{i}\omega t} \,\mathrm{d}\, t \tag{2.36}$$

Rewriting Eq. 2.35 leads to:

$$\frac{\mathrm{d}\, W}{\mathrm{d}\,\Omega} = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \mathrm{d}\, t \int_{-\infty}^{+\infty} \mathrm{d}\,\omega_1 \int_{-\infty}^{+\infty} \mathrm{d}\,\omega_2 \vec{\mathcal{E}}^*(\omega_2)\vec{\mathcal{E}}(\omega_1)\mathrm{e}^{\mathrm{i}(\omega_2-\omega_1)t} \tag{2.37}$$

The time integral is simplified to $\int_{-\infty}^{+\infty} \mathrm{e}^{-\mathrm{i}(\omega_2-\omega_1)t} \,\mathrm{d}\, t = \delta(\omega_2 - \omega_1)$ and thus the entire formula results in

$$\frac{\mathrm{d}\, W}{\mathrm{d}\,\Omega} = \int_{-\infty}^{+\infty} \left|\vec{\mathcal{E}}(\omega)\right|^2 \mathrm{d}\,\omega = \int_0^{\infty} \frac{\mathrm{d}^2\, I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} \,\mathrm{d}\,\omega \quad . \tag{2.38}$$

The last step uses Parseval's theorem (Eq. A.3). It is treated in more detail in the appendix. The last part of the equation is simply a more intuitive way to describe the radiated energy. It is defined by

$$\frac{\mathrm{d}^2\, I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} = \left|\vec{\mathcal{E}}(\omega)\right|^2 + \left|\vec{\mathcal{E}}(-\omega)\right|^2 = 2 \cdot \left|\vec{\mathcal{E}}(\omega)\right|^2 . \tag{2.39}$$

Considering only positive frequencies is possible because $\vec{\mathcal{E}}(t)$ is real and therefore $\vec{\mathcal{E}}(\omega) = \vec{\mathcal{E}}^*(-\omega)$. To calculate $\vec{\mathcal{E}}(\omega)$, one needs to apply Eq. 2.36 to the $\sim 1/R$ part of Eq. 2.29, leading to

$$\vec{\mathcal{E}}(\omega) \quad = \quad \frac{q}{4\pi\varepsilon\sqrt{c\mu_0 2\pi}} \int_{-\infty}^{+\infty} \mathrm{e}^{+\mathrm{i}\omega t} \left[\frac{\vec{n} \times \left(\left[\vec{n} - \vec{\beta}\right] \times \dot{\vec{\beta}} \right)}{\left(1 - \vec{\beta}\cdot\vec{n}\right)^3} \right]_{\mathrm{ret}} \mathrm{d}\, t\, . \tag{2.40}$$

The central term in Eq. 2.40 has to be evaluated at the correct retarded time $t' = t - R(t')/c$. By substituting t by t' and integrating over the retarded time, one arrives at

$$\vec{\mathcal{E}}(\omega) \quad = \quad \frac{q}{4\pi\varepsilon\sqrt{c\mu_0 2\pi}} \int_{-\infty}^{+\infty} \mathrm{e}^{+\mathrm{i}\omega(t'+R(t')/c)} \frac{\vec{n} \times \left(\left[\vec{n} - \vec{\beta}\right] \times \dot{\vec{\beta}} \right)}{\left(1 - \vec{\beta}\cdot\vec{n}\right)^2} \,\mathrm{d}\, t' \tag{2.41}$$

Now, one can approximate R by assuming that the entire particle motion at any time is close to a point considered to be the origin of the coordinate system for simplicity (see figure 2.4). This is always possible if the entire particle trajectory can be contained in a sphere, since when increasing the radius of the sphere all particle can be considered close the center of the sphere. Using this, one can describe the distances between a point of interest at $\vec{x}$ and the particle at retarded time $\vec{r}(t')$ as $\vec{R}(t') = -\vec{r}(t') + \vec{x}$, with

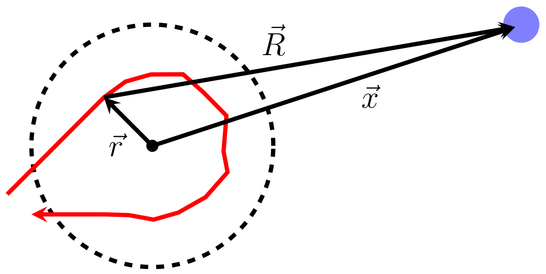


Figure 2.4: Simplification of the retarded distance: This illustrates a common simplification applied in the far field approximation. When assuming that acceleration just occurs around the origin of the coordinate system (black dot) and that the observation point (blue dot) is far away $|\vec{r}| \ll |\vec{x}|$, the distance $R = |\vec{R}|$ between the particle and the observation point can be approximated by $R \approx |\vec{x}| - \vec{n} \cdot \vec{r}(t')$.

$\vec{n} = \vec{R}/R \approx \vec{x}/x$ for $|x| \gg |r|$.

$$R(t') = |\vec{R}(t')| = |\vec{x} - \vec{r}(t')| \approx x - \vec{n} \cdot \vec{r}(t') \tag{2.42}$$

By ignoring the constant offset in time x/c, which will only result in a constant phase shift of the result (see appendix A), one can simplify $\vec{\mathcal{E}}$ to:

$$\vec{\mathcal{E}}(\omega) = \frac{q}{4\pi\varepsilon\sqrt{c\mu_0 2\pi}} \int_{-\infty}^{+\infty} \mathrm{e}^{\mathrm{i}\omega(t' - \vec{n} \cdot \vec{r}(t')/c)} \frac{\vec{n} \times \left(\left[\vec{n} - \vec{\beta} \right] \times \dot{\vec{\beta}} \right)}{\left(1 - \vec{\beta} \cdot \vec{n} \right)^2} \mathrm{d}\, t' \tag{2.43}$$

Inserting this approximation in the definition of Eq. 2.38 results in:

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} = \frac{q^2}{16\pi^2\varepsilon_0 c} \left| \int_{-\infty}^{+\infty} \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta} \right) \times \dot{\vec{\beta}} \right]}{\left(1 - \vec{\beta} \cdot \vec{n} \right)^2} \cdot \mathrm{e}^{\mathrm{i}\omega(t - \vec{n} \cdot \vec{r}(t)/c)} \,\mathrm{d}\, t \right|^2 \tag{2.44}$$

This formula allows to calculate the spectra seen at any point far away from the particle. It consists of a real vector part

$$\vec{A} = \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta} \right) \times \dot{\vec{\beta}} \right]}{\left(1 - \vec{\beta} \cdot \vec{n} \right)^2} \in \mathbb{R}^3 \tag{2.45}$$

and a complex phase determined by a simplified retarded time

$$t_{\mathrm{ret}} = t - \vec{n} \cdot \vec{r}(t)/c \quad . \tag{2.46}$$

It is important to note that the integrand only contributes to the spectra if the particle is accelerated, $\dot{\vec{\beta}} \neq 0$.

2.2 Introductory Example: Nonlinear Thomson scattering

Thomson scattering is the elastic scattering of electromagnetic radiation by free moving charges as for example electrons in a plasma or outer electrons of an atom. It occurs when the energy of the scattered photons is low compared to the energy associated with the rest mass of the electron and the energy transfer between the photon and the electron is negligible. If a significant amount of energy is transferred, one speaks about Compton scattering.
From a quantum field theoretical viewpoint, Thomson scattering is the absorption of a photon and the emittance of a photon of the same energy (Fig 2.5 (a)). When the intensity of the light striking the electron increases, the probability of interacting with more than one photon rises. This allows the electron to absorb several photons before emitting a single photon of higher energy (Fig. 2.5 (b)). A correct quantum-mechanical description of these multi photon interactions requires a strong-field QED approach [9].

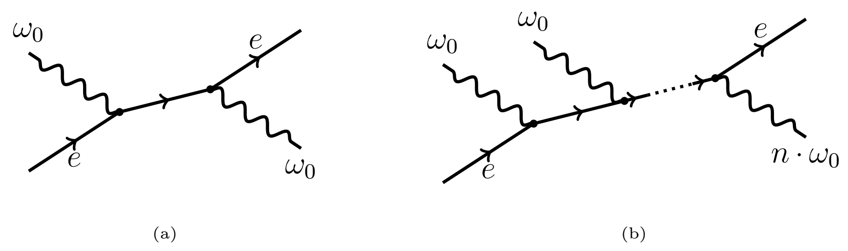


Figure 2.5: Feynman graph of Thomson scattering: (a) tree-level Thomson scattering, absorbing a photon and emitting a photon of same energy. (b) higher order diagram, absorbing n photons of same energy ω_0 and emitting a single photon of energy $n \cdot \omega_0$

From a classical viewpoint, linear Thomson scattering is just an electromagnetic wave, for example light from a laser, interacting with a charged particle and causing it to oscillate. This oscillation again causes the emission of radiation at the oscillation frequency which is equal to the frequency of the incident radiation.
The results of both the classical and the quantum theory describing electron-photon interaction agree for low photon energies and electron energies below $\gamma < 10^4$ [10]. For multi-photon interactions at low photon energies it is convenient to describe the scattered light classically, to avoid the complex strong-field QED calculations [11]. Nonetheless, if the momentum transfer

between photons and electrons can no longer be neglected or if the multi-photon interaction becomes dominant, a classical approach fails and a field theoretical treatment is required [10, 12]. Since the focus of the simulations presented here is on laser plasma interactions, a classical treatment is currently adequate.
In the classical model, one speaks of nonlinear Thomson scattering if the intensity of the incident radiation is becoming large enough so that the charged particles reach relativistic energies during the interaction. In this regime, the particle's mass can no longer be considered constant. This and the force on the electron due to the motion in the (no longer negligible) magnetic field of the electromagnetic wave change the formerly simple differential equations describing the particle's motion during this interaction to a coupled system of differential equations [4]. The particle's motion is no longer sinusoidal and the emitted radiation now also carries higher harmonics of the incident electromagnetic radiation. It is important to note that this is not considered to be Compton scattering because the anharmonic electron motion is not caused by the high photon energy but by the large number of low energy photons interacting with the charged particle.
A laser is considered to be intense enough to cause nonlinear effects for electrons if its dimensionless field strength parameter $a_0 = \frac{eE_L}{m_e c \omega_0}$ is unity or larger, with E_L the maximum electric field of the laser and ω_0 the frequency of the laser. At such intensities, the no longer negligible B-field causes the electrons to oscillate in longitudinal direction at twice the laser frequency [13]. This causes a particle to have a figure-eight kind of motion in its average rest frame. This is illustrated by Fig. 2.6. An electron initially at rest was assumed. Due to an average force caused by the electron's oscillation along a non-zero gradient of the electric fields envelope, called *ponderomotive force*, the electron gets pushed forward into the direction of laser propagation. Fig. 2.6 (a) shows the electron's trajectory in the lab frame assuming constant intensities of $a_0 = 0.5$ (red) and $a_0 = 1.5$ (blue). It is important to note, that the x-axis is normalized to a_0^2, which illustrates that higher laser intensities cause faster drift velocities. Fig. 2.6 (b) shows the electron trajectory in the electron's average rest frame. A detailed description of the electron's motion can be found in [4].
The radiation caused by this relativistic Thomson scattering was investigated by Esarey et al. [14]. The paper presents two cases where analytical solutions of the scattered radiation can be found. One assumes a single electron with high energy moving towards a plane electromagnetic wave as can be found in so called optical undulators (see chapter 2.3). The other case assumes the electron to be, on average, at rest. This is an idealized case of a dense plasma. Both cases assume a constant field strength and ignore effects caused by ponderomotive forces.
Here, only the resulting spectra are presented and their typical structure is discussed.

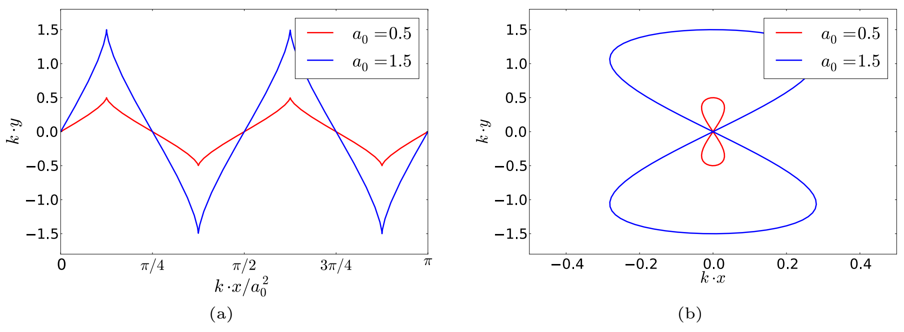


Figure 2.6: Electron trajectory for nonlinear Thomson scattering: Picture (a) illustrates the electron's trajectory for a plane laser wave of constant intensity $a_0 = 0.5$ (red) and $a_0 = 1.5$ (blue) in the lab frame while plot (b) shows the same case in the electron's average rest frame. x is the location along the laser propagation while y is the location in the direction of the laser's polarization. $k = \frac{2\pi}{\lambda_0}$ is the wave number of the laser with wavelength λ_0.

The emitted radiation of an electron moving at a relativistic velocity towards a laser is shown for several laser strengths in Fig. 2.7. In all cases, an electron with a Lorentz factor of $\gamma = 5$ is assumed which travels through $N = 7$ laser periods. The plotted angle θ describes the angle between the point of observation and the direction of the laser propagation $\vec{k}$, and is confined to the polarization plane. With higher laser intensities, higher harmonics become more dominant. As can be seen in Fig. 2.7, even harmonics do not contribute on the laser propagation axis $\theta = 0$ [14]. Because of the electron's relativistic velocity, the emitted frequencies are Doppler shifted compared to the initial laser frequency. This relativistic effect strongly depends on the observer's viewing angle θ and causes the typical u-shaped curves of the harmonic intensities. It is also the reason for the confinement of the radiation to a narrow cone around the axis of propagation of the electron. The cone's opening angle scales with $\sim \frac{1}{\gamma}$ and leads to a search-light-like behavior of radiation emitted by electrons with relativistic energies. The harmonic's peak frequencies become smaller for higher laser intensities due to the *photon drag*, a reduction of the Doppler shift of the emitted radiation caused by the electron's strong transversal motion which reduces the electron's velocity in longitudinal direction.

The radiation in the case of electrons being, on average, at rest is shown in Fig. 2.8. Here, the photon drag plays no role. Again, higher harmonics are stronger for higher laser power and even harmonics do not contribute on axis. All spectral peaks are located at multiples of the incident laser frequency, and even if substructures are visible for different directions, the peak of the n^{th} harmonic stays at $n \cdot \omega_0$.

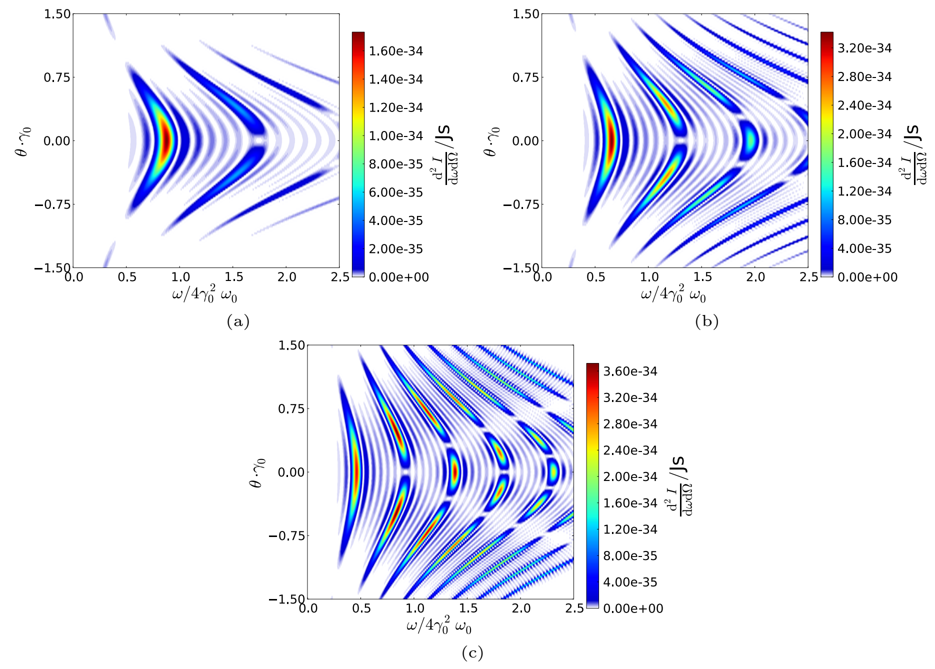
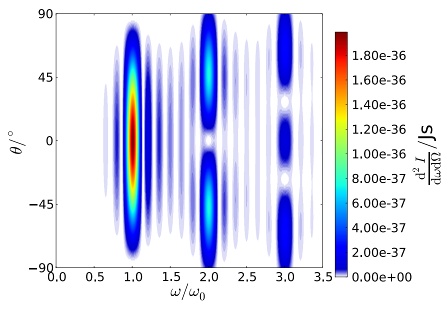


Figure 2.7: Nonlinear Thomson scattering: Analytical solution of the radiation emitted by an electron $\gamma = 5$ moving towards a laser. Energy deposition per unit frequency and unit solid angle for different laser strength are shown: (a) laser with $a_0 = 0.5$, (b) laser with $a_0 = 1.0$, (c) laser with $a_0 = 1.5$. θ is the angle in the polarization plane of the laser, with $\theta = 0$ the opposite direction of the laser.

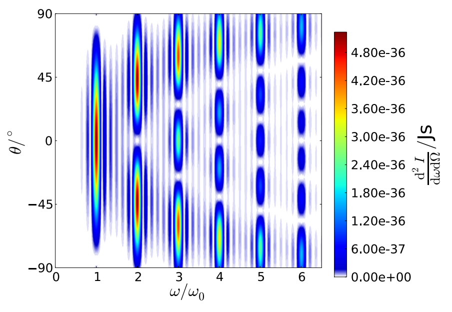
All spectra presented are calculated from the analytical solution presented by Esarey et al. [14] and are not based on simulated data. Therefore nonlinear Thomson scattering is an ideal test case for any radiation code targeting at laser-particle interactions.

2.3 Undulator radiation

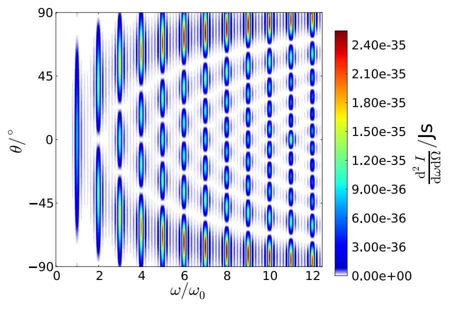
Undulators and wigglers are arrays of paired permanent magnets causing electrons passing through them to undergo a transverse oscillation which causes the electrons to emit radiation. For undulators the amplitude of the transverse motion is small. These devices can for example be inserted in electron storage rings and can provide a source of bright and mono-energetic electromagnetic radiation for a multitude of research applications like solid state physics and biology [15, 16].
A detailed analysis of undulators and wigglers can be found in [17]. This section will just briefly go over the particle's kinematics and concentrate on the resulting radiation. An illustration of an undulator can be found



(a)



(b)



(c)

Figure 2.8: Nonlinear Thomson scattering with electron of vanishing average velocity: Energy deposition per unit frequency and unit solid angle for various laser strengths are presented: (a) laser with $a_0 = 0.5$, (b) laser with $a_0 = 1.0$, (c) laser with $a_0 = 2.0$ emitted by a single electron with average speed of zero $\left\langle \vec{\beta}(t) \right\rangle = 0$. θ is the angle in the polarization plane of the laser, with $\theta = 0$ the opposite direction of the laser.

in Fig. 2.9. By switching the orientation of the magnets surrounding the particles' paths, illustrated in red and blue, after a certain fixed distance λ_u, the electrons follow an approximately sinusoidal trajectory, drawn in green. The electron enters the undulator with a velocity of β in z direction. Because of the magnetic field in y direction, the electrons are bent in x direction. Since particles cannot gain energy in magnetic fields their absolute velocities stay constant, consequently their velocities in z direction are reduced. This effect is equivalent to the *photon drag* in laser-electron interactions [18].
The motion of an electron inside an undulator can be described by:

$$x = \frac{Kc}{\gamma W} \cdot \sin(Wt) \tag{2.47}$$

$$y = 0 \tag{2.48}$$

$$z = c\beta_s t - \frac{K^2 c}{8\gamma^2 W} \cdot \sin(2Wt) \tag{2.49}$$

Here, β_s is the average velocity in the original direction of flight:

$$\beta_s = 1 - \frac{1}{2\gamma^2} - \frac{K^2}{4\gamma^2} \tag{2.50}$$

The values $K = \frac{eB\lambda_u}{2\pi m_e c}$ and $W = \frac{2\pi c \beta_s}{\lambda_u}$ describe the undulator parameter giving the amplitude of the electrons' oscillation and the oscillation frequency, respectively. The electron mass is m_e and the relativistic energy factor of the electron is γ. These formulae are only valid for particles moving close to the speed of light, since the approximation $\beta \lesssim 1$ was used to derive them. The electromagnetic radiation emitted is characterized by a single spectral line at:

$$\omega_{\text{peak}} \approx \frac{2\gamma^2}{1+\gamma^2\theta^2}\left(\frac{2\pi c}{\lambda_u}\right) \tag{2.51}$$

In deriving this formula a β close to 1 and a small angle θ were assumed [6]. If the electrons pass N periods of alternating magnets in the undulator the emitted energy is N^2 times as strong as if only a single undulator period would have been passed. This can be derived easily from Eq. 2.44 and the linearity of Fourier transform (see chapter A.1). Furthermore, the relative

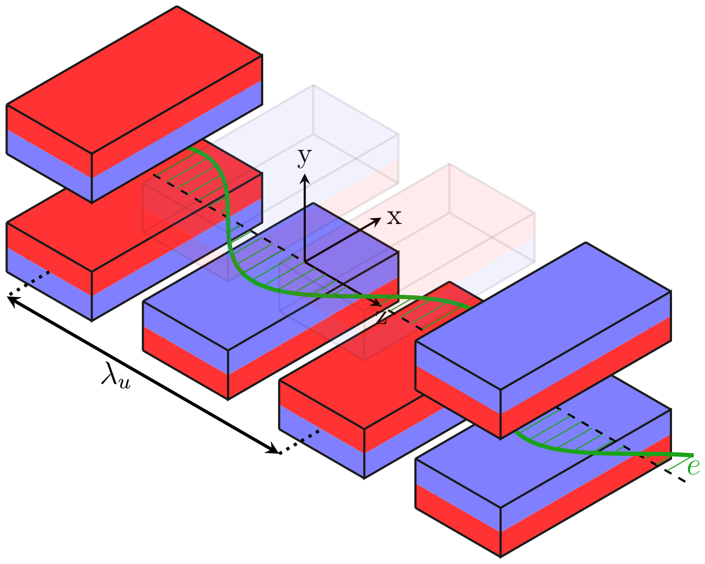
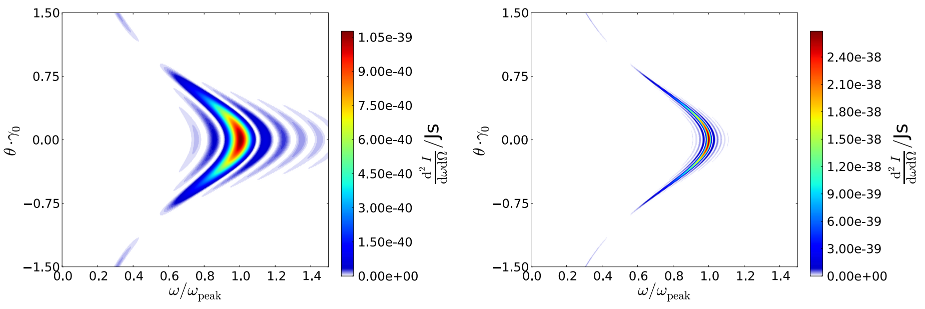


Figure 2.9: Schematic sketch of an undulator: An undulator consists of magnets with alternating polarization. This causes the magnetic field to be periodic with a period length of λ_u. The periodic magnetic field causes an electron to undergo a sinusoidal motion (bold green line).

frequency spread decreases with every magnetic period passed (see Fig. 2.10).

$$\frac{\Delta\omega}{\omega} \sim \frac{1}{N} \tag{2.52}$$

A complete description of undulator radiation can be found in the literature [17]. Here, the results for the radiated energy per unit frequency and unit solid angle are shown.



(a) (b)

Figure 2.10: Undulator spectra: Both pictures show an analytically calculated spectrum from a single electron in an undulator. The electron has an energy of $\gamma = 100$. The x-axis shows the frequency ω normalized to the peak frequency ω_{peak} (see Eq. 2.51). The y-axis shows the angle θ between the electron's direction of flight and the observer's position in the polarization plane, normalized by the electrons energy. (a) is a 10 periods λ_u long undulator, while (b) is an undulator of 50 periods length.

The spectra shown in Fig. 2.10 both result from an electron with energy $\gamma = 100$. The first picture exhibits the radiation from an undulator of 10 periods length while the second spectrum shows the radiation from a 50 period long undulator. To calculate these results, the analytical solution presented in [17] could not be used because it is only valid for small angles θ. Instead, the equivalence of undulators and laser particle interactions [18] were combined with the analytical solution for Thomson scattering [14] to calculate the spectra. The equivalence is best for high energy electrons $\gamma \gg 1$. In such a case, the electromagnetic fields of an undulator in the rest frame of an electron are very similar to those of a laser field. The strength parameter of the undulator K is then equivalent to the unitless laser strength parameter a_0. The laser wavelength λ_L, however, relates to the undulator period length λ_u as follows:

$$\lambda_u = \frac{\lambda_L}{1 - \beta_0 \cos\varphi} \tag{2.53}$$

In this equation, the electron's velocity is β_0, while the angle between the laser propagation and the electron propagation is $\varphi = 180°$ for a head-on

collision.
The physics of undulators can be described completely analytically. This makes them perfectly suited to be used as test cases for simulations.

3 | Numerical solutions for electromagnetic emissions

In this chapter, three different programs, developed as part of the Diploma thesis, are presented. The first provides a numerical solution for calculating the electric and magnetic fields of electrons with relativistic energies (Eq. 2.29 and Eq. 2.30) at any point in space and time, $\vec{E}(\vec{r}, t)$ and $\vec{B}(\vec{r}, t)$. It is able to compute fields for different time steps, therefore it is called a *time domain code*. It is especially useful to determine the electromagnetic fields close to the position of the charge as it is required for simulating coherent synchrotron radiation (see chapter 5.1.3).
The other two programs developed are able to predict the spectrally resolved emission intensity $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n}, \omega)$ in arbitrary directions $\vec{n}$ far away from the particle based on Eq. 2.44. These are considered *frequency domain codes*. They are particularly interesting for comparing experimental results with simulations as physical detectors that record this radiation are usually placed at a much larger distance compared to the spatial extent of the emission region (see chapter 5.2.3 to 5.2.4). The first of the frequency domain simulations is designed for running independent of a specific particle simulation, while the second is directly combined with a particle-in-cell simulation to speed up the radiation calculation.
A summary of all codes developed is given in table 3.1.

3.1 Simulating electromagnetic fields in the time domain

The time domain code calculates electric and magnetic fields, $\vec{E}(\vec{r}_{\mathrm{obs}}, t_{\mathrm{obs}})$ and $\vec{B}(\vec{r}_{\mathrm{obs}}, t_{\mathrm{obs}})$, of point-like charges at user-defined positions $\vec{r}_{\mathrm{obs}}$ and times t_{obs}. The program developed just needs to be provided with discretized particle trajectories. It is especially useful in physics simulations where narrow field peaks need to be considered, but where resolving these details with a standard particle-mesh codes [19] would be too time-consuming.

Table 3.1: Programs developed

Programs	**Time domain code**	**CLARA 2.0**	**Radiation Analyzer**
Calculates	Electric field $\boldsymbol{\vec{E}}(\boldsymbol{\vec{r}}_{\text{obs}}, \boldsymbol{t}_{\text{obs}})$ Magnetic field $\boldsymbol{\vec{B}}(\boldsymbol{\vec{r}}_{\text{obs}}, \boldsymbol{t}_{\text{obs}})$	Spectrally resolved emission intensity $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n}, \omega)$	Spectrally resolved emission intensity $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n}, \omega)$
Calculation for	Location $\boldsymbol{\vec{r}}_{\text{obs}}$ Time $\boldsymbol{t}_{\text{obs}}$	Frequency ω Direction $\boldsymbol{\vec{n}}(\boldsymbol{\theta})$	Frequency ω Direction $\boldsymbol{\vec{n}}(\boldsymbol{\theta})$
Based on	Eq. 2.29 and Eq. 2.30	Eq. 2.44	Eq. 2.44
Requires	Trajectories and momenta of particles stored on disk	Trajectories and momenta of particles stored on disk	Particle data within *PIConGPU*
Parallelization	Several cores of a CPU	Several nodes on a compute cluster	Several graphic cards on a GPU cluster
Applications	Coherent synchrotron radiation (chapter 5.1.3)	Beam emittance scans Radiation from small scale plasma simulation (chapter 5.2.3 and 5.2.4)	Radiation from full scale plasma simulation (chapter 5.2.4)

3.1.1 Motivation and basic concept

The electromagnetic fields at arbitrary positions generated by charged particles are of interest when computing particle-particle interactions [19]. Most codes consider these interactions indirectly by investigating the evolution of electromagnetic fields on a grid separate from the charges and then let these fields act on the particles. These particle-mesh methods have several disadvantages: they require knowledge of the electromagnetic fields at the start of the simulation, then they need to simulate the field evolution until the situation one is interested in is reached, and finally they cannot resolve features of the field on scales shorter than the grid size. Additionally, resolving an electromagnetic field in detail for a certain volume of interest requires a huge number of grid points. For example, resolving an electromagnetic wave with a wavelength of $\lambda = 0.1\,\mu\text{m}$ requires a grid spacing of about half the wavelength. For a simulation volume of $V = 1\,\text{cm} \times 1\,\text{cm} \times 1\text{cm}$ this would result in approximately 10^{15} grid points which have to be calculated for every time step. Such an approach for simulating electromagnetic fields would be infeasible considering the time required for such a simulation. Nonetheless, in cases for which one can neglect short-range interactions, this algorithm is convenient and is used in particle-in-cell simulations (see section 3.3.2).
Particle-particle codes on the other hand usually consider the interaction between particles directly by applying instantaneous Coulomb fields to compute the forces of particle interactions. This is an appropriate approximation in the case of particle velocities far below the speed of light, but fails for relativistic particle velocities.
The newly developed program is able to correctly obtain the electric and magnetic fields by relativistically moving charges using Eq. 2.29 and Eq. 2.30. It can compute the electromagnetic fields at any detail without costly simulating the fields beforehand or requiring the fields to be known at a previous time step. However, it shall be emphasized that this code computes electromagnetic fields from known particle trajectories and does not simulate the resulting influence of the fields back onto the particles.

3.1.2 Determine the retarded and advanced time

Due to the limited speed of light, the fields at a certain time are determined by a charge at a previous time, the retarded time. One needs to solve Eq. 3.1 for the retarded time t_{ret} to calculate the electromagnetic field at a position $\vec{r}_{\text{obs}}$ at time t_{obs},

$$t_{\text{obs}} - t_{\text{ret}} = \frac{|\vec{r}_{\text{obs}} - \vec{r}_e(t_{\text{ret}})|}{c} \quad , \tag{3.1}$$

with $\vec{r}_e(t_{\text{ret}})$ denoting the position of the charge at the retarded time. In most cases, this can be solved only numerically.
In order to compute the fields at a position $\vec{r}_{\text{obs}}$ and a time t_{obs}, the particle

position $\vec{r}_e(t_{\text{ret}})$ at the retarded time need to be determined numerically. Therefore, one either needs to keep track of the position of every particle for all relevant time steps or compute all field values for future time steps in advance. The first option is the more flexible one. By knowing the entire discrete trajectory $\vec{r}_e(t)$ prior to t_{obs}, the fields at any arbitrary point $\vec{r}_{\text{obs}}$ can be calculated, even if the position of the point $\vec{r}_{\text{obs}}$ is not known prior to the current time step. This allows to calculate electromagnetic fields of particles at the exact position of other particles. However, running through entire trajectories of N_t time steps to find the valid retarded time t_{ret} and position $\vec{r}_e(t_{\text{ret}})$ needs an algorithm requiring $\sim \lg N_t$ computations for every point of interest $\vec{r}_{\text{obs}}$. The computational cost of this algorithm is proportional to $\lg N_t \times N_{\text{obs}}$ per time step, with N_{obs} the number of positions the user wants to observe the field at. Additionally, keeping entire trajectories requires a large amount of memory for every particle. This is a problem if considering many charges.
On the other hand, an algorithm calculating the field in advance just needs memory to store the field values for the future time steps of interest t_{obs}. In order to determine the advanced time t_{adv} at which a field arrives at a position $\vec{r}_{\text{obs}}$, the following equation needs to be solved

$$t_{\text{adv}} = t + \frac{|\vec{r}_{\text{obs}} - \vec{r}_e(t)|}{c} \quad , \tag{3.2}$$

with t the current time step. This can be done analytically. If t_{obs} equals t_{adv} the field can be computed. For a single particle, this method might not be of advantage, but for several particles, the memory needed by this algorithm stays constant. More importantly, computing fields at equidistantly spaced advanced time steps t_{obs} allows to use an algorithm of constant complexity for every point of interest. With such an algorithm, only about N_{obs} calculations per time step are needed. The disadvantage of this method is that one usually cannot predict a particle's position beforehand and therefore one is not able to calculate the fields at the exact position another charge will be located at at a future time step.
For the program developed the second algorithm was chosen.

3.1.3 Implementation

In order to calculate the electric and magnetic fields for an arbitrary position and time based on Eq. 2.29 and Eq. 2.30, $\vec{E}(\vec{r}_{\text{obs}}, t_{\text{obs}})$ and $\vec{B}(\vec{r}_{\text{obs}}, t_{\text{obs}})$, the program loads the particle trajectories and converts them into suitable units. The values needed by default are time, location, and momentum. The particle-handling algorithm implemented, the *particle handler* for short, can work with a variety of unit systems and physical quantities and convert between them, e. g. converting momentum to speed and vice versa.
The discrete trajectory leads to discrete times at which the field arrives at

a certain point in space. These times will probably not coincide with the user-defined times t_{obs} at which the fields shall be calculated. By using two time steps, “past” and “now”, to calculate the advanced fields,

$$\vec{E}\left(\vec{r}_{\text{obs}}, t_{\text{adv}}\right) = \frac{q}{4\pi\varepsilon_0}\left[\frac{(\vec{n}-\vec{\beta})}{R^2\gamma^2(1-\vec{\beta}\cdot\vec{n})^3} + \frac{\vec{n}\times\left[(\vec{n}-\vec{\beta})\times\dot{\vec{\beta}}\right]}{cR(1-\vec{\beta}\cdot\vec{n})^3}\right] \quad (3.3)$$

$$\vec{B}(\vec{r}_{\text{obs}}, t_{\text{adv}}) = \frac{1}{c}\cdot\left[\vec{n}\times\vec{E}(\vec{r}_{\text{obs}}, t_{\text{adv}})\right] \quad (3.4)$$

$$\vec{R}(t) = \vec{r}_{\text{obs}} - \vec{r}_e(t) \qquad \vec{n} = \frac{\vec{R}}{|\vec{R}|} \quad , \quad (3.5)$$

it is possible to interpolate the field values onto the time steps of interest t_{obs} as illustrated in Fig. 3.1. This allows to calculate the field values for the user-defined observation times t_{obs}, lying between “past” and “now”.

The execution of the program is relatively simple and is shown as a flow chart in Fig. 3.2. First of all, the program loads particle trajectories created by an external program. Then, all points of interest, called *observers*, are set up by

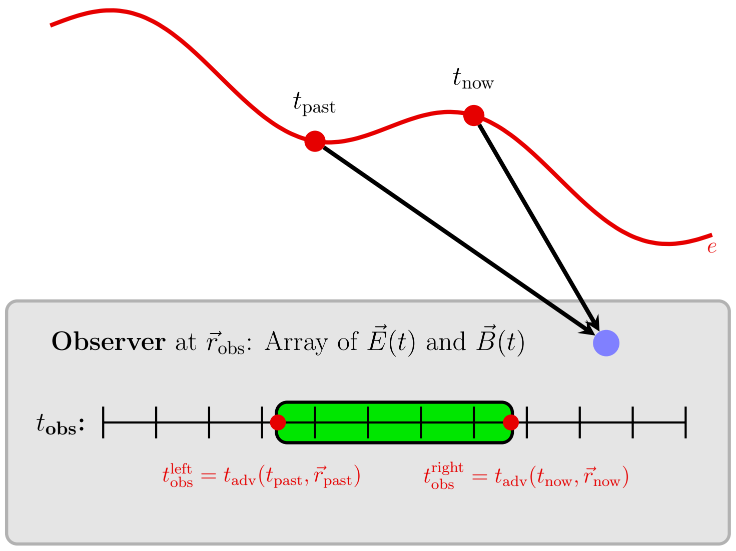

t_{past}
t_{now}
e
Observer at $\vec{r}_{\text{obs}}$: Array of $\vec{E}(t)$ and $\vec{B}(t)$
t_{obs}:
$t_{\text{obs}}^{\text{left}} = t_{\text{adv}}(t_{\text{past}}, \vec{r}_{\text{past}})$
$t_{\text{obs}}^{\text{right}} = t_{\text{adv}}(t_{\text{now}}, \vec{r}_{\text{now}})$

Figure 3.1: Calculating fields at the observer’s position (blue dot) to an advanced time step t_{obs}: Using two consecutive time steps t_{past} and t_{now} (red dots) of the trajectory (drawn in red), it is possible to fill the observer’s field array (time domain) by interpolating between the two advanced time steps $t_{\text{adv}}(t_{\text{past}}, \vec{r}_{\text{past}})$ and $t_{\text{adv}}(t_{\text{now}}, \vec{r}_{\text{now}})$, but less or more time steps must be calculated depending on the projection of advanced time onto observer time t_{obs}. In this example field data of four time steps will be calculated and stored in memory (green).

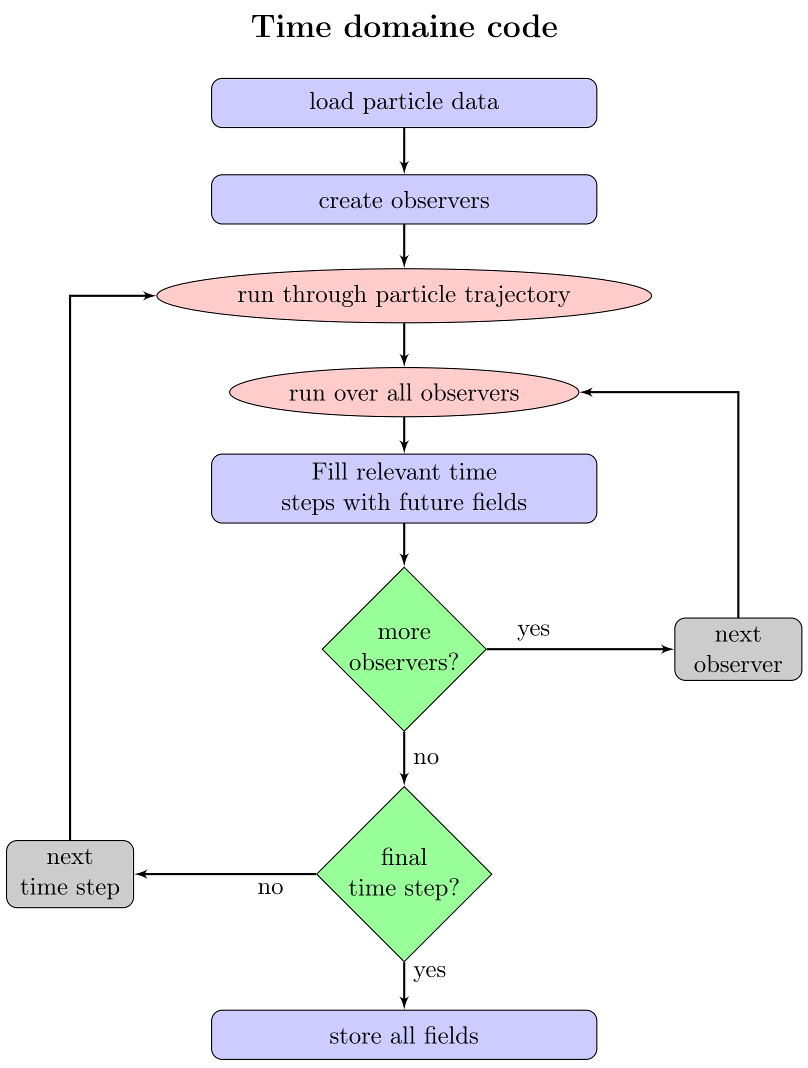


Figure 3.2: Flow chart of the time domain code

the program, following a user-given setup for locations and times at which to compute the fields. These two steps need adjustment for every particular physics case. To ensure for such flexibility, the code can be easily adjusted by a parameter file and a general input routine. This allows to handle different input formats and quickly adjust the positions of the *observers*.
Following the setup, the routine will go through the entire trajectory of a particle, calculating at each time step the future field values of all observers, $\vec{E}(\vec{r}_{\text{obs}}, t_{\text{obs}})$ and $\vec{B}(\vec{r}_{\text{obs}}, t_{\text{obs}})$. The algorithm uses two nested loops, the outer is running over the time steps of the particle trajectory, the inner runs over all *observers*. The calculation for each observer is independent. This part can

be easily parallelized using *data parallelism* (see chapter B.1). However, since the implemented algorithm is already quite fast, a massive parallelization did not seem necessary and instead a more simple parallelization using *OpenMP* (see chapter B.2) was used. This allowed to execute the code on different cores of a CPU resulting in a speedup of approximately ~ 60 on the fastest CPU available at HZDR.
It is also possible to parallelize the loop over time. This would require to calculate all unit conversions and derivatives independently by all threads or to calculate and store these values in advance. This was not done yet but it could be implemented to further speed up the code.
Since the acceleration of the particle is needed to calculate the electric and magnetic field (Eq. 3.3 and Eq. 3.4) but is usually not included in the particle files, the accelerations needs to be computed by differentiating the velocity. Therefore, the *particle handler* also comprises code to calculate numerical derivatives for all stored continuous sets of values. By default, a second order centered derivative (Eq. 3.6) is implemented [20], but any other numerical differential scheme can easily be included.

$$\frac{\mathrm{d}\,f}{\mathrm{d}\,t}(t') = \frac{f(t' + \Delta t) - f(t' - \Delta t)}{2 \cdot \Delta t} + \mathcal{O}(\Delta t^2) \tag{3.6}$$

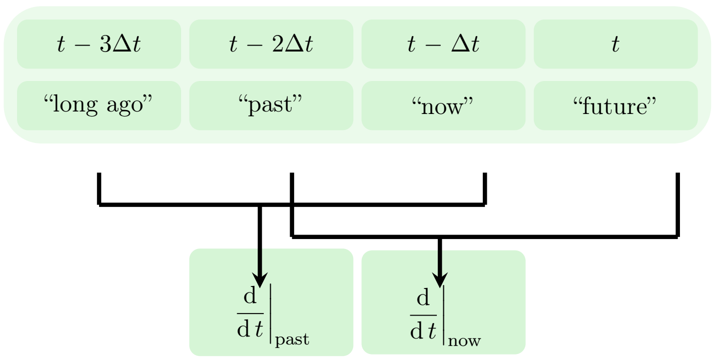

$t - 3\Delta t$
$t - 2\Delta t$
$t - \Delta t$
t
“long ago”
“past”
“now”
“future”
$\left.\frac{\mathrm{d}}{\mathrm{d}\,t}\right|_{\text{past}}$
$\left.\frac{\mathrm{d}}{\mathrm{d}\,t}\right|_{\text{now}}$

Figure 3.3: Illustration of the differentiation algorithm implemented: A second order numeric differentiation scheme accesses trajectory points at four consecutive time steps to calculate derivatives at two time steps.

In order to calculate the numerical derivatives, the trajectory is stored at discrete points referring to the last four time steps, thus allowing to calculate the derivatives at two of these time steps as illustrated by Fig. 3.3. For simplicity, these two time steps are called “past” and “now” in agreement with the method names used in the source code to work with these time steps.

3.2 Simulating energy deposition in frequency domain

This section presents a code called *CLARA 2.0*, standing for CLAssical RAdiation program version two. It allows to calculate the radiated energy per unit frequency and unit solid angle far away from the particle $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ based on Eq. 2.44. For simplicity, this term will be referred to as the spectrally resolved emission intensity or simply the spectrum.

3.2.1 Introduction to CLARA 2.0

Calculating the spectrally resolved radiation at a large distance from the charged particle dynamics provides means to compare particle simulations with experiments. *CLARA 2.0* is able to compute the spectra $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ of a variety of particles for arbitrary directions $\vec{n}$ and a user-defined set of frequencies ω. For each particle, it requests a trajectory and momenta for consecutive time steps from an external source, such as a data file. This separation of the analyzing tool from the actual particle simulation permits to use this code independently of the particular simulation providing the data on the particle dynamics. However, this separation comes at the cost of a strongly reduced efficiency due to the overload produced by data input and output.
The code was developed independently from the *CLARA 1.0* code, designed by A. Debus [18]. *CLARA 2.0* works efficiently on high performance clusters by exploiting their parallel architecture. Thereby, it allows to retrieve results much faster, enabling e. g. more parameter scans in the same time.

3.2.2 Implementation

The spectrum $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ calculated for a single particle and a single direction $\vec{n}$ is needed in order to compute the total spectrally resolved emission intensity for all particles and various directions (Eq. 2.44). This offers means to parallelize the computation, which will be covered at the end of this section. First, calculating a spectra for a single particle and a single direction is discussed (blue part of Fig. 3.5).
Calculating $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ means an integration of radiation amplitudes $\vec{A}$ (Eq. 2.45) over time. These amplitudes $\vec{A}(\vec{\beta}(t),\dot{\vec{\beta}}(t),\vec{n})$ can be calculated using the particle trajectory. Since the integration is similar to a Fourier transform, it is possible to use a fast Fourier transform (FFT) to speed up the calculation. The required derivations are described in section 4.1.2. One consequence of using a fast Fourier transform is the need for equidistantly sampled time steps in retarded time.
If the particle trajectories are sampled equidistantly in simulation time, the

resulting retarded time will not be equidistant since it depends both on the time t and the position of the particle $\vec{r}_e(t)$. Therefore, the retarded time is not suited to perform a fast Fourier transform directly.
In order to overcome this problem, the algorithm iterates over the trajectory calculating a real-valued radiation amplitude $\vec{A}^{\mathrm{FFT}}$ and the associated retarded time t_{ret} for each time step of the trajectory and storing these paired values for later.

$$\begin{aligned} \vec{A}^{\mathrm{FFT}} &= \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta} \right) \times \dot{\vec{\beta}} \right]}{\left(1 - \vec{\beta} \cdot \vec{n} \right)^3} && (3.7) \\ t_{\mathrm{ret}} &= t - \frac{\vec{n}\vec{R}}{c} && (3.8) \end{aligned}$$

The amplitudes $\vec{A}^{\mathrm{FFT}}$ are equidistant in time but not yet equidistant in retarded time as needed for the FFT (Fig. 3.4a). In order to obtain an equidistant sampling in retarded time, the stored amplitudes $\vec{A}^{\mathrm{FFT}}$ are interpolated onto an equidistant grid in retarded time. This is done by using a linear interpolation between two consecutive retarded time steps. The grid in retarded time is generated with $N_{\mathrm{FFT}} = 2^n \geq N_{\text{time steps}}$ points in time, placed equidistantly between the first and last stored retarded time (Fig. 3.4b). After interpolating onto the grid, the spectrum is calculated using a fast Fourier transform (Fig. 3.4c).

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\Omega \mathrm{d}\omega} = \frac{q^2}{16 \cdot \pi^3 \cdot \varepsilon_0 \cdot c} \frac{t_{\mathrm{end}} - t_{\mathrm{start}}}{N_{\mathrm{FFT}}} \left| \underbrace{\sum_{t_{\mathrm{ret}}=t_{\mathrm{start}}}^{t_{\mathrm{end}}} \vec{A}^{\mathrm{FFT}}(t_{\mathrm{ret}}) \cdot \mathrm{e}^{\mathrm{i}\omega t_{\mathrm{ret}}}}_{\text{Fourier transform}} \right|^2 \tag{3.9}$$

Using the fast Fourier transform can cause a variety of issues concerning the resolution of the computed spectra. These are covered in section 4.1.3 and 4.1.5. In order to ensure that the interpolation is able to resolve all details of the amplitude changes over retarded time, $\vec{A}^{\mathrm{FFT}}(t_{\mathrm{ret}})$, it is necessary to set N_{FFT} larger than the number of time steps $N_{\text{time steps}}$ (sampling - section 4.1.3). Additionally, zeros can be added to the end of the amplitudes to improve the spectral resolution (zero-padding - section 4.1.5).
As mentioned in appendix A.3, the implemented fast Fourier transform closely follows the code described in [21] but was written in a more general form.

After performing the Fourier transform, the resulting electromagnetic spectra, being calculated for frequencies depending on the equidistant grid, need to be mapped to predefined frequencies of interest, specified by the user

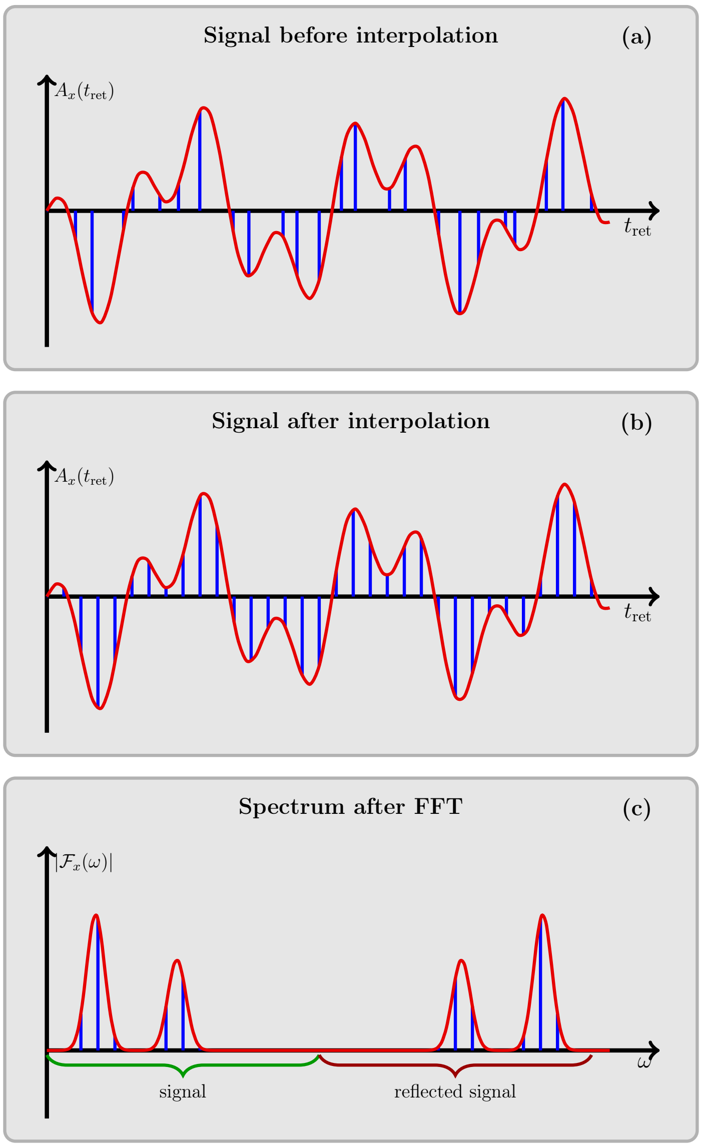


Figure 3.4: In order to process the non-equidistant radiation signal (a), the signal is first interpolated onto an equidistant grid in t_ret (b). With this equidistant signal, a FFT can be applied. The resulting spectrum (c) contains as many frequency values as the grid contains t_ret. Half of the spectrum is considered for spectral analysis, the other half is a reflection at Ω_Nyquist.

before starting the simulation. Two methods are available to map the calculated spectra onto the predefined grid of frequencies: A simple interpolation between the two neighbouring frequencies surrounding the frequency of interest and an averaging scheme over several frequencies in a pre-defined range around the frequency of interest. The first method works best if the frequencies calculated using the FFT are spaced similarly to the frequencies of interest. However, if the frequencies returned by the FFT algorithm are very dense, spectral peaks might be omitted by sampling too roughly with the interpolation method. Here, the averaging approach works better.

After performing all these calculations, one spectrum for a user-defined set of frequencies and for a single particle has been calculated. This is repeated for all directions the user wants to observe and for all particles he wants to cover. In order to do this efficiently, *CLARA 2.0* exploits the parallel architecture of computer clusters to run these tasks in parallel. Calculating these spectra is an independent task for each direction and each particle trajectory, allowing to parallelize the procedure over these quantities.

The basic procedural structure of *CLARA 2.0* is designed similarly to the time domain code described in section 3.1. The main difference is due to the parallelization of the program.
The parallelization scheme implemented assumes a separate calculation on independent computer nodes. Sharing memory or communication is just rudimentarily necessary because the problem is "embarrassingly parallel". This allows to use the code with the already available job submission system or by running it using MPI (for details on both methods see appendix B.2). Additionally, a many-core parallelization using OpenMP (see appendix B.2) might be used if needed.
Computing a spectrum $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ for one particle and one direction $\vec{n}$ is a single tasks. Tasks can be distributed among different nodes (computers) according to a parallelization scheme.
If this user-defined parallelization scheme distributes the calculation for particles among several nodes, each trajectory file will be loaded just once by a single node. This reduces the load on the network connecting the file system with the individual nodes. On the downside, each node needs to store spectra for several directions at the end of the simulation, giving rise to a high network load.
Another parallelization scheme would be that each node calculates the spectra of all particles for a single direction. This requires all nodes to each load all trajectory files, resulting in a high network load. When saving the results, the spectra from different particles are combined in order to reduce the network load. This reduction, however, leads to the loss of individual spectra associated with single particles. If this reduction is not performed, the network load while saving will be comparable to the former paralleliza-

tion scheme.
The first parallelization scheme is implemented by default to prevent this loss of information and because of the fact that for trajectories of more than ~ 20.000 time steps the network load of the second scheme always outweighs that of the first one. However, the parallelization can easily be modified by switching the schemes, for example in case of short trajectories but detailed spectra where the second scheme is more efficient.
As previously mentioned, the parallelization can be implemented using either the job submission system of the cluster or MPI. The first version of the program was designed to work with the job submission system. It creates an array of jobs differing only by a unique ID. This ID is used to load a specific trajectory (or choose a certain direction in case of parallelization over directions) and to give the output stored on disk a unique identifier. The starting and handling of the individual jobs is carried out by the job submission system. Since job handlers are known to not scale well for large numbers of jobs, an MPI based version was developed as well. It emulates a basic job handling routine calling tasks by a unique ID. This framework allows to submit large numbers of independent but similar tasks.
In order to analyze the radiation of a set of moving charges, their trajectories and momenta need to be available as ASCII files on disk. The storage format of the trajectory is arbitrarily chosen and can be modified if appropriate in analogy to the time domain code.
The locations of all trajectory files on disk need to be assigned to a unique ID number in order to run the analysis in parallel. With that ID, all tasks load their associated trajectory and allocate memory for all spectra to be calculated (Fig. 3.5). Then, the routine iterates through all directions previously specified to be of interest. Each iteration calculates the spectrum for a single direction using the FFT algorithm as described at the beginning of this section. After having calculated the spectra $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ for all directions $\vec{n}$, the data created is stored on disk and the next trajectory can be analyzed.
Spectra can be stored as ASCII files or as binary files. ASCII files have the advantage of being human readable and easier to handle by data analysis and plotting tools, however, they are usually about three times larger than a binary file containing the same information. In addition, interpreting text requires a considerable amount of computation if compared to reading files written directly in a machine-compatible way.
Another routine was implemented to combine all created spectra stored on disk. It combines the spectra of all particles, assuming that the phase relation between the radiation of the particles is not important. This is the so called *incoherent radiation method* covered in section 4.2.1. It is a valid approximation if the average distance between particles exceeds the wavelength of the emitted radiation.
For visualization and post-processing a separate program was implemented

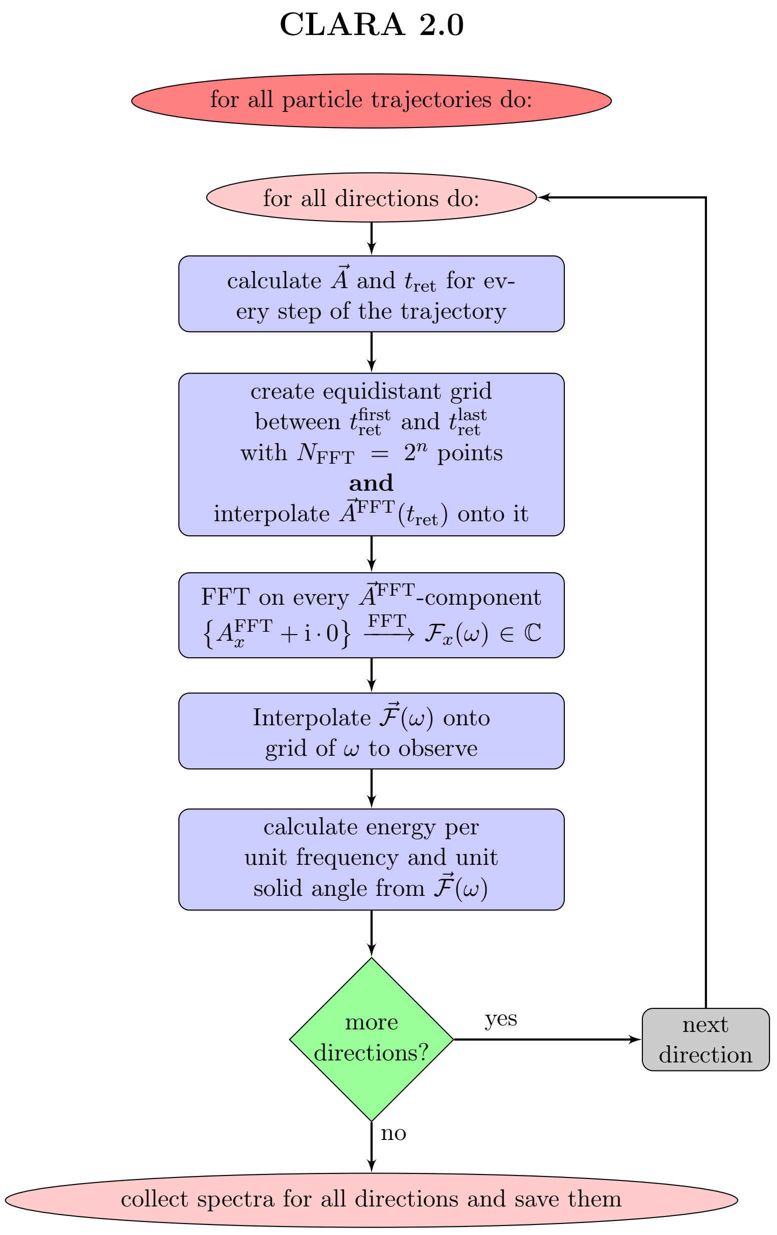

CLARA 2.0
for all particle trajectories do:
for all directions do:
calculate $\vec{A}$ and t_{ret} for every step of the trajectory
create equidistant grid between $t_{\text{ret}}^{\text{first}}$ and $t_{\text{ret}}^{\text{last}}$ with $N_{\text{FFT}} = 2^n$ points **and** interpolate $\vec{A}^{\text{FFT}}(t_{\text{ret}})$ onto it
FFT on every $\vec{A}^{\text{FFT}}$-component $\left\{A_x^{\text{FFT}} + \mathrm{i} \cdot 0\right\} \xrightarrow{\text{FFT}} \mathcal{F}_x(\omega) \in \mathbb{C}$
Interpolate $\vec{\mathcal{F}}(\omega)$ onto grid of ω to observe
calculate energy per unit frequency and unit solid angle from $\vec{\mathcal{F}}(\omega)$
more directions?
yes
next direction
no
collect spectra for all directions and save them

Figure 3.5: Flow chart of the inner part of the frequency domain code

using the programming language Python.

3.3 Radiation code working within particle-in-cell codes

The third code is also able to compute the spectrally resolved intensity emitted by electrons $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$, but in contrast to *CLARA 2.0*, the direct implementation in the particle simulation *PIConGPU* allows to analyze much larger particle systems in a much shorter time. The code is named *Radiation Analyzer*. In combination with *PIConGPU*, it is able to predict radiation emanating from a variety of plasma processes, e. g. from laser wakefield acceleration [5].

3.3.1 Idea behind this approach

Even on large compute clusters, it has several disadvantages to separately calculate the particle motion and the resulting radiation. First of all, trajectories of particles need to be stored on hard drives. However, for plasma simulations with billions of macro particles involved the disk space available is easily filled.
In order to illustrate this, let us assume that trajectories are stored with double precision, causing every floating point value to use 8 bytes of memory. One needs to store a location and a momentum vector per particle and time step, as well as a particle ID or a time, requiring another 8 bytes. Hence, the total disk space required is:

$$M = N_{\text{time}} \cdot N_{\text{particle}} \cdot (3+3+1) \cdot 8\,\text{byte} \tag{3.10}$$

This limits the number of particles to be considered for an exemplary simulation with $N_{\text{time}} = 50,000$ time steps to one hundred millions, assuming one could use the $M = 256\,\text{TB}$ of available disk space at HZDR solely for this simulation.
A laser wakefield simulation typically requires a gas density of $\rho = 10^{19}\,\text{cm}^{-3}$ in a volume of $V = 0.6\,\text{cm} \cdot 80\,\mu\text{m} \cdot 80\,\mu\text{m}$. With a moderate macro particle weighting of $w = 1000$ real particles per simulated macro particle one would end up with

$$N_{\text{particles}} = \frac{\rho \cdot V}{w} \approx 3.8 \cdot 10^{11}\,\text{macroparticles.} \tag{3.11}$$

The number of macro particles used by this typical particle-in-cell simulation is about three orders of magnitude larger than the limit derived above.
However, this is not even the major problem with that kind of approach. More importantly, writing to and reading from a hard drive is relatively slow compared to analog operations on random-access memory (RAM), even on

parallel file systems. The *hypnos* cluster at HZDR can read files at about 250 MB/s if reading is distributed over several nodes. About two weeks would be needed to read the above 256 TB of data at this speed. Writing would be even slower. In contrast, a CPU can transfer data from and to the RAM with a bandwidth of around $\sim 50\frac{\mathrm{GB}}{\mathrm{s}}$. Modern GPUs are distinguished by a bandwidth of up to $\sim 200\frac{\mathrm{GB}}{\mathrm{s}}$ to its global memory [22]. This is about 200 to 800 times faster than data transfer from disk.
Another problem is that particle-in-cell simulations cannot store entire trajectories at once because of memory (RAM) limitations. Therefore, particle data are usually stored in several files per time step. In order to calculate the radiation, one needs at least a partial history of the particle motion to compute derivatives. This means one has to rearrange all particle data prior to calculating the radiation. This requires at least two steps of sorting and matching which have to be performed thus loosing several weeks before starting with the actual analysis.

One way to overcome these problems is to calculate the radiation directly within the particle simulation. This eliminates the problem of storing trajectories on disk as the radiation code can obtain all particle data within the simulation. However, RAM is also limited. This means one cannot keep entire trajectories but only short parts of it. The length of these sections is determined by the number of time steps that can be held in memory at the same time.
Only if a specific part of the trajectory is simulated and accessible in memory, the associated radiation can be calculated. Consequently, calculating every particle's radiation needs to be carried out in parts.
Radiation data for every particle in several directions cannot be held in memory because of the limited memory of the GPU. Therefore, a reduction of data is essential. To emphasize this, radiation can be associated with the acceleration of a particle and therefore can be seen as a property of the particle. This means that even if one is normally interested in the collective radiation, it would be possible to view every single particle as a separate emitter and to keep track of its radiation. This associates radiation properties with a single particle and is exactly what the *CLARA 2.0* code does. Nevertheless, this requires every particle "keeping track" of its radiation in every direction one is interested in. This increases the amount of data kept in memory tremendously. If one considers just a few particles, this might not be a problem. With millions to billions of particles one would quickly reach any memory limitation.
Reduction can be achieved by adding the complex amplitudes ($\vec{A} \cdot \mathrm{e}^{\mathrm{i}\omega t_{\mathrm{ret}}}$) (Eq. 2.45 and Eq. 2.46) of all particles for each time step. This is the so called *coherent radiation method* that will be covered in section 4.2.1. Mathematically, this can be correctly done by interchanging the sum over all particles in Eq. 4.12 with the integral over retarded time. Using the *coherent radiation*

method allows to combine the radiation data of several particles while still running through the simulation. This method considers the phase relation of the emitted radiation, which is physically correct and allows to correctly simulate *coherent radiation*. Numerically, it needs to be demonstrated that the random interferences caused by the macro particle approach do not alter the final spectrum and the results still comprise all significant physics (see appendix C).
The approach of replacing large disk space by small RAM seems counterintuitive since one of the reasons against the CPU-based code was its limited memory in form of the available disk space. However, the new approach does not need to store entire trajectories and, by using the *coherent radiation method*, it efficiently combines radiation data by summing over many particles. This allows to analyze radiation without being held back by bandwidth limitations.
In order to realize all those ideas, they have been implemented in the particle-in-cell Code *PIConGPU* [23] developed at the HZDR.

3.3.2 Particle-in-cell codes

There are three common ways to simulate a plasma. The first regards the plasma as a fluid and uses a *fluid dynamic* approach. This is only correct if the plasma particles can be described by ensemble-averaged variables such as density and temperature. In the other extreme all particles are treated individually and *molecular dynamics techniques* can be employed. This again turns out to be an impossible task for large-scale systems with many particles, because the sheer number of particles and the fact that all particles interact with all other particles via the Coulomb force demands computational power not available even with the utmost supercomputers. Simplifications are necessary to overcome these problems. The *particle-in-cell* (PIC) approach is a mean field model [19]. It computes the electric and magnetic fields on a grid of cells and then calculates how the particles interact with these fields. Any short-range interactions below the cell size are neglected. Additionally, non-point-like but macro particles are considered, which resemble symmetric particle distribution functions of hundreds of real particles.
The PIC algorithm is illustrated in Fig. 3.6. For the sake of simplicity, the starting conditions of the simulation are left out. Every step in the algorithm updates a specific set of values which influences the next set of values. Assuming the electric and magnetic fields are linearized on the grid (yellow box), the Lorentz force acting on each macro particle in each grid cell can be computed by interpolating the fields on the grid (blue box). This force is then used to compute the change in position and momentum of all particles by solving the equation of motion (green box). This is called the *particle push*. There are several algorithms to efficiently combine these two steps. They are, however, in principle two separate calculations and thus are pre-

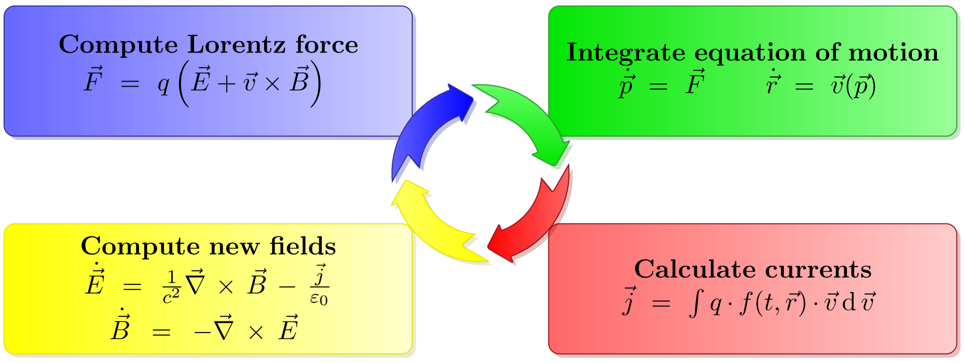

Compute Lorentz force
$\vec{F} = q\left(\vec{E} + \vec{v} \times \vec{B}\right)$
Integrate equation of motion
$\dot{\vec{p}} = \vec{F}$ $\dot{\vec{r}} = \vec{v}(\vec{p})$
Compute new fields
$\dot{\vec{E}} = \frac{1}{c^2}\vec{\nabla} \times \vec{B} - \frac{\vec{j}}{\varepsilon_0}$
$\dot{\vec{B}} = -\vec{\nabla} \times \vec{E}$
Calculate currents
$\vec{j} = \int q \cdot f(t, \vec{r}) \cdot \vec{v} \, \mathrm{d}\, \vec{v}$

Figure 3.6: Particle-in-cell algorithm

sented individually. Since the particles move in each time step, they also induce a current, which needs to be computed (red box). It is modeled as a field linearized on the grid. With the new current, the electric and the magnetic fields can be updated (again yellow box). Using Ampere's and Faraday's laws (Eq. 2.4 and Eq. 2.5), the changes of the fields are superimposed onto the fields on the grid. Now, the electromagnetic fields are updated and one can start over with the algorithm. As with the *particle pusher* all steps mentioned can be implemented with a variety of algorithms which differ in speed, complexity and numerical accuracy. Since these details are not of importance to understand the *Radiation Analyzer*, the interested reader is referred to [19]. Using a grid to describe the fields not only has the advantage of reducing the computational cost of the algorithm but also to make the algorithm only act on local values in the simulation domain. Particles in a cell are only influenced by fields on the surrounding grid points and vice versa. Combining several cells locally allows to calculate the plasma in this region except for a small boundary region. This property allows to spreading the calculation on several computing nodes and to broadcast only a relatively small set of values. This so-called domain decomposition method has to be considered when calculating radiation.

3.3.3 Implementation

In order to calculate the radiation using the discretely sampled trajectories from *PIConGPU*, Eq. 2.44 needs to be adjusted. The integral over the time can be replaced by a sum over all time steps. Furthermore, the radiation from all particles needs to be considered with their phase. This can be done by adding all particle contributions before calculating the absolute square of the integrals [6]. It is no problem to interchange the sum over all particles with the sum over time.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\Omega \mathrm{d}\omega} = \frac{1}{16 \cdot \pi^3 \cdot \varepsilon_0 \cdot c} \cdot \left| \sum_{t=t_{\text{start}}}^{t_{\text{end}}} \sum_{k=1}^{N_{\text{p}}} q_k \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta}_k \right) \times \dot{\vec{\beta}}_k \right]}{\left(1 - \vec{\beta}_k \cdot \vec{n} \right)^2} \cdot e^{i\omega\left(t - \frac{\vec{n}\vec{R}_k}{c}\right)} \Delta t \right|^2 \tag{3.12}$$

A total of N_{p} macro particles is considered. The index k describes a single macro particle with position $\vec{R}_k$, velocity $\vec{\beta}_k$ and acceleration $\dot{\vec{\beta}}_k$. Since macro particles can describe a variety of particles, the total charge of a macro particle q_k can differ and needs to be considered before adding the amplitudes of all particles. The total energy emitted per unit solid angle and unit frequency can be calculated for arbitrary directions of the observer $\vec{n}$ and for arbitrary frequencies ω. This offers a means to parallelize the computation if considering several directions and frequencies.

The positions and momenta of all marco-particles are calculated by *PIConGPU* and can be accessed for calculating the radiation. For each time step the sum over all particles needs to be evaluated for each direction $\vec{n}$ and frequency ω. The result can be added to the previous results until the end of the simulation. After the final time step of the PIC simulation, the total energy emitted per unit frequency and unit solid angle can be calculated using Eq. 3.12.

This sketch of the algorithm implemented hides many details of the calculations, the concrete implementation in *PIConGPU* is more difficult. In order to compute the emitted radiation directly in *PIConGPU*, the radiation code was implemented as a so-called *Analyzer*. The *Analyzer* is a module that can be executed after one PIC cycle. It is used to analyse the simulated plasma. *Analyzers* come with a variety of predefined methods to permit standardized configuration and invocations by the *Analyzer Controller* as shown in Fig. 3.7. Before the startup of the PIC simulation, *PIConGPU* calls the *Analyzer Controller* to check which analyzers have been activated and to load the user-defined start parameters. These are then handed over to the *Analyzers* which allocate enough memory on all computer nodes and GPUs and set up all necessary parameters. The *Radiation analyzer* also extends the particle properties by adding a previous momentum. This is necessary to numerically differentiate the momentum and requires to extend the particle pusher algorithm at compile time.

During the simulation run and if the GPU based PIC-algorithm has completed one cycle, the CPU-based part of PIConGPU handles all necessary communications with other computer nodes, but also starts the *Analyzer Controller* routine. This scheduler calls sequentially all activated analyzers. This includes, if activated, the *Radiation Analyzer*. It starts the *Radiation Kernel*, which performs all necessary calculations on the GPUs and will be described later. If asked for by the user, the *Radiation Analyzer* collects the

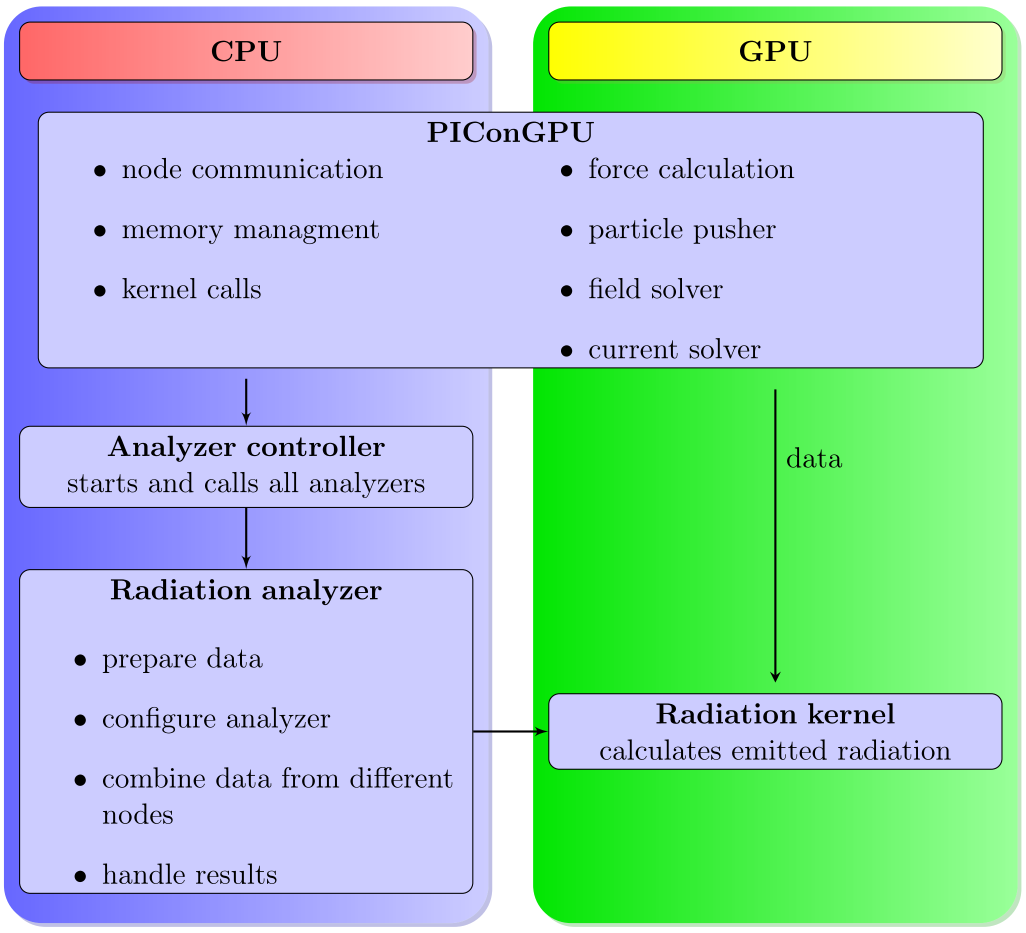


Figure 3.7: Flow chart of PIConGPU and the radiation kernel

results of the calculations from the GPUs, merges them on a single node and stores them on disk.
After the termination of the PIC simulation, again all data are collected and stored. Finally all allocated memory is freed and PIConGPU can terminate. Before discussing the algorithm which calculates the radiation for every particle, the *Radiation Kernel*, it is important to think about how to parallelize this computation for a highly parallel architecture as the GPU. The computation itself is parallel over all particles, all directions and - if one uses a discrete Fourier transform method (DFT, see appendix A.2) - even over all frequencies. However, not all possible parallelization schemes can be exploited efficiently on the GPU. One way to parallelize the problem is shown in Fig. 3.8. This is the first scheme implemented with a high efficiency. It is no longer used since there is a slightly faster scheme which relies on more

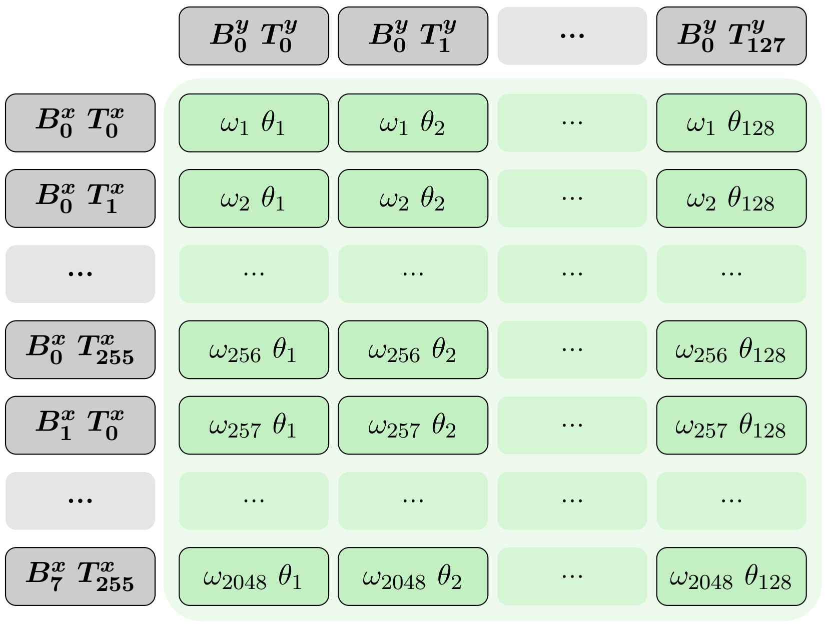


Figure 3.8: Parallelization over all particles and directions

communication to speed up the calculation by a few percent. Nevertheless, this example nicely illustrates how to exploit the parallel architecture and how the basic idea of the new scheme works. It utilizes the independence of frequencies and directions and uses a two dimensional CUDA-grid to account for that.
A CUDA-grid is the overall structure that describes all calculations needed to be computed in parallel. It contains several blocks that group several of these calculations, named threads, together. All threads can access the same global memory, while only threads in one block share data with each other via the fast *shared memory.* A grid can describe blocks in several dimensions to facilitate the parallelization. In the following parallelization schemes, blocks are marked with B_i^x, representing the i^{th} block in dimension x, and threads with T_j^x, with j the ID of the thread and x its dimension. For details on terms used in parallel programming based on GPUs see appendix B.3.
One grid dimension is associated with the direction $\vec{n}(\theta)$. It just consists of one block (B_0^y) of as many threads as are needed for all directions. These can point towards any direction but are usually described by a single angle θ. The second dimension relates to frequencies. For 2048 frequencies it is divided up into 8 blocks each comprising 256 threads.
Every thread is associated with one and only one independent task, which can be interpreted as one pixel in graphs like Fig. 2.7. The parallelism in particle data is exploited by letting each thread in a frequency block load data of a single particle, then calculate the current amplitude vector $\vec{A}$ (Eq. 2.45)

and the retarded time t_{ret} (Eq. 2.46), and finally share them with all other threads in the block. This tremendously reduces the amount of communication compared to a version where every thread has to load the entire particle data on its own. Finally, every thread still has to iterate over all particles to calculate the contribution of all shared $\vec{A}$ and t_{ret} to its final result. This algorithm is still sequential for particles on a GPU, even if the load process was parallelized, but it is parallel in directions and frequencies.
Since there are usually several GPUs used for a simulation, work is divided among them. A parallel execution with regard to the particles is achieved and the contributions from all particles on all GPUs have to be combined by the *Radiation Analyzer*. At the end of each calculation on the GPU, the *Radiation Analyzer* is responsible to combine all results from the different nodes.
The second scheme uses a simple one-dimensional parallelization as illustrated in Fig. 3.9. There is just a one-dimensional grid with a block for each direction. For better illustration, the treads contained in a block are drawn on a different axis. Every block consists of 256 threads, which can be associated with different tasks during the execution of the kernel. All tasks

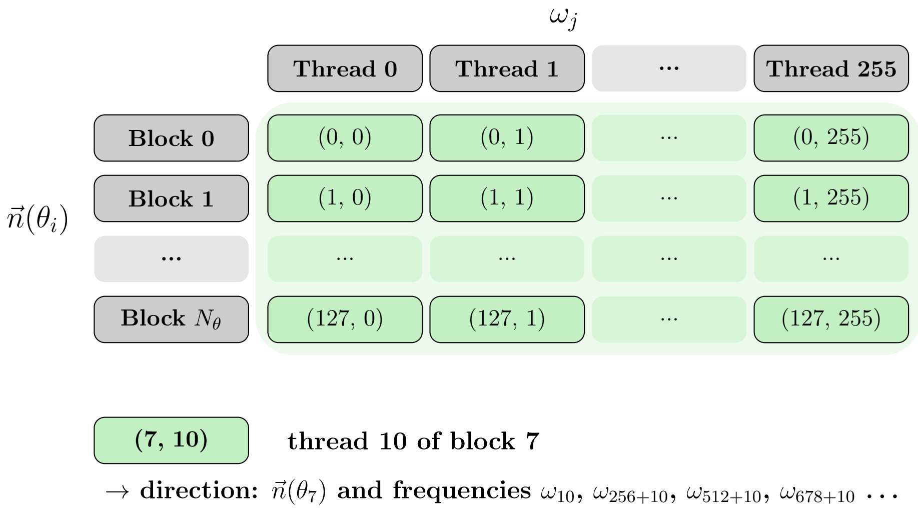


Figure 3.9: Current parallelization scheme

associated with the individual threads are shown in a flow chart depicting the *Radiation Kernel* (Fig. 3.10). Every block runs sequentially over all *super cells*, a collection of individual cells which was created to better handle data in *PIConGPU*, on the GPU. Every *super cell* is associated with several *particle frames*, which is just a list of at most 256 particles. The frame's size is determined by the PIC algorithm on the GPU, which needs to switch the association of threads between particles and cells. Therefore, the number of

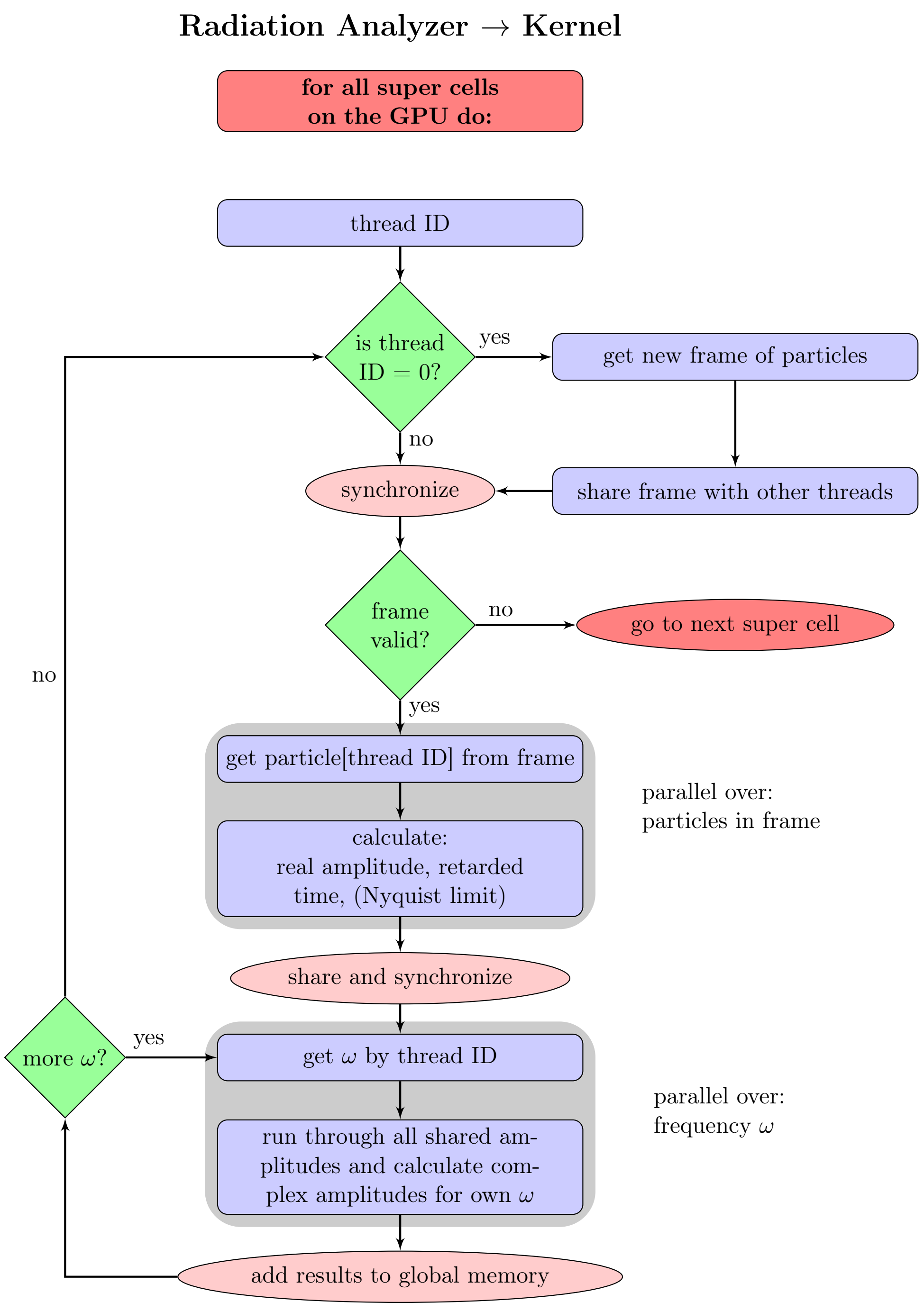

Radiation Analyzer → Kernel
for all super cells on the GPU do:
thread ID
is thread ID = 0?
yes
get new frame of particles
no
share frame with other threads
synchronize
frame valid?
no
go to next super cell
yes
get particle[thread ID] from frame
calculate: real amplitude, retarded time, (Nyquist limit)
parallel over: particles in frame
share and synchronize
more ω?
yes
no
get ω by thread ID
run through all shared amplitudes and calculate complex amplitudes for own ω
parallel over: frequency ω
add results to global memory

Figure 3.10: Flow chart of radiation kernel

cells equals the number of particle frames in a supercell. Usually more than one frame per super cell is required since there can be more than 256 particles in a super cell. The super cell contains a reference to the last particle frame, which might contain less then 256 particles that are located in the super cell. This frame points to its fully filled predecessor, and that to its predecessor, until the first frame points to an invalid frame, which represents the end of the list of particle frames. By iterating over all frames, one can thus iterate over all particles located in a super cell.
In order to process these frames in parallel, each thread must first determine its thread ID in the block. If it is the first thread, it determines at start the last frame of the super cell and later the predecessors. It shares the location of the particle list with all other threads but is not loading the data. Now, all threads check if the current frame is valid, or if they already dealt with all frames and are now finished with all particles of that specific super cell. Since all threads are checking the same frame, they will either jump to the next super cell until no super cell is left on the GPU or start with loading particle data.
Every thread loads one particle of the 256 particles in the frame to its own register. Then, it calculates the real amplitude $\vec{A}$ (Eq. 2.45) and the retarded time t_{ret} (Eq. 2.46) for this particle for a direction determined by the block ID of the thread. As an example thread $(7, 10)$ in Fig. 3.9 computes both values using the direction $\vec{n}(\theta_7)$, with θ_7 a user-defined angle. The real amplitude and retarded time are shared with all other threads in that block.
If the frame was the last frame, not all threads might end up with a particle or in some cases not all particles are selected for the radiation calculation. In those cases, only those threads with valid data share it. Since the data is needed by all threads in the next phase of the kernel, the threads need to wait until the last of them is finished. Then, each thread computes a frequency ω by its thread ID j and its direction θ by its block ID i. The specific implementation of calculating the frequency and direction can be defined by the user. Using a logarithmic frequency range or calculating the radiation observed on an entire sphere surrounding the plasma (sky map) is possible. An example of a simple linear frequency and direction mapping is given below.

$$\omega = \omega_{\mathrm{start}} + j \cdot \Delta\omega \tag{3.13}$$

$$\theta = \theta_{\mathrm{start}} + i \cdot \Delta\theta \tag{3.14}$$

$$\vec{n} = \vec{n}(\theta) = (\sin\theta, \cos\theta, 0) \tag{3.15}$$

Here, ω_{start} and θ_{start} are the offsets of the frequency and angle; $\Delta\omega$ and $\Delta\theta$ are their step width. The direction is computed by means of trigonometric functions that can be set at compile time. All threads in the block then use the shared real amplitudes $\vec{A}$ and retarded times t_{ret} to calculate the complex amplitudes $\vec{A} \cdot \mathrm{e}^{-\mathrm{i}\omega_j t_\mathrm{ret}}$ of each particle for their frequency and for the direction of their block. All complex amplitudes a thread computes are added up and the result is kept in each threads' register memory. After the threads have finished iterating over all particles in the frame, they store the calculated radiation amplitudes in global memory by adding the new amplitudes to the already existing ones. If the number of frequencies to calculate is larger than the number of threads in a block, all threads will repeat this last procedure until the radiation amplitudes for all frequencies have been calculated. If this is the case, the next frame will be determined and the procedure starts over again as illustrated in the flow chart (Fig. 3.10).

The *Radiation Kernel* comprises also a *Nyquist limiter* algorithm (section 4.2.2) to correctly calculate the radiation of particles with a widely spread energy distribution. The Nyquist limit of each particle is calculated by the thread loading the particle and shared additionally with the other threads. Each thread then decides according to its assigned frequency whether an amplitude is added to the total sum or not. Since the Nyquist limiter needs more shared memory, it can optionally be excluded by the user.

3.3.4 Discrete versus fast Fourier transform

The above scheme describes a calculation of the radiation using a discrete Fourier transform (DFT, see appendix A.2) scheme with a non-equidistantly sampled retarded time. It needs $N_\omega \cdot N_t$ calculations for just one particle and one direction, with N_ω being the number of frequencies to calculate and N_t the number of time steps of the PIC simulation. This is a huge amount of calculations and it comes to mind to use the fast Fourier transformation (FFT, see appendix A.3) instead. Another *Radiation Analyzer* has been implemented based on the FFT algorithm. In order to use the FFT, an equidistantly sampled signal in retarded time is required. Due to the limited memory available on a GPU, it is not possible to keep entire trajectories of all particles. Therefore, the method used in the CPU-based frequency domain code, *CLARA 2.0*, is not an option. However, it is possible to predict and thus to limit the range in retarded time for an arbitrary direction. For simplicity, a spherical shape of the simulated region is assumed instead of the cuboid shape used in the simulation (see Fig. 3.11). The range in retarded time thus might be longer than actually necessary, but calculating the ranges is so quite easy and less error prone.

An equidistant grid between the lower and upper limit can be created. The

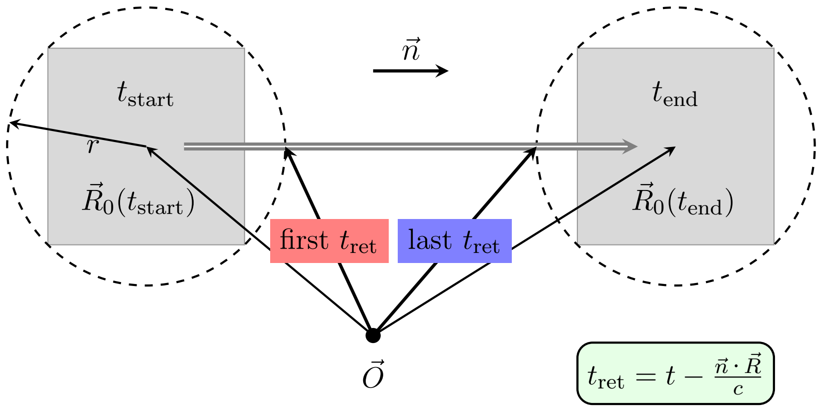

$\vec{n}$
t_{start}
r
$\vec{R}_0(t_{\text{start}})$
t_{end}
$\vec{R}_0(t_{\text{end}})$
first t_{ret}
last t_{ret}
$\vec{O}$
$t_{\text{ret}} = t - \frac{\vec{n}\cdot\vec{R}}{c}$

Figure 3.11: Estimating the range of retarded time: The particles of the first simulation time step t_{start} are centered around $R_0(t_{\text{start}})$. The particles of the last simulation time step t_{end} are centered around $R_0(t_{\text{end}})$. The volume simulated can be comprised in a sphere with radius r. The radiation emitted at the first time step t_{start} from particles located on the right side of the sphere is observed first by an observer in direction $\vec{n}$, while the radiation emitted at the last time step t_{end} from particles located on the left side of the sphere is observed last by this observer.

lower $t_{\text{ret}}^{\text{first}}$ and upper $t_{\text{ret}}^{\text{last}}$,

$$t_{\text{ret}}^{\text{first}} = t_{\text{start}} - \frac{\vec{n}\cdot\vec{R}_0(t_{\text{start}})}{c} + \frac{r}{c} \tag{3.16}$$

$$t_{\text{ret}}^{\text{last}} = t_{\text{end}} - \frac{\vec{n}\cdot\vec{R}_0(t_{\text{end}})}{c} - \frac{r}{c} \quad , \tag{3.17}$$

are calculated using the central position $\vec{R}_0$ of the simulated region and the radius r of the surrounding sphere. The spacing between the grid points is defined by the maximum frequency one wants to observe and can be calculated by using the Nyquist-Shannon sampling theorem (Eq. A.12). In order to sum up the real amplitudes on this grid, one needs to keep track of the particles' trajectories for a short period of time, so it is possible to reconstruct the particle's motion for at least two time steps t_{now} and t_{old}. This can be done by extending the particle pusher and the particle data class again. With these two times in lab frame, two points in retarded time, $t_{\text{now}}^{\text{ret}}$ and $t_{\text{old}}^{\text{ret}}$, are calculated. For those, it is possible to compute the real amplitudes and interpolate between them onto the equidistant grid in retarded time as illustrated in Fig. 3.12. It is possible that during one PIC-cycle several or none of the grid points are filled by a particle. In one PIC cycle several particles might contribute to different grid points because they are spatially separated. It is not unlikely that different particles contribute to the same grid point. Therefore addition to the same space in memory by different

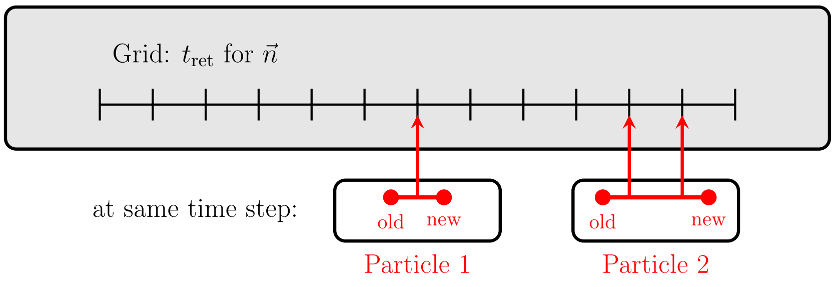


Figure 3.12: Adding amplitudes to a grid in retarded time

threads at the same time needs to be handled sequentially. This makes this procedure inefficient on GPUs. Another possibility would be to keep several grids on the GPU and add onto these grids in parallel. However, this is not an option because of the limited memory of a GPU. Thus, only the calculation of the real amplitude but not the summation of real amplitudes can be efficiently implemented on GPUs. The computation of the real amplitudes only makes up for 90% of the computation time, the rest is needed for interpolating onto the grid. By Amdahl's law (Eq. B.3, [24]), putting parts of the FFT version of the radiation analyzer onto the GPUs will maximally result in a speedup of 10. Comparing the time for such an FFT implementation with the time needed for the DFT shows that the FFT is clearly slower than the DFT. Therefore, the FFT version has not been developed further.

4 | Fourier transform and radiation spectra

Two different programs were developed which simulate the electromagnetic intensity spectra detectable far away from the corresponding particle dynamics (section 3.2 and 3.3). This included developing several algorithms that have been tested and applied to well-known physical problems. This chapter focuses on these algorithms and on some pitfalls one might encounter when using these numerical solutions.

4.1 Investigating different aspects of calculating radiation spectra numerically

4.1.1 Simplifying the calculation of radiation spectra by partial integration

The equation 2.44 allows to calculate the spectrally resolved radiation intensity $\frac{\mathrm{d}\,I^2}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega}(\vec{n},\omega)$ far away from an accelerated charge. There exist a number of equivalent equations used to simplify calculations, or adopt Eq. 2.44 to a particular algorithm.
Quite often, a trick presented in [6] is used. By

$$\frac{\vec{n}\times\left[\left(\vec{n}-\vec{\beta}\right)\times\dot{\vec{\beta}}\right]}{\left(1-\vec{\beta}\cdot\vec{n}\right)^2}=\frac{\mathrm{d}}{\mathrm{d}\,t}\left[\frac{\vec{n}\times\left(\vec{n}\times\vec{\beta}\right)}{1-\vec{\beta}\cdot\vec{n}}\right] \quad , \tag{4.1}$$

applying an integration by parts to Eq. 2.44 one arrives at

$$\begin{aligned}\frac{\mathrm{d}^2\,I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} \quad = \quad & \frac{q^2}{16\pi^2\varepsilon_0 c}\left|\left[\frac{\vec{n}\times\left[\left(\vec{n}-\vec{\beta}\right)\times\dot{\vec{\beta}}\right]}{\left(1-\vec{\beta}\cdot\vec{n}\right)^2}\mathrm{e}^{\mathrm{i}\omega(t-\vec{n}\cdot\vec{r}(t)/c)}\right]_{t_1}^{t_2}\right. \\ & \left. +\omega\int_{t_1}^{t_2}\vec{n}\times\left(\vec{n}\times\vec{\beta}\right)\mathrm{e}^{\mathrm{i}\omega(t-\vec{n}\cdot\vec{r}(t)/c)}\,\mathrm{d}\,t\right|^2 \quad . \end{aligned} \tag{4.2}$$

The result is a formula consisting of two parts, one describing the endpoints of the integrated temporal window, the other the simplified temporal integration. The actual integration consists of a vector part which is easier to handle. The endpoints can often be ignored [6], but are necessary for a correct general description.
This formalism is often used in theoretical calculations. Trying this approach showed that it is numerically unstable and is therefore not well suited for simulations. This instability arises from the integral kernel's independence of the acceleration. The kernel function can consist of a large constant part. In order to illustrate this, an electron with a high energy moving in an undulator is considered. It wiggles just a little bit but mainly keeps its high original velocity $\vec{\beta}(t) \approx \vec{\beta}_0 + \vec{\beta}_u(t)$, with $|\vec{\beta}_u| \ll |\vec{\beta}_0|$ along the undulator axis. Using the linearity of the Fourier transform, it is obvious that β_0 will only contribute to $\omega = 0$ and together with the multiplication with ω, the contribution of $\vec{\beta}_0$ to the radiation will result to zero. As expected, only the acceleration will contribute to the radiated energy. However, this is only true if staying outside the realms of numerical calculations. A numerical approach sums up samples of the integration kernel at discrete points in time. But this is always flawed by errors caused by the sampling rate, the limited precision of floating point numbers, and the non-commutative properties of floating point arithmetics. The resulting error is proportional to the integration kernel and can be large compared to the contribution of the acceleration. It is however possible to circumvent these problems for special cases. If one knows a priori that a particle has a nearly constant velocity, this constant part can be subtracted beforehand when calculating the radiation. Nevertheless, this is not applicable to a case with strong acceleration or to any general situation.

4.1.2 Speeding up the calculation of radiation spectra by fast Fourier transform

A brief look at Eq. 2.44 reveals a similarity to a Fourier transform (see Appendix A) except that one integrates over simulation or lab time t, while the complex phase depends on the retarded time $t_{\mathrm{ret}}(t, \vec{r})$. This leads to two options. The integral can be solved by using either a discrete Fourier transform (DFT, see A.2) with a non-equidistant sampling rate, or by integrating over an equidistantly sampled signal in t_{ret} and using a fast Fourier transform (FFT, see A.3). The first option is a straightforward approach replacing the integral over time by a sum.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} = \frac{q^2 \Delta t^2}{16\pi^2 \varepsilon_0 c} \left| \sum_{t_i = t_{\mathrm{start}}}^{t_{\mathrm{end}}} \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta}(t_i) \right) \times \dot{\vec{\beta}}(t_i) \right]}{\left(1 - \vec{\beta}(t_i) \cdot \vec{n} \right)^2} \cdot \mathrm{e}^{\mathrm{i}\omega (t_i - \vec{n} \cdot \vec{r}(t_i)/c)} \right|^2 \tag{4.3}$$

The second option requires more effort. In order to integrate over t_{ret}, it is necessary to substitute t by t_{ret} in Eq. 2.44. It can be shown [6] that

$$\frac{\mathrm{d}\,t_{\mathrm{ret}}}{\mathrm{d}\,t} = 1 - \vec{\beta}\cdot\vec{n} \quad . \tag{4.4}$$

Using this relation leads to an integration over retarded time.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\,\Omega\,\mathrm{d}\,\omega} = \frac{q^2}{16\pi^2\varepsilon_0 c}\left|\int_{-\infty}^{+\infty} \frac{\vec{n}\times\left[\left(\vec{n}-\vec{\beta}\right)\times\dot{\vec{\beta}}\right]}{\left(1-\vec{\beta}\cdot\vec{n}\right)^3}\cdot \mathrm{e}^{\mathrm{i}\omega t_{\mathrm{ret}}}\,\mathrm{d}\,t_{\mathrm{ret}}\right|^2 \tag{4.5}$$

This is a Fourier transform over retarded time t_{ret}. The FFT algorithm requires an equidistant sampling. Equidistant step width in t can be achieved by most simulations. However, an equidistant sampling in t will rarely result in an equidistant sampling in t_{ret} since the retarded time t_{ret} depends on both the time t and the location of a charge $\vec{r}(t)$. There are two ways to obtain an equidistant sampling in t_{ret}: The first assumes that it is possible to store all real amplitudes $\vec{A}$ (Eq. 2.45) with their associated t_{ret} (Eq. 2.46). After having calculated and saved all values, they can be interpolated onto an equidistant grid. The additional computation needed for the interpolation scales linearly with the number of grid points and is less time-consuming than the Fourier transform.

The second way is to construct a suitable, equidistant grid in retarded time before calculating every value. If such a grid is available, only two adjacent values in t_{ret} need to be kept in memory. Between these two time steps, an interpolation can be performed to fill any grid points in between. Both methods create an equidistantly sampled signal in retarded time, allowing to compute spectra by means of the highly efficient FFT algorithm.

4.1.3 Equidistant and non-equidistant sampling of retarded time

The sampling rate, or equivalently the spacing of the calculated amplitudes $\vec{A}$ in t_{ret}, leads directly to the sampling problem covered by the Nyquist-Shannon sampling theorem (see A.2). It states that for a signal sampled at a fixed rate $1/\Delta t$ there exists a highest detectable frequency $\Omega_{\mathrm{Nyquist}}$, the Nyquist frequency. If a signals contains a frequency ω' above $\Omega_{\mathrm{Nyquist}}$, the Fourier transform $\mathcal{F}(\omega')$ cannot be reconstructed.

By choosing a discrete Fourier transform over a fast Fourier transform, one can use Eq. 4.3 together with a varying sampling rate in t_{ret}. It is known from [25] that the *average sampling period* $\langle\Delta t_{\mathrm{ret}}\rangle$ needs to be below a critical value defined by the band limitation, or highest contained frequency of the radiation ω_{max}, to reconstruct the signal.

$$\langle \Delta t_{\mathrm{ret}} \rangle \leq \frac{\pi}{\Omega_{\mathrm{Nyquist}}} \ll \frac{\pi}{\omega_{\mathrm{max}}} \tag{4.6}$$

It still needs to be shown that it is always possible to find a suitable time step Δt for the particle simulation, such that the Nyquist frequency $\Omega_{\mathrm{Nyquist}}$ is greater than the maximum frequency ω_{max} of a band-limited radiation signal: Since the differentials $\mathrm{d}\,t$ and $\mathrm{d}\,t_{\mathrm{ret}}$ are related by Eq. 4.4, this is also true for any finite time step.

$$\frac{\Delta t_{\mathrm{ret}}}{\Delta t} \approx 1 - \vec{\beta} \cdot \vec{n} \tag{4.7}$$

By using the definition of the Nyquist frequency (Eq. A.12), one obtains the relation between the maximum frequency that can be correctly calculated, $\Omega_{\mathrm{Nyquist}}$, and the time step of the simulation Δt.

$$\Delta t = \frac{\pi}{\Omega_{\mathrm{Nyquist}} \cdot \left(1 - \vec{\beta} \cdot \vec{n}\right)} \tag{4.8}$$

As always, higher frequencies require a finer sampling rate. However, the relativistic effect caused by a charge moving towards an observer positioned in direction $\vec{n}$ reduces the required sampling rate since $(1 - \vec{\beta} \cdot \vec{n})$ is getting small and therefore the necessary sampling step width Δt is increased. Vice versa, a particle moving away from an observer located in the direction of $\vec{n}$ needs a higher sampling rate for keeping the same maximally resolvable frequency $\Omega_{\mathrm{Nyquist}}$. But this relativistic effect on the minimal sampling rate is not linear. To resolve the same frequency ω, a particle moving towards an observer at a speed close to the speed of light, c, demands just a long simulation step width Δt, while a particle moving away from an observer with a velocity of approximately c requires a sampling rate of twice that of the stationary charge $1/\Delta t = 2 \cdot \frac{\pi}{\omega}$, because $0 < 1 - \vec{n} \cdot \vec{\beta} < 2$. Therefore, it is always possible to find a particle-simulation step-width Δt to resolve any contributions at frequency ω.
Radiation at high frequencies is commonly caused by particles moving at a high velocity and is only emitted in a narrow cone in the direction of the particles motion. Therefore, a moderate sampling rate is usually enough to evaluate the spectrum at high frequencies correctly. Experience shows that the DFT algorithm can display even frequencies beyond $\Omega^{\mathrm{expected}}_{\mathrm{Nyquist}} = \frac{\pi}{\Delta t}$ but not above $\Omega^{\mathrm{relativistic}}_{\mathrm{Nyquist}} = \frac{\pi}{\Delta t \cdot \left(1 - \vec{\beta} \cdot \vec{n}\right)}$ as long as the sampling due to the simulation time step is precise enough to smoothly sample all particle trajectories.
A problem arises when considering several particles. Some may contribute to high frequencies which might be above the Nyquist frequency for other par-

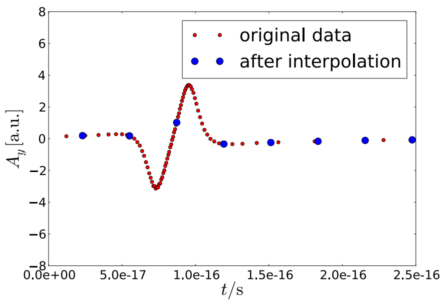
ticles, thus causing a mix of real signals and signals reflected at the Nyquist frequency (see section A.2 for more details on Nyquist reflection). This can be overcome by a method called *Nyquist limiter*, which will be discussed later in section 4.2.2.
The equidistant sampling of the FFT algorithm contrasts with the adaptive sampling in t_{ret} of the DFT algorithm. This can cause problems illustrated in Fig. 4.1. The advantage of an adaptive sampling step width Δt_{ret}, which decreases when the particle moves at high speeds towards the observer and radiates at higher frequencies, is lost when fixing the sampling rate beforehand. In case of a later interpolation using the entire trajectory, a fixed number of sampling points might not sample the actual radiation signal precise enough, therefore resulting in a noisy signal. To illustrate this problem Fig. 4.1 shows actual samplings using *CLARA 2.0* (see section 3.2) and the corresponding radiation spectra. It exhibits the radiation of a single electron initially at rest interacting with a short and highly intense laser pulse with intensity $a_0 = 4.4$.
The red dots represent the non-equidistant signal calculated, while the blue dots depict the interpolation onto an equidistant grid. The upper example uses as many grid points for the equidistant sampling of the retarded time as for the non-equidistant signal, while the lower plot is calculated using ten times more equidistant interpolation points. The non-equidistant sampling can resolve the fast change of the real amplitude by decreasing the step width locally, while the first equidistant sampling is not able to resolve the peak.
In the case of a posterior interpolation onto an equidistant grid, the sampling step width can be set to

$$\Delta t_{\mathrm{ret}} = \frac{t_{\mathrm{ret}}^{\mathrm{last}} - t_{\mathrm{ret}}^{\mathrm{first}}}{N_{\mathrm{FFT}}} = \frac{\pi}{\Omega_{\mathrm{Nyquist}}} \quad . \tag{4.9}$$

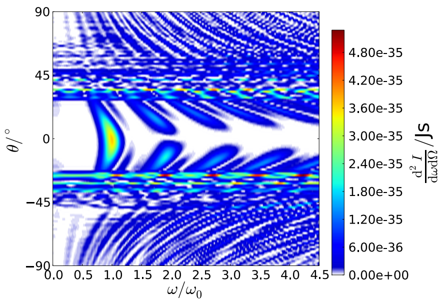
If the band limitation of the emitted radiation is below the Nyquist frequency, $\omega_{\mathrm{max}} \leq \Omega_{\mathrm{Nyquist}}$, the signal can be reproduced, if not, the spectra will contain Nyquist-reflected noise as in Fig.4.1 (1b).
With a predefined grid in t_{ret}, estimates about the first and last retarded time are additionally needed. An implementation of an algorithm to determine the upper and lower limits is described in section 3.3.4. The same problems of resolving frequencies occur for predefined grids in t_{ret} and need to be considered before stetting up any simulation.

4.1.4 Heuristic arguments on the band-limitation of classical radiation

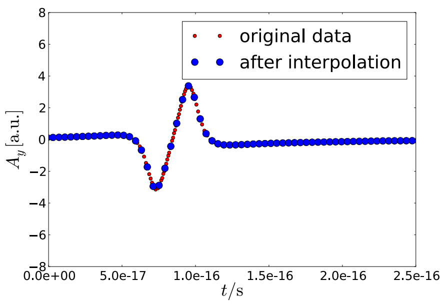
Discrete sampling can only resolve frequencies up to $\Omega_{\mathrm{Nyquist}}$. Therefore, a band-limited radiation needs to be assumed, when doing any numerical Fourier transform. The question is whether this is a legitimate assumption



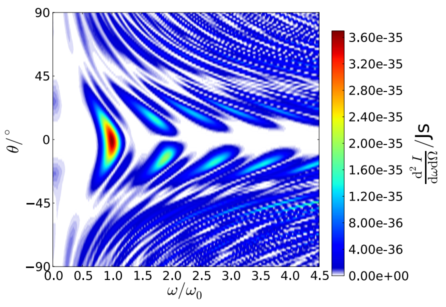
(1a)



(1b)



(2a)



(2b)

Figure 4.1: Illustration of numerical problems occurring when using equidistant sampling in t_{ret}: Picture (1a) and (2a) show part of the signal at $\theta = 30°$ plotted over the retarded time. The red dots illustrate the original signal. The blue dots represent the signal after interpolation onto an equidistant grid. The original data samples peaks with a smaller step width than used by the interpolation while resolving the rest of the signal less precise (not shown). The equidistant sampling (1a) has the same number of sampling points as the original signal but is not able to describe the original signal correctly. (1b) consists of 10 times more sampling points and can replicate the original signal. The resulting spectra are shown in (1b) and (2b). In (1b) a noise signal is clearly visible right where the sampling rate is too low. With a higher sampling rate, the noise disappears (2b).
It is important to mention that this example was created with a too large simulation time step in order to demonstrate the problem.

for arbitrary cases. The emitted electromagnetic waves are caused by accelerated charges and are carrying away energy from the charge and thereby decelerating it. Since the energy of the particle is limited, the emitted radiation is limited in energy as well. Heuristically, the signal must be band-limited since following the Planck-Einstein-equation,

$$E_{\text{photon}} = \hbar \cdot \omega < E_{\text{electron}} \quad , \tag{4.10}$$

there exists a maximum frequency an emitted photon can assume. This argument has to be handled with care. Up to this point, all calculations have been based on classical electrodynamics. In classical electrodynamics, the energy of an electromagnetic wave only depends on its amplitude and not on its frequency. Radiation can in principle assume infinitely high frequencies. An example of classically non-band-limited radiation is the Rayleigh-Jeans description of black-body radiation which results in a diverging spectrum for high frequencies [26] even if the energy of the emitter is limited. This is of course a case where the classical approach fails.
Despite that, in a case correctly described by classical electrodynamics, the quantum mechanical picture is also valid. Therefore, the main peaks of the radiation spectrum need to be band-limited. This does not mean that all classical spectra will drop off above a certain frequency. A spectrum will most likely drop off smoothly around the actual peaks in frequency space. Therefore, the classically calculated radiation is, strictly speaking, not band-limited. Signals above Ω_{Nyquist} will be back-reflected to lower frequencies. As the drop-off is usually fast, these reflected signals, being several orders of magnitudes weaker than the main signals, will only contribute insignificantly to the correct signal and can be ignored.
When entering the realms of quantum physics, not only the band-limiting arguments falls but also all equations used here are invalid. This needs to be kept in mind when using the codes presented.

4.1.5 Increasing spectral resolution by zero-padding

It is not only important to ensure that the highest frequencies can be analyzed correctly by choosing a high sampling rate, but it is also necessary to resolve the calculated spectra with sufficient sampling points to represent the structure of the spectra in detail. When using a DFT method, any frequency can be analyzed. Therefore, the resolution of the spectra can be chosen by the user. When using the FFT method, the spectral resolution $\Delta\omega$ depends on the numbers of sampling points of the original time-domain signal N_{FFT} and the highest frequency calculated in the fast Fourier algorithm, $2 \cdot \Omega_{\text{Nyquist}}$ (see appendix A.3). Since the maximally resolvable frequency depends inversely on the sampling period of the time domain Δt, the spectral resolution scales inversely with the total signal time T.

$$\Delta\omega = \frac{2\Omega_{\text{Nyquist}}}{N_{\text{FFT}}} = \frac{2\pi}{\Delta t \cdot N_{\text{FFT}}} = \frac{2\pi}{T} \tag{4.11}$$

This causes short signals to have a low frequency resolution and can cause peaks or sidelobes to not be resolved correctly. This is illustrated in Fig. 4.2. In this plot, the spectrum of an undulator with only 10 periods of length and a sampling of 40 sampling points per oscillation period is shown in red. No sidelobes are resolved. This problem can be solved correctly by convolving

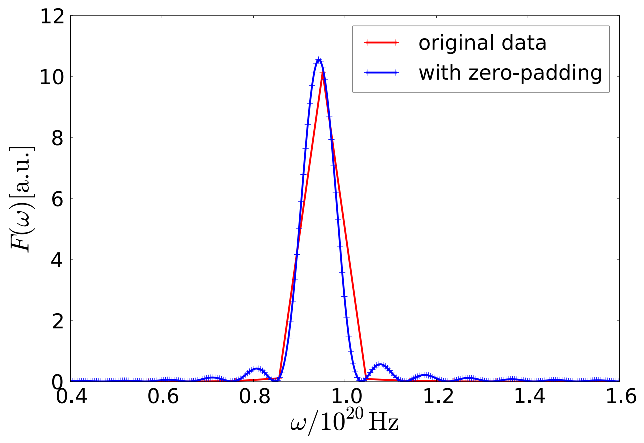


Figure 4.2: Results of zero-padding: undulator spectrum for 10 period long undulator, electron energy of $\gamma = 100$ and 400 sampling points. In red the original spectra calculated using a FFT algorithm is shown. In blue, the spectra of the same signal, but 20 times longer and filled with zeros ("zero-padded") is plotted. The sidelobes could not be resolved correctly with the unmodified signal.

the spectrum with a sinc function [27]. An alternative numerical method is the so-called *zero-padding*. By adding zero-values to the end of the original time-domain signal, the resolution increases, while the added zeros do not contribute to the Fourier transform. This increases N_{FFT} without effecting the spectral amplitudes $\mathcal{F}(\omega)$. In Fig. 4.2, the zero-padded signal is plotted with a blue line. The modified time domain signal is 20 times longer and filled with zeros at the end. This allows to resolve the sidelobes of the main peak correctly.

4.2 Algorithms for many-particle radiation calculations

This section will cover different aspects of calculating the total radiation of multiple particles. It will describe the differences between coherent and incoherent radiation and will discuss the resulting consequences for numerical solutions.

4.2.1 Aspects of coherent and incoherent radiation

When considering the radiation emitted from multiple particles, Eq. 2.44 needs to be adjusted. The phase relation between the emitted radiation needs to be taken into account in order to combine spectra from N_p particles [6]. This can be achieved by adding all complex amplitudes before calculating the square of the absolute values in Eq. 2.44.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\Omega \mathrm{d}\omega} = \frac{q^2}{16 \cdot \pi^3 \cdot \varepsilon_0 \cdot c} \cdot \left| \sum_{k=1}^{N_\mathrm{p}} \int \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta}_k \right) \times \dot{\vec{\beta}}_k \right]}{\left(1 - \vec{\beta}_k \cdot \vec{n} \right)^2} \cdot e^{i\omega \left(t - \frac{\vec{n}\vec{R}_k}{c} \right)} \mathrm{d}t \right|^2 \tag{4.12}$$

The sum over all particles and the integration over time can be interchanged. This is especially useful in case there is not enough memory to store the values of all integrals for every single particle.
However, phase differences lead to complicated interference effects if they vary strongly between all particles. On the other hand, particles oscillating similarly with only small phase differences, cause homogeneous and intense radiation. In the latter case, the total emitted intensity scales with $\sim N_p^2$. In electrodynamics, the first case is usually refereed to be *incoherent* while the second is called *coherent* radiation.
It is important to notice the difference between this definition and the one usually found in optics. There, one considers two light beams as coherent if they have a fixed phase relation and can create interference patterns if they overlap, and as incoherent if they have no such relation. This definition is less strict than the former and mixing both definitions can lead to confusion. To avoid this, only the electrodynamics convention will be used in this text.
Most natural radiation sources are incoherent. There are usually many individual emitters involved and the interference patterns smooth out statistically. But when simulating radiation from particles, one usually can only consider a small sample of all emitters. Therefore, incoherent radiation simulated using only a few particles will result in an overestimation of the interference pattern caused by the random sample of emitters.

Having as many simulated particles as real particles would avoid this problem, but this is computationally not feasible. To circumvent this problem, Eq. 4.13 was used for obviously incoherent radiation sources. It is similar to Eq. 4.12 except that the sum over all particles and calculating the absolute square have been interchanged. Thus, the phase relation between different particles is ignored which is a good approximation for incoherent radiation. The following section shall motivate this choice.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\Omega \mathrm{d}\omega} = \frac{q^2}{16 \cdot \pi^3 \cdot \varepsilon_0 \cdot c} \cdot \sum_{k=1}^{N_\mathrm{p}} \left| \int \frac{\vec{n} \times \left[\left(\vec{n} - \vec{\beta}_k \right) \times \dot{\vec{\beta}}_k \right]}{\left(1 - \vec{\beta}_k \cdot \vec{n} \right)^2} \cdot \mathrm{e}^{\mathrm{i}\omega \left(t - \frac{\vec{n}\vec{R}_k}{c} \right)} \mathrm{d}t \right|^2 \tag{4.13}$$

For incoherent radiation, the phase relation between all particles differs largely. If the number of particles is huge, the actual phase relation should

play no role anymore.
This can be illustrated easily by assuming there are N_p particles radiating similarly except for an individual phase shift φ_k, which could be caused by a local displacement of the particles. The radiation spectra can be calculated using the following equations.

$$\frac{\mathrm{d}^2 I}{\mathrm{d}\Omega \mathrm{d}\omega} = \left| \int \sum_{k=1}^{N_\mathrm{p}} \vec{A}_k(t) \mathrm{e}^{\mathrm{i}\omega(t+\varphi_k)} \,\mathrm{d}\, t \right|^2 \tag{4.14}$$

$$= \left| \sum_{k=1}^{N_\mathrm{p}} \mathrm{e}^{\mathrm{i}\omega\varphi_k} \int \vec{A}_k(t) \mathrm{e}^{\mathrm{i}\omega t} \,\mathrm{d}\, t \right|^2 \tag{4.15}$$

$$= \left| \sum_{k=1}^{N_\mathrm{p}} \mathrm{e}^{\mathrm{i}\omega\varphi_k} \vec{\mathcal{F}}_k(\omega) \right|^2 \approx \mathcal{F}^2(\omega) \left| \sum_{k=1}^{N_\mathrm{p}} \mathrm{e}^{\mathrm{i}\omega\varphi_k} \right|^2 \tag{4.16}$$

In these formulae, $\vec{A}(t)$ describes the complex radiation amplitude in the time domain (Eq. 2.45). This would be equal to the vector part in Eq. 4.3. By assuming the phase shift to be constant over time, it can be separated from the Fourier transform. After calculating the spectra $\mathcal{F}_k(\omega)$ for every particle via Fourier transform, a phase shift for every particle's spectra remains (see also *translation in time* in appendix A.1). Suppose the spectra of all particles are similar $\mathcal{F}_k \approx \mathcal{F}$, which is true in the case of spatial displacement, they can be separated from the summation over all particles, leaving only a sum over the complex unit circle. The phase of the complex exponential function still depends on ω, but the frequency ω can be considered a constant parameter for this calculation. By substituting $\omega\varphi_k \to \phi_k$, the sum

$$\left| \sum_{k=1}^{N_\mathrm{p}} \mathrm{e}^{\mathrm{i}\phi_k} \right|^2 \tag{4.17}$$

remains to be solved in Eq. 4.16. For simplicity, the phase difference ϕ_k can be assumed to follow a Gaussian distribution with a standard deviation of σ_ϕ. When evaluating the expression in Eq. 4.17, the sum results to N_p^2 for standard deviations smaller than $\sigma_\phi < 2\pi$, equal to a spatial standard deviation smaller than the emitted wavelength. On the other hand, the sum results to N_p for standard deviations larger than $\sigma_\phi > 2\pi$, meaning a spatial distribution larger than the emitted wavelength.
To illustrate these findings, the results of the sum are plotted over different standard deviations σ_ϕ in Fig. 4.3 for different particle numbers.
For $\sigma < 2\pi$, the sum evaluates to N_p^2. The error of this mean value as well as the standard deviation are small. This means that a single evaluation of the sum will most likely result in N_p^2. In contrast, for $\sigma > 2\pi$ the sum

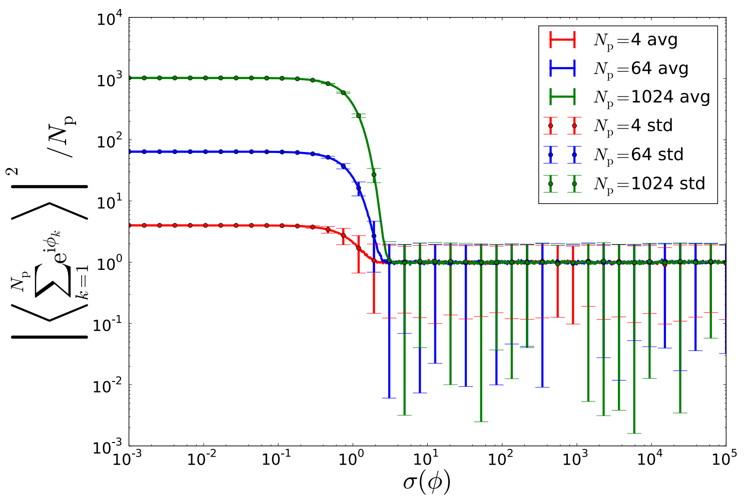


Figure 4.3: Monte Carlo simulation of the complex sum (Eq. 4.17): the complex phase ϕ follows a normal distribution with standard deviation σ. The average result (avg) of the sum (over 1000 simulations) is drawn as line for three different numbers of particles N_p, including error bars for the mean values. The errors of the mean values σ/N are small and not visible in the plot. The visible bars represent the standard deviation (std) σ of the result of the sum. The y-axis is normalized to N_p.

will in average evaluate to N_p. The error of this mean value is small but the standard deviation is relatively large. That means that the results of a single evaluation of the sum, as in the case of a single radiation simulation, have a wide spread.
For broad particle distributions, the total intensity is on average proportional to the number of particles N_p. This scaling is exactly represented by Eq. 4.13. Particle distributions with a spatial extent greater than a wavelength are common in experiments, which justifies the use of this formula. However, if a large number of particles radiates coherently, Eq. 4.13 will result in an intensity scaling proportional to N_p and therefore will underestimate the radiated energy. But using this equation is often the only feasible way to calculate radiation spectra without over evaluating the effects of interference. In laser particle interactions, both coherent and incoherent radiation play a role [28]. Simulating these cases with Eq. 4.13, referred to as *incoherent radiation method* from now on, will cause the coherent part of the radiation to be too low, while using Eq. 4.12, the *coherent radiation method* for short, will be influenced by the random particle selection. The influence of choosing a small subset of random particles representing all particles can be seen in the large "error bars" in Fig. 4.3 representing the standard deviation of the 1000 samples used to calculate the average. Even if the standard deviation

of the average result of Eq. 4.17 converges to N_p, a single evaluation of the sum can have a wide spread. This spread is not decreased by increasing the number of particles. The simulated spectra will always look noisy for incoherent radiation. In appendix C, it is shown that it is still possible to regain all informations from the noisy spectral signal.

4.2.2 The Nyquist limiter

The Nyquist limiter, introduced in this section, solves the problem of mixing low and high energy spectra in many-particle simulations.
On the one hand, a simulation contains electrons with $\gamma \approx 1$. Following Eq. 4.8, their radiation amplitudes are sampled approximately with the step width of the simulation Δt_sim. On the other hand, there are electrons with velocities close to the speed of light. They radiate mainly into their direction of flight. Towards this direction, the sampling rate is increased, allowing to resolve higher radiation frequencies. A problem arises, if these fast electrons radiate at a frequency above $\frac{\pi}{\Delta t_\mathrm{sim}}$. It is easy to resolve this frequency for the fast electrons, but for the other electrons such a frequency lies above their Nyquist frequency Ω_Nyquist. Therefore, their spectral values at this frequency are wrong and only consists of reflections of the low frequency spectrum (see section A.2). Now, the radiation spectra of all electrons need to be combined. When calculating the total spectrum, the result will be disturbed at higher frequencies by the erroneous contributions of the low-energy electrons. It is not possible to know each particle's trajectory beforehand. Therefore, one cannot exclude the low energy electrons at the initialization of the simulation. It is necessary to reduce the step width of the simulation, Δt, until the Nyquist frequency of all particles is higher than the frequencies to observe in order to avoid the wrong contributions. This approach, however, would be not feasible with regard to the time required by the particle simulation. In order to solve this problem, an algorithm named *Nyquist limiter* was developed to filter out reflections. Before adding any complex amplitudes to the total spectrum at frequency ω, the following equation needs to be checked

$$\omega < \frac{\pi \cdot \left(1 - \vec{\beta} \cdot \vec{n}\right)}{\Delta t} \quad , \tag{4.18}$$

with β the normalized speed of the electron and $\vec{n}$ the normalized vector pointing in the direction of observation. If the inequality is true, the particle's amplitude will be added to the sum over all particles contributing the frequency ω. If not, the calculated contribution to the spectra is discarded. The algorithm uses the limit on reproducible frequencies given by the Nyquist-Shannon theorem (Eq. A.12) and combines it with the restrictions on non-equidistant sampling [25]. It requires that the particle's current sampling in retarded time Δt^p_ret is precise enough to resolve the frequency contribution to compute $\mathcal{F}(\omega)$. By only letting those amplitudes contribute which have a sampling rate high enough for a certain frequency ω, the average sampling rate will always be high enough to correctly sample ω. This is a simple but efficient method to calculate spectra with many particles which have a wide range of energies. It is not suitable for signals close to the Nyquist frequency, but such cases can be avoided in any case by reducing the time step of the simulation.

5 | Code verification and results

All codes developed for this thesis have been verified by comparing their results with well-known analytical solutions. This rigorous testing was needed in order to gain confidence in the results of future simulations using these codes. All tests have proven that the codes produce correct results. Therefore, the time domain code and *CLARA 2.0* are now applied by different research groups and the Radiation Analyzer is going to simulate the emissions of a full scale plasma simulation. The tests and applications are presented in this subsequent chapter.

5.1 Verification of the time domain code

The time domain code computes the electric and magnetic fields at arbitrary positions and times. This is especially useful for simulating electromagnetic fields at a high spatial resolution and has applications in simulating the influence of the field on surrounding particles.

5.1.1 Verification by simulating fields of non-accelerated charges

The most simple test regards the electric field of a particle at rest. The simulated electric field on a spatial grid agrees very well with the analytical Coulomb field (Eq. 2.1). Since the simulated charge does not move, the magnetic field should be zero everywhere. However, the simulated magnetic field does not vanish on all grid points. The magnetic field values fluctuate around zero but their magnitude always stays more than fifteen orders of magnitude below $|\vec{E}|/c$. These nonzero magnetic field values are caused by numerical errors attributed to the limited precision of floating point numbers. When calculating $\vec{E} \times \vec{n}$, this product should vanish for non-moving particles because $\vec{E} \parallel \vec{n}$ and therefore $\vec{E} \times \vec{n} = 0$. The cross product implemented in the simulation relies on floating point arithmetics and therefore the result being zero is not guaranteed. Since the occurring numerical errors

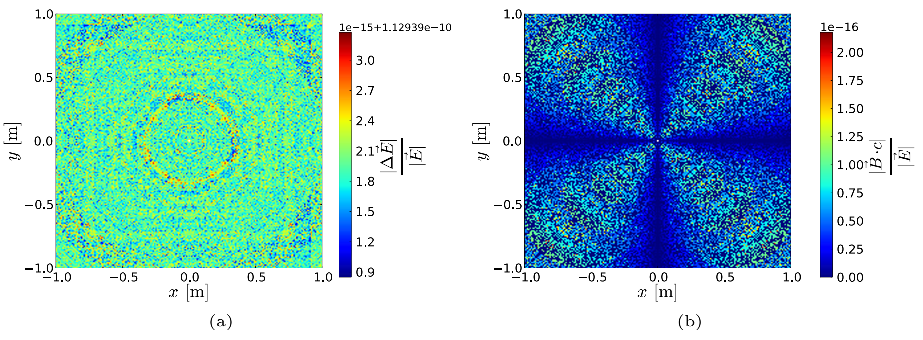


Figure 5.1: The physical case investigated is an electron at rest centered at $x = 0$, $y = 0$. Plot (a) shows the difference between the analytical solution and the simulated electric field. The relative error is in the order of 10^{-10} (color values $\cdot 10^{-15} + 1.1 \cdot 10^{-10}$). The numerical errors of the magnetic field are depicted in (b). Since the electron is not moving, the magnetic field should be zero but floating point arithmetics causes magnetic field values in the order of double precision 10^{-15} compared to $\vec{E}/c$.

are extremely small they can certainly be ignored (Fig. 5.1).
Further tests include a particle moving at constant speed. The resulting electric and magnetic fields agree well with the Lorentz-boosted Coulomb field [7] calculated by Eq. 5.1 to Eq. 5.3. For simplicity, the following formulae assume that the fields are evaluated at $t = 0$ and the particle is located at the origin of the coordinate system at that time.

$$\vec{E} = \frac{q}{4\pi\varepsilon_0\sqrt{1-\beta^2}} \cdot \frac{\vec{r}}{\left[\frac{x^2}{1-\beta^2} + y^2 + z^2\right]^{3/2}} \tag{5.1}$$

$$B_y = \frac{\beta}{c} \cdot E_z \tag{5.2}$$

$$B_z = \frac{\beta}{c} \cdot E_y \tag{5.3}$$

These formulae can be derived from Eq. 2.29 and Eq. 2.30 when assuming a charge moving at constant velocity in x-direction. The relative difference between analytical solution and simulated values stays below 10^{-3} (Fig. 5.2). This is quite good for this more complex case.

5.1.2 Verification by simulating synchrotron radiation

Above tests do not include any effects scaling with $\vec{E} \sim \frac{1}{r}$ caused by an acceleration of the charge. In order to test these influences, a particle moving in a circle, similarly to the particle motion in a synchrotron [8], is a good test

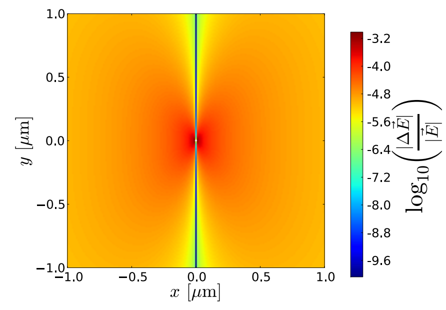


Figure 5.2: The physical case investigated is an electron moving with $\beta = 0.8$ in x-direction. It is centered at $x = 0$, $y = 0$. The plot depicts the relative difference between the analytical solution and the simulated electric fields. Even close to the electron, the relative numerical error of the electric field stays below 10^{-3}.

case since it includes both near and far field and it can be checked against an analytical solution. The resulting fields of the simulation were checked against a semi-analytical solution developed in Mathematica and against an analytical results from [29]. In contrast to the simulation, the semi-analytical solution uses Eq. 2.29 and solves for the retarded time by using the particle's equation of motion. A numerical solver was used to determine t_{ret} for the synchrotron-like motion. This is necessary, since the retarded time can only be described by a transcendental equation [29]. The simulated fields agree perfectly with both the semi-analytical and analytical solution.
A test was performed by assuming a particle following a circular track at velocity $v = \beta \cdot c = 0.9 \cdot c$. Fig. 5.3 depicts the logarithmic magnitude of the electric field with the circular trajectory of the electron drawn as green dots. To quantitatively examine the resulting electric fields, cuts were performed, drawn as blue line in Fig. 5.3 for the data presented in Fig. 5.4. These data cuts were performed for the far ($\vec{E}^{\text{rad}} \sim r^{-1}$, Eq. 2.32) and near field ($\vec{E}^{\text{vel}} \sim r^{-2}$, Eq. 2.31) components of the field separately. The results were checked against the analytical solution implemented in Mathematica as illustrated in Fig. 5.4. The agreement between simulated and semi-analytical data is excellent. The test presented here is one in a series of many tests, chosen because it shows a dominant far field and a relatively large peak width, making it perfectly suited for visualization.
As a final test of the time domain code, synchrotron radiation for an electron with relativistic energy $\gamma = 100$ was simulated and compared against an analytical solution by Chengkun et al. [29]. These analytical calculations are aimed at investigating the influence of coherent radiation inside an electron bunch in a synchrotron, the so-called *coherent synchrotron radiation* (CSR) [30, 31].
Only the longitudinal component of the electric field acts on the electrons

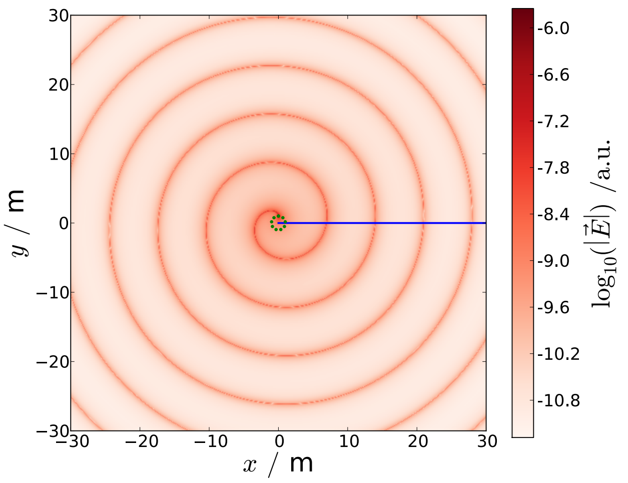


Figure 5.3: Illustration of the synchrotron test used in the validation process of the time domain code: Assumed was a single electron moving clockwise on a circle with radius $r = 1\,\text{m}$ at $\beta = 0.9$. In red, the logarithmic magnitude of the resulting electric field is drawn. Using green dots, the electron's trajectory is illustrated. The blue line represent a cut analyzed in detail and shown against analytical data in Fig. 5.4

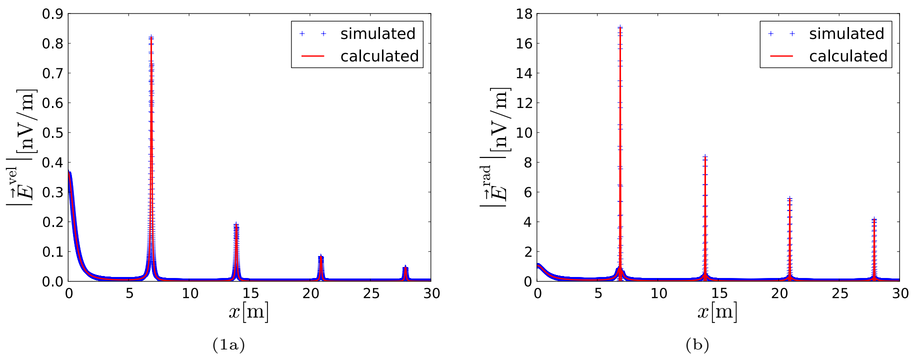


Figure 5.4: This is a plot of the electric fields from the cut along the blue line in Fig. 5.3. Blue dots represent the simulated data. The red line is the analytical solution calculated using Mathematica. The electric field has been decomposed to a far and a near field component. Plot (a) shows the near field component scaling with $\vec{E}^{\text{vel}} \sim r^{-2}$. Plot (b) shows the far field component scaling with $\vec{E}^{\text{rad}} \sim r^{-1}$

and can thereby influence the entire electron bunch. Therefore, the longitudinal component of the far field $\vec{E}^{\text{rad}}$ was simulated and compared with the analytical results of Chengkun. The test case included a single electron with

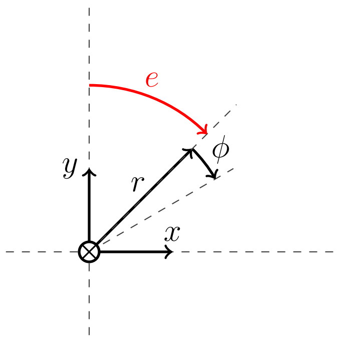


Figure 5.5: Labels for the synchrotron plots: The electron trajectory is drawn in red. The angle ϕ and the distance r define the position of the field calculated.

$\gamma = 100$ moving on a circular trajectory with a radius of 1 m (Fig. 5.5). The longitudinal electric radiation field $\vec{E}^{\mathrm{rad}} \cdot \vec{e}_\phi$ is calculated around the particle (Fig. 5.6). The results show an excellent agreement between the analytical predictions and the simulated data.

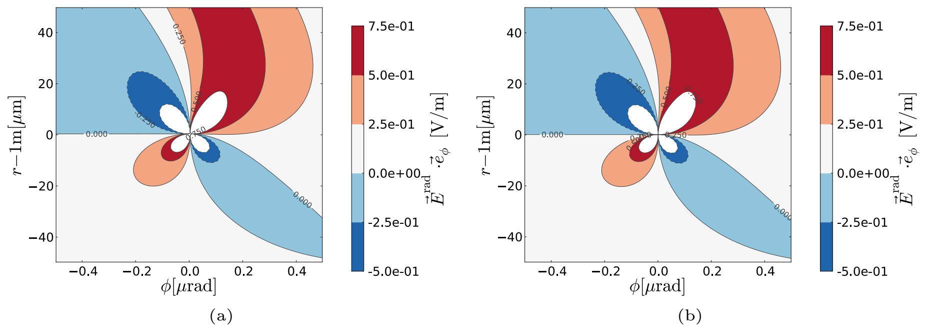


Figure 5.6: Both graphs show a contour plot of the radiation or far field component of the electric field $\vec{E}^{\mathrm{rad}}$ of a single electron with $\gamma = 100$ running in a synchrotron with a radius of $r = 1\,\mathrm{m}$. Only the tangential component E_ϕ of the radiation field $\vec{E}^{\mathrm{rad}}$ is drawn. The electron is located in the middle of the plots. Plot (a) shows the results from the time domain code while (b) shows the analytical results. The right graph is taken from [29].

The left plot shows the simulated field while the right plot displays the analytical solution from [29]. A good agreement on the clover leaf pattern can be seen.

5.1.3 Applications to Coherent Synchrotron Radiation

The concept of coherent radiation of electrons running in a circle, the Coherent Synchrotron Radiation (CSR), was first developed in 1971 to explain the high-intensity radiation emitted by pulsars [30]. It was first believed that these effects were not realizable in circular accelerators and storage rings, but in 1990, a Japanese collaboration was able to realize coherent radiation in synchrotrons [31].
If the electron bunch size is smaller than the emitted wavelength, the radiated power is increased dramatically. This is due to the constructive interference occurring in the electron bunch. In the case of coherent synchrotron radiation (CSR) the total emitted power scales quadratically with the beam current. This allows to create highly intense synchrotron radiation using only a small bunch charge.
However, at such small bunch sizes needed to create CRS, the electrons interact with each other through their electromagnetic field. The electrons can be pushed by the longitudinal component of the other electrons' electric field. This feedback of the field can cause the beam emittance to grow and the bunch to become unstable [29], resulting in a destruction of the desired coherent radiation. To understand the effects of these interactions between the electrons and their fields a "self-consistent dynamical simulation" [29] is needed but such a simulation seems to be not feasible. Therefore, several one-dimensional CSR models were developed to take the effects of charge distribution into account.
The group around Chengkun Huang et al. at Los Alamos National Laboratory now developed a two-dimensional analytical model of CSR to more accurately take into account effects by the beam's shape and the spatial dependence of the electric and magnetic fields [29]. They also developed an analytical approximation to calculate the occurring electromagnetic fields caused by the electrons and estimated effects of a Gaussian beam shape by using convolution. However, they assumed the single particle fields to be invariant under small radial displacements and rotations [29]. This approximation is only valid for small bunch sizes, but to take different integration kernels into account, a simulation of the fields is required.
Using the time domain code allows to simulate the radiation of any particle distribution and to take into account beam parameters not covered by the 2D model. This allows to determine the limits of the 2D model and to simulate the coherent synchrotron radiation more realistically.

At the moment, the first tests have been performed using the analytical trajectories provided by Chengkun. A further collaboration is planned.

5.2 The frequency domain code

The frequency domain codes, *CLARA 2.0* and the *Radiation Analyzer*, are able to calculate the angular resolved spectra far away from the particle emitting it. This is similar to experiments where one usually observes radiation a distance away from its point of origin. Both codes allow to predict the radiation observed in experiments and are well suited to search for characteristic radiation signatures. The calculated spectra can be used for diagnostics concerning the experimental setup but also to predict radiation occurring in a wide range of physical scenarios ranging from gamma-ray burst emission [32] to laser wakefield acceleration [5]. A variety of tests have been performed to ensure that the simulations produce correct results. These will be presented together with first applications.

5.2.1 Simulating undulator radiation

Electrons passing through an undulator produce a clean spectral signature. Their trajectories can be calculated without the need of simulations. Therefore, computing the radiation produced in an undulator is an ideal test case for validating any simulation of spectrally resolved radiation. Since the trajectory is known analytically, any discrepancies between simulation and the analytical solution in the calculated spectra can be ascribed to errors in the radiation code. Additionally, any analytically known trajectory allows for arbitrary sampling of the trajectories which is useful for testing programs that highly depend on the sampling rate.
In order to validate the cluster-based frequency domain code, *CLARA 2.0*, a method for generating traces of electrons in undulators has been developed. In this method a variety of parameters such as electron energy, magnetic field strength or the undulator wavelength could be adjusted to simulate different undulators and thereby check different aspects of the radiation code. The simulated spectra (Fig. 5.7) were compared to the analytical solutions. The only differences were found in the amplitude of the sidelobes of the spectra but turned out to be insignificant. As part of the tests, the electron energy was increased steadily to check for numerical instabilities caused by the limited precision of $\vec{\beta}$ at those energies. At electron energies around $\gamma > 3 \cdot 10^6$, the first discrepancies appeared due to the limited precision of double precision floating point numbers.
Since electrons in laser plasma interaction can currently reach energies "only" up to GeV-level [33], the above limit does not pose a problem for applying the code to this physical case.
For the *Radiation Analyzer* included in PIConGPU, the restrictions due to the limited precision of floating point numbers are far more severe. PIConGPU was developed on the first generation of CUDA-capable graphics cards where performing a double-precision operation was far slower than the single-precision

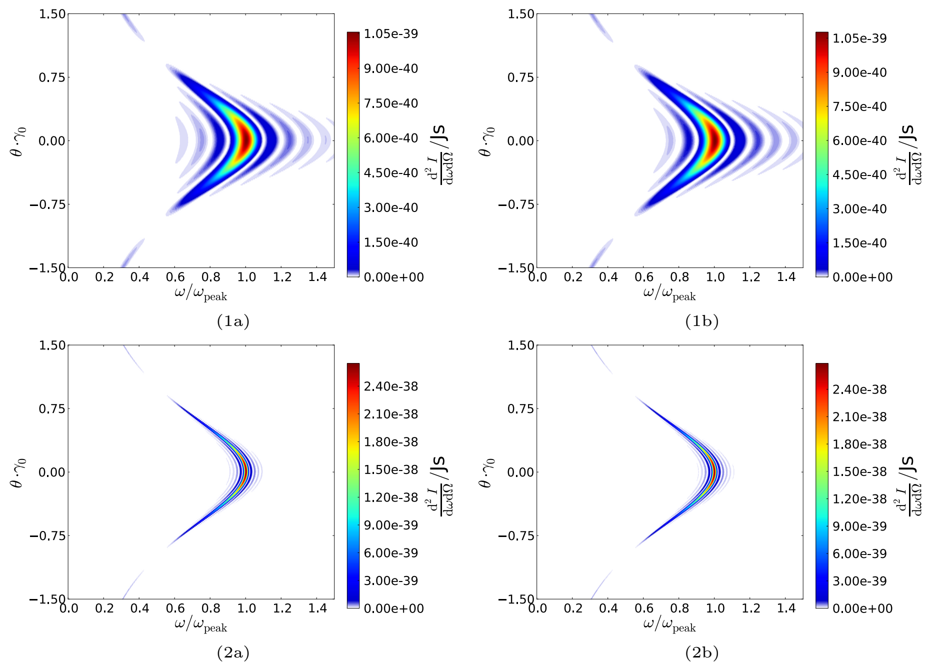


Figure 5.7: The radiation of an electron $\gamma = 100$ in an undulator $\lambda = 800\,\mathrm{nm}$, $K = 3.7 \cdot 10^{-5}$ is shown. The plots on the left side are simulated spectra (a), while the plots on the right side are theoretical predictions analogous to Fig. 2.10. The first plots (1) shows the spectra of an 10 period long undulator, while (2) shows a 50 period long undulator.

equivalent. Therefore, all PIConGPU-internal values were stored in single precision. This has been kept this way since the speed difference between single and double precision floating point operations is still around three on today's graphic cards. Hence, single precision floating point values are essential to run PIConGPU efficiently.
Thus, all particle data provided by PIConGPU have only a precision of about 7 decimal digits. This leads to severe problems when using $\beta(\gamma)$, especially in the common case of a $1 - \vec{\beta} \cdot \vec{n}$ close to zero. For certain $\vec{n}$ and $\vec{\beta}$, this results in one minus a value close to one. The precision of the remaining difference is on the order of 10^{-7}. Thus, these differences can be extremely imprecise. For example at $\gamma \approx 700$, the relative error of this difference will be about 10%.
To avoid these numerical issues, a trick was developed: Under certain conditions, a different method is used to calculate $1 - \vec{\beta} \cdot \vec{n} = 1 - \beta \cdot \cos\alpha$.
By using the relativistic energy factor γ, the magnitude of $\vec{\beta}$ can always be calculated with

$$\beta = \sqrt{1 - \frac{1}{\gamma^2}} \quad . \tag{5.4}$$

This allows to express $1 - \vec{\beta} \cdot \vec{n}$ differently.

$$1 - \vec{\beta} \cdot \vec{n} = 1 - \beta \cdot \cos\alpha = 1 - \cos\alpha \cdot \sqrt{1 - \frac{1}{\gamma^2}} \tag{5.5}$$

For small values of γ^{-2}, the square-root can be approximated by its Taylor series $\sqrt{1-x} \approx 1 - \frac{1}{2}x - \frac{1}{8}x^2 - \frac{1}{16}x^3 - \cdots$. This allows to avoid to subtract one and approximately one.

$$1 - \vec{\beta} \cdot \vec{n} \approx \cos\alpha \cdot \left(\frac{1}{2}\frac{1}{\gamma^2} + \frac{1}{8}\frac{1}{\gamma^4} + \frac{1}{16}\frac{1}{\gamma^6} + \cdots \right) + (1 - \cos\alpha) \tag{5.6}$$

The $(1 - \cos\alpha)$ part of the sum is still imprecise, but the Taylor developed part of the sum allows to calculate values to a high precision. This alternative calculation is only chosen if the particle energy fulfills $\frac{1}{\gamma^2} < \varepsilon_c = 0.18$. The number $\varepsilon_c = 0.18$ was chosen because around this value, the error of the Taylor series in fifth order equals the error of calculating the value without the method developed.

5.2.2 Simulating nonlinear Thomson scattering

Beyond the basic tests presented above, it is also important to check if radiation from nonlinear electron dynamics can be correctly predicted by the developed code or not. Nonlinear particle oscillations are common for interactions of matter with highly intense lasers. A simulation not able to handle these cases would not be suitable for research in this field.
As mentioned in section 2.2, nonlinear Thomson scattering is a common source of radiation occurring in laser-particle collisions. The nonlinear oscillation of the particles caused by the high electromagnetic fields generates a complex radiation pattern with higher harmonics depending on the laser intensity and the particle dynamics. As pointed out before, a detailed description of the electron trajectories and of the resulting radiation can be found in Esarey et al. [14]. The paper describes two distinct approximations: a relativistic electron moving towards the laser and an almost stationary electron wiggling around a fixed position. It covers both cases of linear and circular laser polarization. The amplitude of the laser is approximated by a rectangularly shaped envelope. This implies a negligible ponderomotive forces.
For testing the frequency-domain far-field simulation, the solution of the electron's trajectory, presented in the Esarey et al. paper, is used. Additionally, reproducing the trajectory with PIConGPU is an ideal evaluation of the particle pusher algorithm.
In order to realize the first case described by Esarey et al. [14] in PIConGPU, the laser was set to peak intensities of $a_0 = 0.5$, 1.0 and 1.5 with a wavelength of $\lambda_0 = 800\,\mathrm{nm}$. Similar to the Esarey paper, seven oscillation periods of the

electron were chosen to contribute to the radiation calculation. The resulting radiation is shown in Fig. 5.8. On the abscissa, the frequency, normalized to the blue-shifted first harmonic peak without photon drag, is plotted, while the ordinate is associated with the observation angle, normalized to the relativistic compression of the radiation cone. Both theoretical and simulated spectra are plotted together to better exhibit the differences. Comparing these simulated results with the analytical results, only minor differences can be detected. Most obvious are slightly different curvatures of the sidelobes arising from the numerical Fourier transform implemented in the simulation. However, the overall agreement is excellent.
The simulated spectra for the almost stationary electron are shown in Fig. 5.9. Again, the theoretical and simulated spectra are plotted together. Since the electron does not drift, the radiation caused by its wiggling is not Doppler shifted. Therefore, the frequency-axis was normalized to the laser frequency ω_0. The observation angle θ describes the angle between the electron momentum and the observer in the polarization plane of the laser and is plotted on the y-axis. The missing drift also results in no photon drag. Since the electron is, on average, at rest, the radiation is not Lorentz boosted and the spectral peaks stay at multiples of the laser frequency for all observation directions.
As with the previous test, the simulated radiation agrees very well with the theoretical prediction (Fig. 2.8). Only small differences around the sidelobes are observed.
For both cases described by Esarey et al. [14], a laser with an rectangular amplitude is assumed, but the field grid of a PIC simulation cannot resolve such a laser, since it contains high frequencies. Instead, a super-Gaussian laser profile was implemented. The radiation caused by this laser profile is not correctly predicted by [14]. However, dealing with a simulation allows to selectively switch on and off the radiation calculation during the simulation. This allowed to only calculate the radiation at instances correctly described by the theory. It is possible to let the particle enter the rising field of the laser without considering its radiation, then calculating the emitted radiation while the maximal field amplitude is constant and stopping the calculation again before the downslope of the laser profile. This selection is illustrated in Fig. 5.10(a) by plotting the entire trajectory in blue and the part contributing to the radiation analysis in red.
Before starting with the radiation calculation, the upslope of the laser causes the particle to undergo ponderomotive forces, pushing it slightly in the direction of the laser propagation and thereby changing its speed. For the first test, this is not a problem for the used laser intensities since the effects of the ponderomotive force can be neglected because of the high initial electron momentum. Tests with higher peak laser intensities would require higher initial momenta to compensate for the ponderomotive drift. For the second approximation in [14], this correction is always necessary. Even a

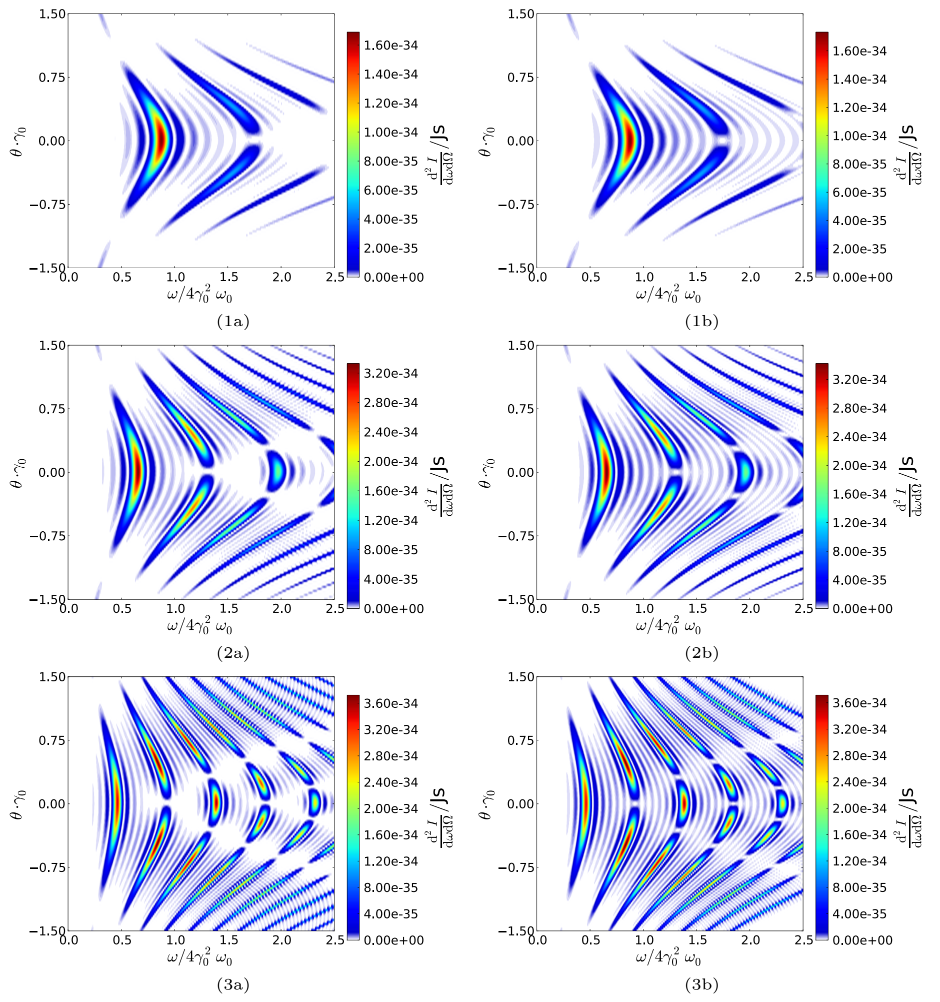
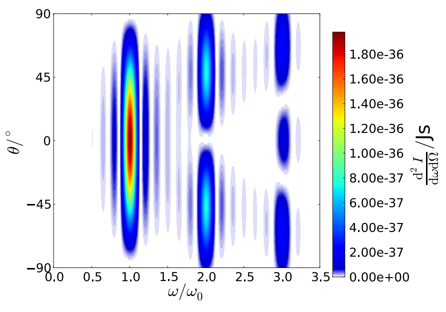
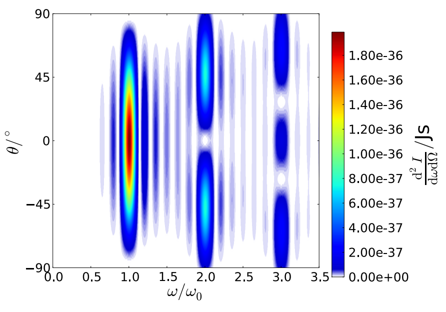


Figure 5.8: Simulated spectra of nonlinear Thomson scattering (a) are compared to analytical solutions (b) analogous to Fig. 2.7. The energy deposition per unit frequency and unit solid angle for different laser strengths of an electron moving towards the laser with $\gamma = 5$ is shown: (1) laser with $a_0 = 0.5$ (2) laser with $a_0 = 1.0$ (3) laser with $a_0 = 1.5$.

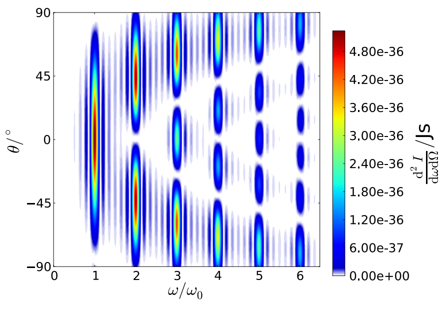
small ponderomotive force will lead to a drifting electron if it was initially at rest. Therefore an initial velocity is required such that the electron is, on average, at rest when it reaches the laser envelop plateau. Using *PIConGPU*, the effect of the ponderomotive force caused by the rising edge of the super-Gaussian pulse were studied. The initial velocities towards the laser that lead to an almost stationary electron at the plateau of the super-Gaussian laser are given below. The values are given for a super-Gaussian laser profile with up- and downslopes that have a standard deviation of one laser



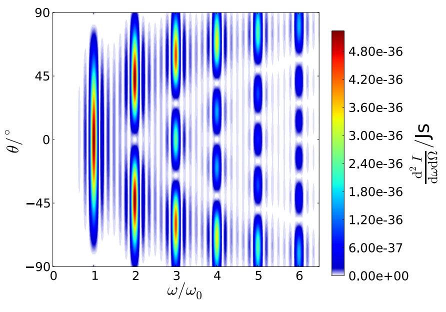
(1a)



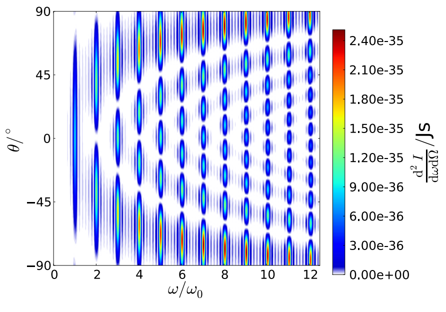
(1b)



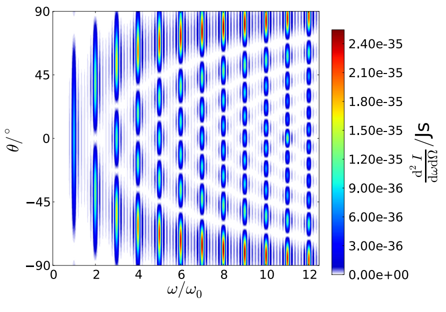
(2a)



(2b)



(3a)



(3b)

Figure 5.9: Simulation of nonlinear Thomson scattering of an electron with no average drift (a) are compared to theoretical predications (b) analogous to Fig. 2.7. The energy deposition per unit frequency and unit solid angle for different laser strengths is plotted: (1) laser with $a_0 = 0.5$ (2) laser with $a_0 = 1.0$ (3) laser with $a_0 = 2.0$.

wavelength $\sigma = \lambda_0$.

$\boldsymbol{a_0}$	$\boldsymbol{\beta}_{\text{Start}}$
0.5	0.058977
1.0	0.1975
2.0	0.5

Since PIConGPU at the time of writing this thesis included only methods to use linearly polarized lasers, the circularly polarized laser cases have been ignored. However, if a circular polarized laser should be included in

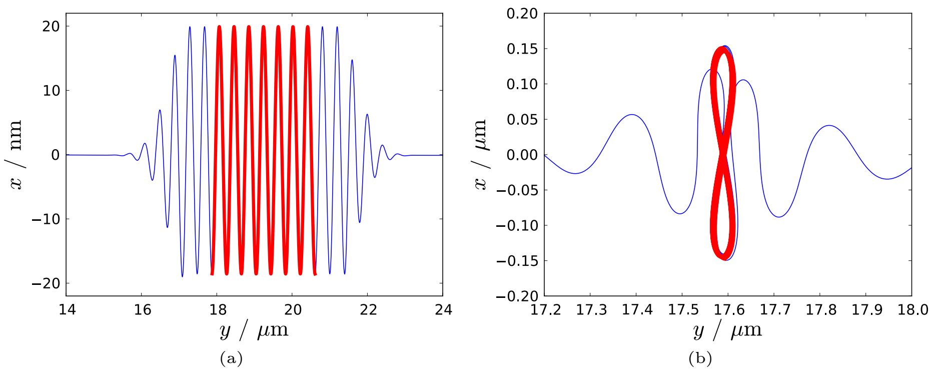


Figure 5.10: Analyzed electron trajectory: Blue represents the electron trajectory while red represents the part of the trajectory contributing to the radiation analysis. In (a) an electron with initial $\gamma = 5$ moves towards a laser $a_0 = 1.5$. In (b) an in average stationary electron $< \vec{\beta} >= 0$ interacts with a laser $a_0 = 2.0$.

PIConGPU, repeating this simulation using circularly polarized laser pulses would pose an ideal test for both the new laser and the radiation code, since several aspects of the laser implementation could be checked by simply simulating the emitted radiation of a single electron.

5.2.3 Simulating radiation to determine the beam emittance at ELBE

The electron accelerator at the Helmholtz Zentrum Dresden Rossendorf, the **E**lectron **L**inac for beams with high **B**rilliance and low **E**mittance, short ELBE, is used for a variety of tasks using electron beams. Its applications range from nuclear experiments to creating highly intense light when electron bunches pass through an undulator or interact with the available high-intensity laser DRACO.
Most of the ELBE beam parameters can be accurately determined, however finding out the beam emittance is a challenging task. More importantly, knowing the beam emittance is vital for different experiments especially for those that use the electron beam as driver of X-ray pules via Thomson scattering.
One possible way to determine the beam emittance at ELBE is to use the Draco laser and to collide the electron beam head-on with the laser beam. This optical undulator (chapter 2.3) emits radiation in the direction of flight of the electron bunch. However, the intensity of the radiation depends strongly on the beam emittance. Since the electrons inside the bunch cross the laser at slightly different angles and energies caused by the bunch emittance, the resulting spectra are smeared out. On axis, this leads to a

wider spectral peak. This results from a symmetric broadening caused by the laser bandwidth and the electron beam energy spread and an asymmetric broadening by the angular spread of the electron momentum [34]. The resulting asymmetry of the spectral peak leads to a negative third standardized moment or skewness that can be used to characterize the electron beam emittances by the emitted spectra [35]. One can predict the radiation and the skewness for a variety of beam parameters by simulating a multitude of beams with different beam emittance and using the simulated electron trajectories together with the frequency-domain code. Comparing those to the experimentally observed spectra will allow to determine the beam emittance.

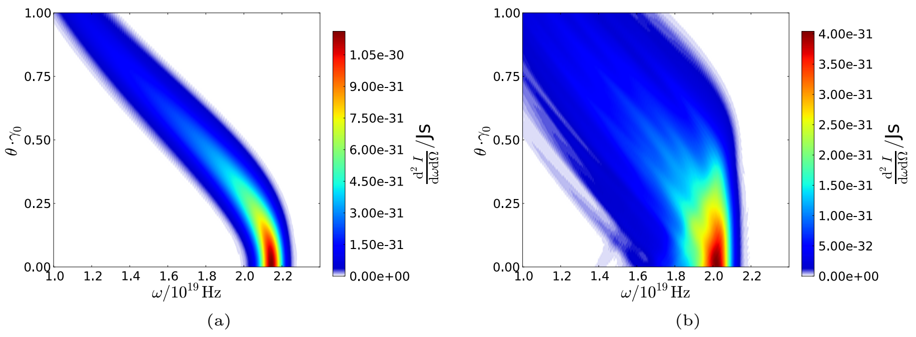


Figure 5.11: Radiation calculation for varying beam emittance: The pictures show two different simulated spectra of an electron beam interacting with the Draco laser at ELBE. The beam is represented by 2000 simulated electrons. The beam in (a) has a beam emittance of $\varepsilon_x = 5.09\pi\,\mathrm{mm\,mrad}$ and in (b) a beam emittance of $\varepsilon_x = 16.1\pi\,\mathrm{mm\,mrad}$. The observation angle θ describes the angle between the beam propagation and the observer and is taken along the polarization plane of the laser.

Fig. 5.11 shows two simulated spectra to illustrate this procedure. The left spectrum was calculated from 2000 electron-trajectories with a beam emittance of $\varepsilon_x = 5.09\pi\,\mathrm{mm\,mrad}$.
These traces were obtained with the software *General Particle Tracker* (GPT) [36, 37]. For the spectra on the right side of Fig. 5.11, 2000 electrons with a beam emittance of $\varepsilon_x = 16.1\pi\,\mathrm{mm\,mrad}$ were simulated with GPT and subsequently analyzed with *CLARA 2.0*. Clear differences between the two spectra are visible. The higher beam emittance causes a wider spectrum and is distinguished by a smaller peak intensity. Obviously, the skewness along the frequency axis is more negative for the higher than for the lower emittance. These spectra are clearly distinguishable and should allow to conclude about the beam emittance if observed in experiments.
Until now, this software has been used to verify the results of the already

existing CLARA code [18]. However, its parallel design allows to produce results much faster by using the HZDR compute cluster more extensively.

5.2.4 Radiation from full scale plasma simulations

Physical scenario

A physical scenario of interest to be studied is the radiation due to the injection of electrons during the bubble or blow-out regime in a laser-plasma interaction [5]. This regime can be reached by a high-intensity laser pulse interacting with an under-dense plasma. Due to the laser's high intensity $a_0 > 1$ a nonlinear plasma wave is excited. This causes a separation of the plasma electrons from the ions, which can be assumed to be stationary at that timescale. The nearly electron-free bubble created right behind the laser pulse traps electrons from the sides because of its positive charge density. Due to the positive ion background the electrons injected in the bubble are accelerated inside the electron-free plasma bubble. In this cavity-like bubble, the electrons undergo oscillations caused by their initial position in the phase space of the bubble. This quivering motion is called betatron oscillation and is the origin of characteristic radiation [38–40]. Often, several of these bubbles with electron injection appear, located one behind the other.

Feasibility study using CLARA 2.0

Three separate steps had to be performed in order to analyze such an scenario using *CLARA 2.0*: simulating the particle dynamics, sorting the data in order to create trajectories, and finally calculating the radiation. All these steps cause a lot of input and output to the file system.
The large particle-in-cell simulation was performed using the program Illumination [41] on 128 computer nodes at the hypnos cluster at HZDR. The physical scenario studied covered a high-intensity laser pulse $a_0 = 4.5$ and $\tau_{\mathrm{FWHM}} = 5\,\mathrm{fs}$ interacting with an under-dense plasma of density $n = 10^{19}\,\mathrm{cm}^{-3}$ particles per cubic centimeter.
The simulation created about 180 TB of electron data, containing position and velocity of each macro-electron for all time steps. In order to distinguish each particle and be able to calculate derivatives of the velocity, each particle was given a unique ID determined at its creation in the simulation. Since PIC codes in general do not need to keep track of individual particles, particle data get mixed during the simulation and are stored more and more randomly arranged with progressing simulation time. Extracting electron trajectories from the data in a feasible time requires to sort the electron data beforehand. Therefore, a first processing step was needed to sort all particle data for every time step. A software, able to read the compressed data in parallel, to sort the IDs, and to store everything again in a compressed way was developed. To run in parallel, *MPI* was used to work with different time steps at the

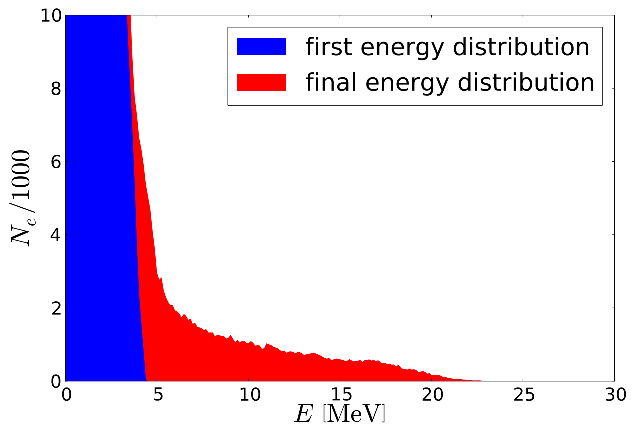


Figure 5.12: A histogram of the electron energy: for detailed study, the beginning of the injection was selected. Right before the injection, the maximum energy of the electrons is about 5 MeV (blue). During the injection electrons in the first bubble are accelerated. 1000 time-steps later, there are already several electrons with a high energy (red). The upcoming task will be to analyze the radiation occurring during these 1000 time steps of the start of the injection.

same time while *OpenMP* was used to load, decompress, compress and store all particle data (see appendix B.2 for details on MPI and OpenMP).
Running through all 180 TB of electron data took about one month. The limiting factor of this process was not the number of processors available but the bandwidth of the network connecting the computer nodes to the hard drives. It limited the data transfer to about $250 - 350$ MB/s.

In order to not occupy most of the available disk space, the only feasible option was reducing the electron data. It was decided to delete the entire data except for the injection of electrons into the first bubble which should be investigated in detail. Therefore, the electron energy was analyzed to determine the time of the first injection. Prior to the self injection, the electrons' energy spectrum is limited to about 5 MeV. During the self injection, electrons inside the bubble get accelerated. This causes the electrons' energy spectrum to "grow a tail" towards higher energies. The clear signature was used to decide on which time steps of the simulation to keep and which to delete. 1000 time steps covering the first self injection have been kept. The electrons' energy histogram for the first and last time step preserved is illustrated in Fig. 5.12. A cut along the laser polarization plane through the first electron bubble is shown in Fig.5.13. The area where the particle energy is on average above 6 MeV is highlighted by a black hatching.
This reduction of the electron data freed a lot of disk space. But $2.3 \cdot 10^8$

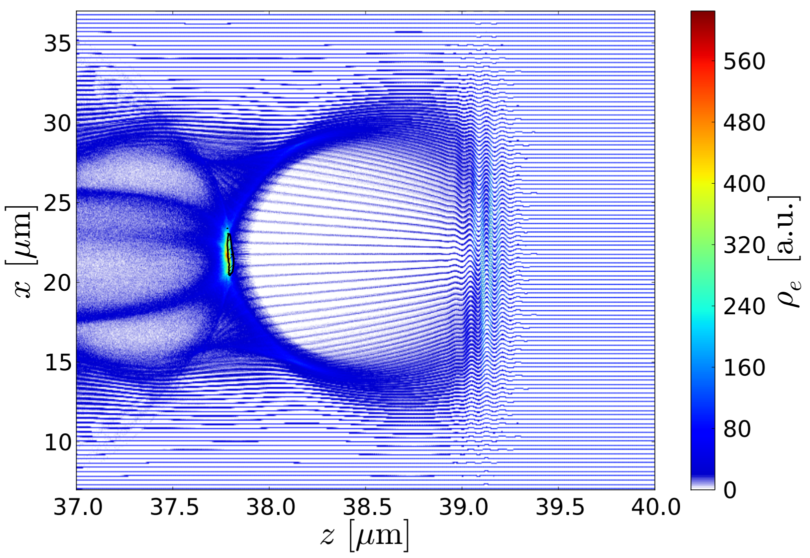


Figure 5.13: The electron density in the bubble: The picture shows a cut along the laser polarization plane through the first electron bubble. The laser pulse is coming from the left and is located at $z \approx 39\,\mu$m. The black hatched area represents the region with an average electron energy above 6 MeV, the start of the electron injection into the bubble. Clearly visible are the streamline electron motion typical for the grid-like initialization of the electrons in Illumination.

macro particles still needed to be sorted. A further reduction could be achieved by deleting individual particles in the simulation. This however results in a poorer statistical description of the emitted radiation. Even choosing only the $1.2 \cdot 10^5$ particles with energy above 5 MeV would still takes month to compute.

Performing this kind of simulation demonstrates clearly the limits of computing the radiation based on a full set of electron trajectories stored on disk. Due to the finite bandwidth for in- and output, the calculation time increases tremendously even on high performance compute clusters. This can only be partially circumvented by parallelizing the computation of the radiation. Thus, an approach based on trajectory files is impractical with regard to the required computation time. Only combining the particle simulation with the radiation code, as done by the *Radiation Analyzer* in *PIConGPU*, seems to be a feasible method to simulate the radiation of full-scale plasma simulations.

Efficient alternative using the Radiation Analyzer

The *Radiation Analyzer* included in *PIConGPU* allows to do full scale plasma simulations and compute the emitted radiation efficiently. Billions of particles can be considered allowing for a great statistical treatment of the total emitted radiation.
The *Radiation Analyzer* simulates the emitted radiation for many observers in parallel. For the first time, this allows to compute a complete spectral sky map of the radiation emitted during a laser wakefield acceleration [5] (Fig. 5.14). Radiation signatures emitted towards a specific direction can thus be identified and linked to specific plasma dynamics. Detecting such spectral signatures could optimize the diagnostics of laser-driven particle accelerators.

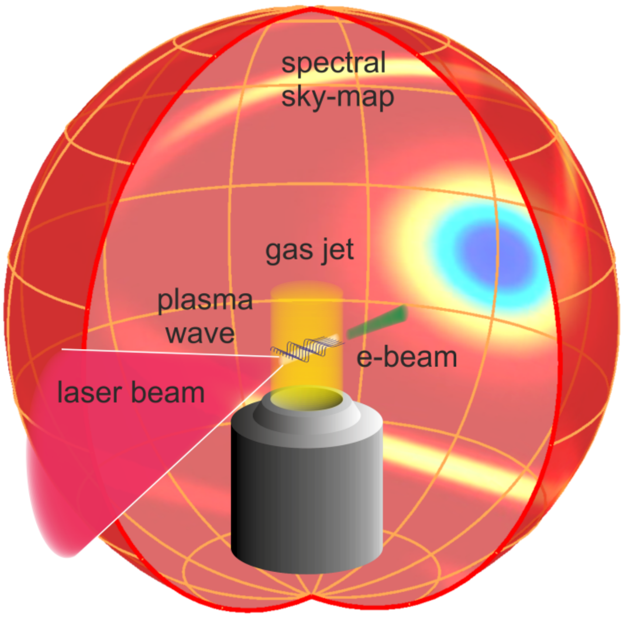


Figure 5.14: Schematic drawing of the radiation sky map together with a physical scenario of interest: The laser is focused into a gas jet and ionizes the gas, thereby creating a plasma.The ultrashort laser pulse excites a plasma wave that accelerates electrons into the forward direction. For predicting spectral signatures based on simulations the entire spectral sky map needs to be computed. In order to diagnose this wakefield accelerator the electromagnetic spectrum can be detected and compared with the predictions of the sky map simulation.

Additionally, the *Radiation Analyzer* can compute the emission intensity for arbitrary frequencies. This enables to calculate logarithmically scaled spec-

tra, ranging from infrared to X-ray, in a single run of *PIConGPU*.
In order to perform these full-scale simulations, the code needs to run efficiently on high performance compute systems. The speedup achieved by using more GPUs for a fixed-sized problem was measured on the TITAN cluster at Oak Ridge National Lab (ORNL), USA. The time needed for different numbers of GPUs was compared to that of only 32 GPUs. The speedup of the code in this strong scaling test is close to perfect (Fig. 5.15). This exceptional result is rarely reached by parallel codes.

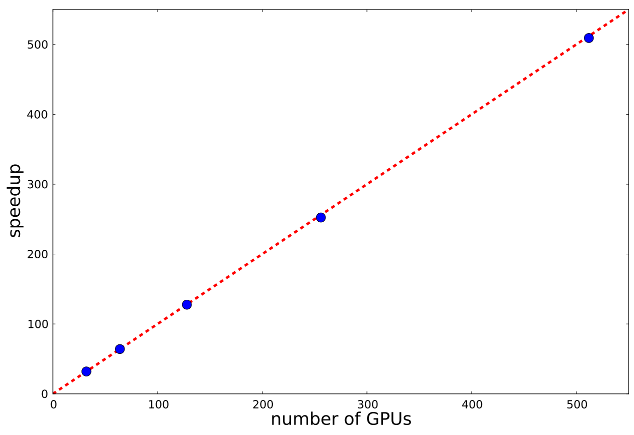


Figure 5.15: Speedup per compute node (strong scaling) of *PIConGPU* with *Radiation Analyzer*. The red line represents the ideal speedup, while the blue dots are the speedup values actually achieved with the code.

This means that running the code on hundreds of graphic cards in parallel is possible without losing parallel efficiency. There is no apparent reason why the *Radiation Analyzer* should not scale for up to ten thousand GPUs.

In order to simulate the radiation emitted during a plasma interaction in the blow-out regime with *PIConGPU*, hundreds to thousands of GPUs running in parallel are required.
A proposal of running PIConGPU together with the radiation code on the TITAN cluster was submitted to and accepted by the Oak Ridge Leadership Computing Facility (OLCF) at Oak Ridge National Lab (ORNL), USA, the largest computer cluster at present [42]. This will allow to run a simulation with up to 1000 GPUs in parallel. A maximum of 1 million CPU-hours can be spent on simulations. Since every GPU is assigned to 16 CPU cores, this means that 62500 GPU-hours are available to run PIConGPU. When using around 1000 GPUs, a single simulation would need less than three days. An equivalently large simulation would need nearly two years on the *joker* cluster at the TU Dresden.

Feasible setup parameters of the simulation are given in the table 5.1. There are three different setups for the simulation, each with a different number of observation points on the sky map. The number of observation points is limited to around one hundred. This is due to the restrictions of hardware memory used in accordance with the project specifications. If all of the $18,688$ GPUs available in OakRidge are used and one assumes close-to-perfect scaling as indicated by Fig. 5.15, one could use almost $20,000$ observation points to cover the complete 4π sphere. This would allow a detailed scan of the entire spectral sky map for frequencies ranging from infrared to X-ray in a matter of days.

Table 5.1: Setup parameters for the TITAN simulation

parameter	**values**
GPUs used	$5 \times 32 \times 6$ GPUs $= 960$ GPUs
Total number of cells	$400 \times 4608 \times 408$
Cell size	$\Delta x = \Delta z = 0.1722 \cdot 10^{-6}\,\mathrm{m}$ $\Delta y = 0.22 \cdot 10^{-7}\,\mathrm{m}$
Time step	$\Delta t = 0.08\,\mathrm{fs}$
Total simulation time	$25,000 \cdot \Delta t = 2,000\,\mathrm{fs}$
Volume simulated	$V = 4.9 \cdot 10^{-4}\,\mathrm{mm}^3$
Initial electron density	$\rho = 2.2 \cdot 10^{25}\,\frac{1}{\mathrm{m}^3}$
Macro particle per cell	4
Macro particle per GPU	3,133,440
Total number of macro particles	$N_\mathrm{p} = 3.008 \cdot 10^9$
Option 1	128 observers 11.4 s/time step total runtime: 79.2 h $\rightarrow 1.22 \cdot 10^6$ CPU-h
Option 2	96 observers 8.8 s/time step total runtime: 61.2 h $\rightarrow 0.88 \cdot 10^6$ CPU-h
Option 3	80 observers 7.4 s/time step total runtime: 51.4 h $0.74 \cdot 10^6$ CPU-h

In order to illustrate that the *Radiation Analyzer* works for large collections of particles, a result of a test run is given in Fig. 5.16. An electron bunch with a Gaussian bunch size of $\sigma_x = \sigma_z = 2\sigma_y = 6\,\mu\mathrm{m}$ and $\gamma = 5$ interacts with an electromagnetic plane wave of intensity $a_0 = 1.0$. The bunch size is about 2 orders of magnitude larger than the wavelength of the first harmonic emitted. The resulting radiation is incoherent and causes a noisy spectrum. However, the spectrum agrees well with the predictions in section 5.2.2. There are more sidelobes since the radiation was computed for the entire interaction with the super-Gaussian laser pulse. The energy emitted per particle is higher since the electrons passed more than seven laser periods. For this test 32 GPUs on the JUDGE cluster at the Research Center Jülich,

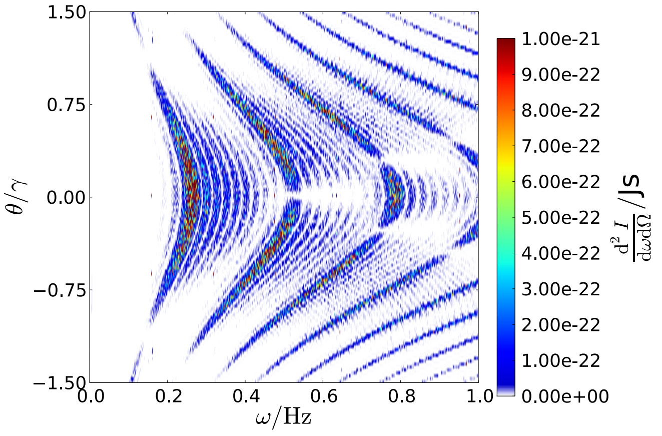


Figure 5.16: Example of radiation calculation on several GPUs. This is the original spectra as simulated, no post processing was applied.

Germany [43] were used. A total of 5 million particles have been used for the calculation.

All tests covered in this thesis were aimed at verifying the correctness of the developed *Radiation Analyzer*. Since such a huge simulation can just run once, all components had to be checked extensively. This has been done using the joker cluster at the TU Dresden and the JUDGE cluster in Jülich. All tests of the *Radiation Analyzer* have turned out to agree very well with the predictions (see section 5.2.2 to C for details). This allows to face the planned simulation in Oak Ridge with confidence.

6 | Conclusions

In this work I presented the development three different codes to simulate electromagnetic fields from electrons moving at relativistic speeds.

The time domain simulation allows to simulate electric and magnetic fields from electrons at arbitrary positions and times. Because the code considers the full solution of the Maxwell equations for point-like particles by Liénard-Wiechert potentials, the electromagnetic fields calculated are correct for particles at relativistic speeds. This allows to simulate fields where standard mesh-based field solvers would need too many grid points to correctly calculate the electromagnetic fields. An example of such a case is coherent synchrotron radiation (CSR). The time domain code is a useful tool to consider the reaction of the bunch to its own field. Right now, the software is being tested by the "Radiation Source ELBE" division at the Helmholtz-Center Dresden-Rossendorf and by the Center for Nonlinear Studies at the Los Alamos National Lab. It agrees very well with analytical solutions for several test cases.
Furthermore, it might be useful for initializing the electromagnetic fields of arbitrary particle sources in PIC simulations. Such a relativistically correct field initialization is inevitable when considering situations with non-zero electromagnetic fields at start and if the particles' initial velocity is already close to the speed of light, as for example in the case of electron bunches injected into a laser-driven wake field for post-acceleration.

The frequency domain code allows to spectrally and directionally resolve the radiated intensity from accelerated charges. It is well suited to compare spectra from experiments with simulated particle dynamics or to predict characteristic radiation with simulations. By exploiting the highly parallel structures of compute clusters, results can be produced faster allowing parameter scans with respect to beam and laser properties. The software is independent of any specific particle simulation and can easily be adjusted to different input files. Currently, the simulation developed is used in parallel to the CLARA 1.0 code to simulate effects of beam emittance at ELBE for head-on Thomson scattering of ELBE bunches with laser pulses from the high-power laser DRACO, and it will be used to analyze the radiation from electron self-injection in laser plasma interactions in blow-out regime.

The Radiation Analyzer was developed to calculate the energy emitted from accelerated electrons simulated with PIConGPU. It is able to produce the same results as the cluster-based frequency domain code but because it runs directly within PIConGPU it can completely avoid time loss due to storing and reading trajectory files from simulations. It also exploits the highly parallel architecture of graphic cards to speed up the radiation simulation immensely. This gives the unique opportunity to simulate the radiation for entire laser plasma simulations which to the author's best knowledge had not been implemented in any other PIC code due to the heavy computational load of this method. With this simulation considering millions of particles, it will be possible to investigate radiation effects during laser plasma interactions and perhaps link characteristic spectral signatures to effects predicted by simulations. It is able to simulate coherent radiation, allows to analyze frequencies from infrared to X-ray in a single run and is fast enough to simulate the radiation on a full spectral sky map.

As part of the diploma thesis, all these codes have been thoroughly tested to assure that the simulated results are correct. The results of these test simulations where checked against known analytical solutions. For the frequency domain code, these tests ranged from simple Coulomb fields to highly relativistic *coherent synchrotron radiation*. For the frequency domain codes, tests of undulator and non-linear Thomson scattering were performed. In addition, tests were also performed to reveal the limits of these codes. These limitations could be pushed further by applying several tricks conceived. For example, the *Nyquist limiter* is a new method developed to prevent incorrect reflection in frequency space when dealing with many particles and the $(1 - \vec{\beta}\vec{n})$ approximation permits to simulate radiation on GPUs.

The extensive validation process, especially for the Radiation Analyzer used in PIConGPU, now allows one to assess changes and newly developed methods to PIConGPU with standardized radiation tests to check the effects of the changes against known analytical spectra. In contrast to all other output created by PIConGPU, radiation spectra have the advantage of summing up all effects on the electrons and thereby revealing errors quickly. This validation method has already been applied to several newly developed particle-pusher algorithms and was able to disclose previously undiscovered bugs.

The three codes are powerful tools for comparing simulations with experiments and for predicting field and energy distributions caused by electrons.

The fact that these programs are already applied by different groups show that there had been a need for radiation codes that could be covered by these programs.

Predicting radiation from laser plasma interactions was previously mainly treated analytically which only allowed to cover a few special cases. Simulations often only reproduced tiny fractions of the entire process due to memory limitations. The PIConGPU's Radiation Analyzer now allows to predict for the first time the radiation for the entire plasma. Therefore, new effects are likely to be seen. Hopefully the first new results will be found next year with the first-large scale simulations.

This simulation will be the first PIC code ever to realistically simulate the entire radiation emitted during a laser wakefield acceleration.

7 | Outlook

Large-scale simulations on the Oak Ridge National Laboratory supercomputer TITAN will start at the beginning of 2013. This will be the first full-scale radiation simulation of laser wakefield acceleration (LWFA). It will give a unique insight into the radiation signatures seen in LWFA. Linking spectral signatures to specific particle dynamics will allow to improve experimental diagnostic methods and thus allow to better optimize experimental setups.

A possible extension of the *Radiation Analyzer* will consider coherent radiation independently for small-scale spatial subsets of cells. This would give the opportunity to combine the radiation originating from these cells coherently, while combining the radiation over all cells incoherently. With this method, the noisy background due to the *coherent radiation method* could be avoided, while still being able to simulate coherent radiation correctly. Additionally, this spatial decomposition of calculating the radiation would allow to pinpoint radiation signatures to a specific location of origin.

For future simulations the effects of the radiation back reactions needs to be considered: In the previous chapters, several aspects of electromagnetic radiation emitted by electrons have been covered. However, consequences of the energy carried away by the radiation have been ignored. This is due to two reasons. First of all, the back reaction of the emitted electromagnetic radiation on the accelerated charge, the so-called *radiation reaction*, can only be dealt with correctly using quantum electrodynamics [6]. And secondly, in most cases, the changes in the particle's dynamics caused by the emitted radiation can be neglected.
Nonetheless, there exist several approximate classical treatments of radiation reaction and for future high intensity lasers with $a_0 > 10$, equal to a laser peak intensity larger than $I > 2.2 \cdot 10^{20}\,\mathrm{W/cm^2}$ for a laser wavelength of $\lambda_0 = 800\,\mathrm{nm}$, the effects of electromagnetic emissions can no longer be ignored [44]. Therefore, this chapter will briefly introduce the classical treatment of radiation reaction, outline its disadvantages, and propose a novel approach based on the numerical solution for electromagnetic radia-

tion presented in this thesis. This approach might solve some of the current methods' shortcomings.
The power lost by radiation can be described by Larmor's formula (Eq. 7.1). It is derived by integrating the Poynting vector (Eq. 2.33) over a sphere surrounding the charge emitting the radiation [45].

$$P = \frac{\mu_0 \cdot q^2 \cdot c}{6\pi} \cdot \dot{\beta}^2 \tag{7.1}$$

This formula assumes that the particle's velocity is negligible compared to the speed of light $v \ll c$. The emitted power is proportional to the square of the charge q and the square of the normalized acceleration $\dot{\beta}$. In case of a particle with relativistic velocity, this scaling remains true but additional terms gain influence. The relativistically correct formula for emitted power (Eq. 7.2) was first derived by Liénard [6].

$$P = \frac{\mu_0 \cdot q^2 \cdot c}{6\pi} \cdot \gamma^6 \left[\dot{\beta}^2 - \left(\vec{\beta} \times \dot{\vec{\beta}}^2\right)\right] \tag{7.2}$$

A common way to include the radiation reaction is by using the Abraham-Lorentz formula (Eq. 7.5). To keep things simple, all following derivations will assume a non-relativistic scenario. The Abraham-Lorentz approach accounts for the energy loss (Eq. 7.1) by introducing a radiation reaction force $\vec{F}_{\text{rad}}$ acting on the charge. However, the particle constantly exchanges energy with the near field ($\vec{E} \sim \frac{1}{r^2}$), which is negligible in the Larmor formula because it does not carry away energy. Therefore, the radiation force introduced will only be correct in an average description of the electron's dynamic [45].

$$\int_{t_1}^{t_2} \vec{F}_{\text{rad}} \cdot \vec{\beta} \cdot c \, \mathrm{d}\, t = -\frac{\mu_0 \cdot q^2 \cdot c}{6\pi} \int_{t_1}^{t_2} \dot{\beta}^2 \, \mathrm{d}\, t \tag{7.3}$$

Assuming the particle motion to be periodic or $\dot{\vec{\beta}} \cdot \vec{\beta} = 0$ allows to integrate the left part of Eq. 7.3 by parts [6], leading to:

$$\int_{t_1}^{t_2} \left(\vec{F}_{\text{rad}} - \frac{\mu_0 \cdot q^2 \cdot c}{6\pi} \ddot{\vec{\beta}}\right) \cdot \vec{\beta} \cdot c \; \mathrm{d}\, t = 0 \quad . \tag{7.4}$$

It is no problem to assume the force to be:

$$\vec{F}_{\text{rad}} = \frac{\mu_0 \cdot q^2}{6\pi} \ddot{\vec{\beta}} \quad . \tag{7.5}$$

All other options for $\vec{F}_{\text{rad}}$ turn out to be more complicated and even less correct. It is important to keep in mind that the Abraham-Lorentz equation just describes the time averaged influence of the radiation on the particle over a special time interval. Additionally, this formula ignores effects perpendicular

to $\vec{\beta}$, which would change the direction of the particle's momentum but not its energy. Furthermore, the force introduced by Abraham and Lorentz leads to unphysical "runaway" solutions of the particle's motion and to preacceleration effects.
An alternative to Abraham-Lorentz is the description by Laundau and Lifshitz. By applying a time derivative on Newton's second law, one can find a formula for the normalized acceleration

$$\ddot{\vec{\beta}} = \frac{1}{m \cdot c} \dot{\vec{F}}_{\text{ext}} \quad . \tag{7.6}$$

Applying this to the Abrahm-Lorentz force results in:

$$\vec{F}_{\text{rad}} = \frac{\mu_0 \cdot q^2}{6\pi m c} \dot{\vec{F}}_{\text{ext}} \quad . \tag{7.7}$$

This is the Landau-Lifshitz formula which implies that the change in acceleration $\ddot{\vec{\beta}}$ is solely caused by the change of external forces $\dot{\vec{F}}_{\text{ext}}$ [46]. It is only an approximation of the Abraham-Lorentz equation but it can be of advantage concerning the preacceleration and the diverging solutions. However, it can be shown that both formulae result in unphysical behavior while Abraham-Lorentz usually shows the better results in the limit of point-like charges [46].
Both formulae describe the energy loss caused by the emitted radiation by arbitrarily introducing a radiation force. This force arises from the particle's interaction with its own field. This is hard to imagine for a point-like particle where the field diverges at the position of the charge. However, for continuous, not point-like, charge distributions, it can be shown that the charge interacts with its own retarded field. This remains true even in the limit of the charge's spatial extension approaching zero (see e. g. [45] for the case of a dumbbell or [46] for a spherical shell).
In particle-in-cell simulations, the fields caused by all charges are solved on a grid using Maxwell's equation. These fields do act back on the charges. Therefore, back reactions of the charges on themselves are already included up to a certain limit. This limit is given by the spatial resolution of the field lattice. If the electromagnetic fields are solved on a grid with a node distance of λ_{lattice}, the minimal wavelength λ_{limit} that can be resolved on a cubic grid is about

$$\lambda_{\text{limit}} \approx 2 \cdot \lambda_{\text{lattice}} \tag{7.8}$$

depending on the direction of field propagation. Effects with a periodicity below λ_{limit}, or equivalently at frequencies above $\omega_{\text{limit}} = \frac{2\pi c}{\lambda_{\text{limit}}}$, are omitted and do not act back on the emitting charge. As an example, the field lattice needs to resolve the laser. It does this by sampling the laser about an order of magnitude more precisely than needed by the Nyquist-Shannon sampling theorem (see Eq. A.12). Therefore, back reactions caused by ef-

fects at frequencies around the laser frequency are already considered by the particle-in-cell algorithm. In contrast the impact of self fields at frequencies more than an order of magnitude higher than the laser frequency are ignored. Adding one of the previously described radiation forces to the equation of motion will result in double-counting of the low frequency part of the radiation, while not adding F_{rad} ignores the radiation reaction caused by high frequency effects. However, radiation effects at high frequencies usually arise from particles with high velocity and therefore high energy. Considering the relativistic Larmor equation (Eq. 7.2), these particles radiate energy proportionally to the sixth power of their energy $P \sim \gamma^6$. Thus, the radiation damping is strong for these particles.
It seems that it is only possible to overestimate the radiation damping of the many low-energetic particles in laser plasma simulations, or to underestimate the effects of radiation damping for the few high-energetic particles where the energy loss actually becomes important.
Effects below ω_{limit} are taken into account by the particle-in-cell simulation and only the emitted radiation with frequencies above ω_{limit} needs special treatment. To do this, one needs to consider the spectrally resolved emitted intensity. Integrating the emitted intensity over all frequencies above ω_{limit} and over the total solid angle results in the total emitted energy not considered by the particle-in-cell algorithm. This energy loss can be treated similarly to the original Abraham-Lorentz equation.

$$\int_{t_1}^{t_2} \vec{F} \cdot \vec{\beta} \cdot c \,\mathrm{d}t = -\int \mathrm{d}\Omega \int_{\omega_{\text{limit}}}^{\infty} \mathrm{d}\omega \, \frac{\mathrm{d}^2 I}{\mathrm{d}\omega \,\mathrm{d}\Omega} \tag{7.9}$$

$$= -\frac{q^2}{16\pi\varepsilon_0 c} \int \mathrm{d}\Omega \int_{\omega_{\text{limit}}}^{\infty} \mathrm{d}\omega \left| \int_{t_1}^{t_2} \mathrm{d}t \, \frac{\vec{n} \times \left[(\vec{n} - \vec{\beta}) \times \dot{\vec{\beta}} \right]}{(1 - \vec{\beta} \cdot \vec{n})^2} \cdot \mathrm{e}^{\mathrm{i}\omega(t - \vec{n} \cdot \vec{r}/c)} \right|^2 \tag{7.10}$$

This formula describes the total energy lost by a particle due to radiation reaction not considered indirectly through the field lattice. However, deriving an instantaneous force from this equation turns out to be difficult because, as the Abraham-Lorentz equation, it would only be valid in a time average. Additionally, considering the Fourier transform correctly holds other difficulties. Implementing an algorithm based on this basic principle of separate treatment of the radiation reaction poses a still unsolved problem. However, it needs to be solved when simulating laser plasma interactions at laser intensities above $a_0 > 10$ where the radiation back reaction starts influencing the electrons' dynamics.

A | Fourier analysis

The following chapter is supposed to present the definitions of Fourier transform used in this book and recapitulate the most important theorems of Fourier analysis, Fourier series, discrete Fourier transform and the famous Fast Fourier Transform.

A.1 Fundamentals of Fourier analysis

Fourier analysis is a method to describe any function $\mathbb{R} \to \mathbb{R}$ or $\mathbb{C} \to \mathbb{C}$ by trigonometric or complex exponential functions. An extension to more than one dimension is possible. This transformation is especially useful for functions with a periodic behavior.
In the following brief description, I will concentrate on the one dimensional complex Fourier analysis and will speak of time and frequency space, even if this represents just one possible interpretation. The decomposition of a function in time domain to a function in frequency domain ($f(t) \to F(\omega)$) is called Fourier transform and can be defined as:

$$F(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} f(t) \mathrm{e}^{+\mathrm{i}\omega t} \,\mathrm{d}\, t \tag{A.1}$$

The inverse operation is called Fourier synthesis and results from above's definition as follows:

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} F(\omega) \mathrm{e}^{-\mathrm{i}\omega t} \,\mathrm{d}\, \omega \tag{A.2}$$

There are several different definitions of the Fourier transform and synthesis, mainly using different constants. This definition has simply been chosen because it is used in Jackson's Classical Electrodynamics [6]. Any other definition would be as good, only resulting in minor changes of some constants.
The Fourier transform has a variety of properties. Only those of importance to radiation are covered here. For a complete description, please refer to [47].

- The Fourier transform is a linear operation, meaning that for any $a, b \in \mathbb{C}$ a function $h(t) = a \cdot f(t) + b \cdot g(t)$ will transform to $H(\omega) = a \cdot F(\omega) +$

$b \cdot G(\omega)$. This property can be applied when calculating the emitted radiation of several particles. An addition in time and frequency space is equivalent and therefore can be implemented as needed.

- A translation in time will lead to a phase shift in the frequency domain. If translating $f(t)$ by a constant $a \in \mathbb{R}$, the Fourier transform $F(\omega)$ will change to $F(\omega) \cdot \mathrm{e}^{+\mathrm{i}\omega a}$. This needs to be used when considering emitted radiation infinitely far away from it's point of origin. The unknown, and infinite, time the radiation needs to propagate is just resulting in an unknown and often uninteresting phase.

- As can be derived from the rules of integration by substitution, scaling of the time $f(t) \to f(a \cdot t)$ also scales the Fourier transform $F(\omega) \to \frac{1}{|a|} F(\frac{\omega}{a})$ in frequency space. This needs to be used when a unit transform, as in *PIConGPU*, is applied.

- Of importance is also Parseval's theorem:

$$\int_{-\infty}^{+\infty} |f(t)|^2 \,\mathrm{d}\, t = \int_{-\infty}^{+\infty} |F(\omega)|^2 \,\mathrm{d}\, \omega \tag{A.3}$$

  It allows for substituting a power integration over time by an integration over frequency and still get the correct total energy as applied in the derivation of section 2.1.3.

- The convolution theorem states that a convolution in time space $h(t) = f(t) \otimes g(t) = \int_{-\infty}^{+\infty} f(s) \cdot g(t-s) \,\mathrm{d}\, s$ is related to a multiplication of the Fourier transforms $H(\omega) = F(\omega) \cdot G(\omega)$ and vice versa. This theorem is useful when, for example, deducing the effects of an envelope function added to an existing solution.

A.2 Discrete Fourier transform

For a numerical approach to Fourier transforms, one is not dealing with functions anymore but with series of values. This relates closely to the topic of Fourier series and illustrates the idea behind Fourier transforms.
Assuming a function with a periodicity T, it is only necessary to know the function at $t \in [0, T]$ to know it everywhere. In order to describe this function using complex exponential functions, one can concentrate on those $\mathrm{e}^{\mathrm{i}\omega_k t}$ with

$$\omega_k = \frac{2\pi}{T} \cdot k \quad , \tag{A.4}$$

because no other complex exponential function will fulfill the periodicity T. With this knowledge, one can define a general form for the function:

$$f(t) = \sum_{k=-\infty}^{+\infty} F_k \cdot \mathrm{e}^{-\mathrm{i}\omega_k t} \quad . \tag{A.5}$$

By defining a Fourier transform similar to Eq. A.1

$$F_k = \frac{1}{T} \int_0^T f(t) \cdot \mathrm{e}^{+\mathrm{i}\omega_k t} \, \mathrm{d}\, t \tag{A.6}$$

and knowing that

$$\frac{1}{T} \int_0^T \mathrm{e}^{-\mathrm{i}\omega_l t} \cdot \mathrm{e}^{+\mathrm{i}\omega_k t} \, \mathrm{d}\, t = \delta_{kl} \tag{A.7}$$

one sees that the Fourier transform selects the contribution F_k of a single frequency ω_k from the series Eq. A.5. The different normalization of Eq. A.1 and Eq. A.6 arises from $T \to \infty$, which causes ω_k to become a continuous parameter ω. However, since definition Eq. A.6 would lead to a divergence, a redefinition of the Fourier transform is needed: $\lim\limits_{T\to\infty} (T \cdot F_k) = (2\pi)^{-1/2} F(\omega)$. From experimental measurements or simulations only values at discrete points in time are usually known. These can be analyzed by using the discrete Fourier transform. In order to simplicity the derivation, a signal f_j at equidistant time steps is assumed:

$$t_j = j \cdot \Delta t \qquad \text{with: } j \in \{0, 1, \ldots, N_t - 1\} \quad . \tag{A.8}$$

Since the signal outside of $t \in [0, N \cdot \Delta t]$ is by definition unknown, for sake of simplicity a periodic behavior is assumed. However, this assumption causes errors that will be discussed later. Applying Eq. A.6 to the discrete series in time results in

$$F_k = \frac{1}{N \cdot \Delta t} \cdot \sum_{j=0}^{N} f(t_j) \cdot \mathrm{e}^{+\mathrm{i}\omega_k t_j} \Delta t \tag{A.9}$$

$$= \frac{1}{N} \cdot \sum_{j=0}^{N} f_j \cdot \mathrm{e}^{+2\pi \mathrm{i} jk/N} \quad . \tag{A.10}$$

By using the abbreviation $W_N = \mathrm{e}^{\frac{2\pi\mathrm{i}}{N}}$, Eq. A.7 can be rewritten for the discrete case.

$$\sum_{j=0}^{N-1} W_N^{(k-l)j} = N \cdot \delta_{k,l} \tag{A.11}$$

It is important to notice, that a real signal, e.g. a cosine $\cos(\omega t)$, will not only contribute at ω but also at $-\omega$ because $\cos(\omega t) = \frac{1}{2}\left(\mathrm{e}^{+\mathrm{i}\omega t} + \mathrm{e}^{-\mathrm{i}\omega t}\right)$. A

frequency contribution of a real signal is contributing equally at positive and negative frequencies.
A serious problem arises from the sampling of the signal itself. Assuming e.g. a cosine signal with periodicity T_c and a sampling every $\Delta t = T_c/2$ time step, one obtains values of the kind $\{+1, -1, +1, \dots\}$. A discrete Fourier transformation results correctly in a frequency of $\omega = 2\pi/T_c$. However, by decreasing the sampling rate, it is not possible to reconstruct the signal anymore. Instead, one arrives at a Fourier transform containing complete nonsense. With a sampling step-width of Δt, one can only analyze bandwidth-limited signals containing frequencies below:

$$|\omega| < \Omega_\text{Nyquist} = \frac{\pi}{\Delta t} \quad . \tag{A.12}$$

As mentioned before, contributions of negative frequencies in the spectra are the usual case. But a negative frequency $\omega = -j \cdot 2\pi/T$ is isolated in Fourier transform by:

$$\mathrm{e}^{\mathrm{i}(-\omega)t} = \mathrm{e}^{\frac{2\pi \mathrm{i}(-j)k\Delta t}{T}} = W_N^{i(-j)} = W_N^{i(N-j)} = \mathrm{e}^{\mathrm{i}(2\cdot\Omega_\text{Nyquist} - \omega)t} \tag{A.13}$$

This gives the impression of negative frequencies reoccurring above the Nyquist frequency. For real signals, this appears as if all frequencies are mirrored at Ω_Nyquist.
But what happens when the sampling rate is not large enough or the signal is not band-limited at all? Than one sees an effect called *aliasing*. A signal with a frequency of $\omega_\text{sig} = \Omega_\text{Nyquist} + \Delta\Omega$ will appear as a signal of frequency $\omega_\text{DFT} = \Omega_\text{Nyquist} - \Delta\Omega$. As lower frequencies seem to be mirrored at Ω_Nyquist, the same is true for higher frequencies. Therefore, a badly sampled signal might result in a wrong spectrum and hence, a high sampling rate should be preferred. However, this is not as easy for radiation calculation as shown in section 4.
Returning to the problematic assumption that the signal $t \in [0, T]$ will periodically continue. This might lead to serious errors, when a low frequency signal is cut of at an incomplete cycle. This problem is less important for higher frequencies and can even be avoided completely by *zero-padding* the signal. Zero-padding is simply adding additional values of zero to the signal. Extending T or N by adding zeros does not contribute to the original signal, but increases the frequency resolution of the discrete Fourier transform as can bees seen in Eq. A.14.

$$\Delta\omega = \frac{2\pi}{T} = \frac{2\Omega_\text{Nyquist}}{N} = \frac{2\pi}{\Delta t \cdot N} \tag{A.14}$$

The more zeros are added before and after the signal, the less important become the problems with frequencies not fitting in the period T.

A.3 Fast Fourier transform

The DFT algorithm can be seen as a multiplication of a signal vector f_k of length N with a $N \times N$-matrix describing the time-frequency-relation or for short the complex phase. Therefore the DFT algorithm scales with $N \times N$, costing a lot of calculation time. However, Gauss and more than a century later James Cooley and John Turkey [48] discovered that a *divide and conquer algorithm* could be applied to Fourier transform which reduces the complexity to $2N \log_2 N$ and revolutionized not only data analysis but also a wide variety of other fields dealing with computation. Here, I will follow closely James Cooley's and John Turkey's original paper to introduce the basic idea behind the *Fast Fourier Transformation*, short FFT [48] .
Assuming that the total number of data points can be expressed as $N = r_1 \cdot r_2$, one can define:

$$j = j_1 \cdot r_1 + j_0 \qquad j_0 \in \{0, 1, \ldots, r_1 - 1\}, j_1 \in \{0, 1, \ldots, r_2 - 1\} \tag{A.15}$$

$$k = k_1 \cdot r_2 + k_0 \qquad k_0 \in \{0, 1, \ldots, r_2 - 1\}, k_1 \in \{0, 1, \ldots, r_1 - 1\} \tag{A.16}$$

With these definitions, it is possible to rewrite Eq. A.10:

$$F_j = \frac{1}{N} \sum_{k_0=0}^{r_2-1} \sum_{k_1=0}^{r_1-1} f_{(k_1 \cdot r_2 + k_0)} \cdot W^{jk_1 r_2} \cdot W^{jk_0} \tag{A.17}$$

The first exponential function can be rewritten.

$$W^{jk_1r_2} = W^{(j_1r_1+j_0)k_1r_2} = W^{j_1k_1N+j_0k_1r_2} = W^{j_0k_1r_2} \tag{A.18}$$

As in [48], the inner sum over k_1, depending only on j_0 and k_0, can be redefined.

$$F_1(j_0, k_0) = \frac{1}{r_1} \sum_{k_1=0}^{r_1-1} f_{(k_1 \cdot r_2 + k_0)} \cdot W^{j_0 k_1 r_2} \tag{A.19}$$

This simplifies Eq. A.17 to:

$$F_j = \frac{1}{r_2} \sum_{k_0=0}^{r_2-1} F_1(j_0, k_0) \cdot W^{jk_0} \tag{A.20}$$

For the calculation of F_1, there are $N \cdot r_1$ operations needed. Additionally, $r_2 \cdot N$ operations are necessary to compute F_j from F_1. The total amount of operations (one complex multiplication with one complex addition) is therefore $N \cdot (r_1 + r_2) < N^2$ if $N > 4$ and the two factors r_1 and r_2 are chosen wisely. Since F_1 has a similar form as F this procedure can be applied successively if the total number of data points can be described by $N = r_1 \cdot r_2 \cdot \ldots \cdot r_m$. In the special case of all r_i being equal to r, only

$r \cdot N \cdot \log_r N$ operations are necessary, reducing the complexity of the numerical Fourier transform dramatically.
As shown in [48], choosing $r = 2$ is not the fastest method, but close to it and additionally better suited for computers and their binary arithmetic. This requires signals of length $N = 2^n$, which can always be realized by zero-padding. In this widely used case, $k_0 \in \{0, 1\}$, therefore splitting the original signal f_k into two separate signals, one containing all even steps $f_{1,k_1} = f_{2k_1}$ and the other all odd values $f_{2,k_1} = f_{2k_1+1}$ with $k_1 \in \{0, 1, \ldots, N/2 - 1\}$, results in two separate sums.

$$F_{1,j} = \frac{1}{N/2} \sum_{k_1=0}^{N/2-1} f_{1,k_1} W_{N/2}^{k_1 j} \tag{A.21}$$

$$F_{2,j} = \frac{1}{N/2} \sum_{k_1=0}^{N/2-1} f_{2,k_1} W_{N/2}^{k_1 j} \tag{A.22}$$

Applying Eq. A.17 results in:

$$F_j = \frac{1}{2} F_{1,j} + \frac{1}{2} W_N^j F_{2,j} \tag{A.23}$$

This scheme can be pursued until only two values are left for summation. The relation between the two values being added seems to be complex, but for $N = 2^n$ and $r = 2$, it turns out that the order of summation is determined by a bit-reversal of the original index [47]. Values with bit-reversed index next to each other are summed up first. A possible implementation of this algorithm with a more detailed description of the algorithm can be found in the Numerical Recipes [49]. The implemented algorithm for the radiation code is based on an improved version of the Numerical Recipes implementation proposed in [21]. Additionally, the algorithm was written as a templated [50] version, to work with the classes developed for the radiation code.

B | Parallel Programming

Requirements on simulations often exceed the abilities of a single computer in matters of available memory or aspired calculation time. Therefore, combining several computers to perform certain tasks in parallel is the standard approach to speed up the necessary time for a simulation or to increase the size of a simulation beyond the hardware limits of a single computer. In this appendix, a brief introduction to the abilities and limits of parallel computing is given. Several methods of parallelization will be introduced in condensed form. It is aimed at allowing a better understanding of the parallelization of the developed code. However, it cannot replace a detailed description of the used programs and libraries.

B.1 General remarks on parallel computing

Even as the processors speed stagnates for more than a decade, due to technical reasons [51], Moor's Law stays valid. It states that the number of transistors on a processor doubles about every 18 month [52]. This allows to use the more complex structure of modern processors to run operations in parallel. Alternatively, one may use several independent computers to process programs faster. However, both methods require to know how to exploit parallel architectures efficiently making this a inevitable skill for physicists developing simulations today.
Parallelization of programs can be put into four different categories proposed by Michael J. Flynn [53].

- *single-instruction and single-data organization* (SISD) means a completely sequential process, reading data and executing instructions both sequentially. Most programs on todays desktop computers follow this organization scheme.

- *single-instruction and multiple-data* (SIMD) means that a single instruction can be performed on several data-streams in parallel. This finds it's application in todays graphic cards and modern streaming capable CPUs [51].

- *multiple-instructions and single-data* (MISD) Running several instruction on the same stream of data is still a theoretical concept and has no real application jet.

- *multiple-instructions and multiple-data* (MIMD) finds it's application when several computing nodes run simultaneously on different data streams. This is the usual parallelization scheme on compute clusters and computers with several cores. One usually distinguishes between *distributed memory*, where all computing units have independent memory as on different computers of a cluster, and *shared memory*, where computing nodes have access to partially the same memory and can share data using that part of the memory bank.

One also distinguishes between *processes* and *threads*. A process is a set of instructions executed on a computer with an unique ID, a counter of the execution status, register values and several different kinds of allocated memory for example stack and heap memory. In contrast, a thread is also a set of instructions with its own ID, execution status, register and heap memory. However, it shares other forms of memory with other threads who also execute the same instructions [51]. Threads are light-weighted versions of processes and usually a process can start several threads. Parallelization over several computers use different processes while instructions on a GPU are executed by parallel running threads.
Executing a certain set of instructions in parallel usually results in a shorter execution time. The achieved speedup s_n is defined by the ratio between the time needed for a sequential execution T_1 and the time needed when executing the task n times parallel T_n.

$$s_n = \frac{T_1}{T_n} \tag{B.1}$$

To determine how well a code can be parallelized the efficiency e_n can be calculated by the speedup s_n normalized to the number of parallel tasks n.

$$e_n = \frac{s_n}{n} \tag{B.2}$$

There are two ways to interpret to possibilities of parallel computing. The first is based on the idea that every task is made of a part that can be executed in parallel ρ and one that only can be handled sequentially σ and that both parts are constant in size. Therefore, if a program runs completely sequentially, the total time needed is $T_1 = \rho + \sigma$. However, when parallelizing the software the time needed can be reduced to $T_n = \frac{\rho}{n} + \sigma$. By normalizing the sequential execution time to one $T_1 = 1 = \rho + \sigma$, the resulting speedup is $s_n = \frac{1}{\sigma + \frac{1-\sigma}{n}}$. This leads to the fact that the highest possible speedup of a task of fixed size is:

$$s_n \leq \lim_{n\to\infty} s_n = s_{\max} = \frac{1}{\sigma} \tag{B.3}$$

This is Amdahl's law [24] and it reveals that the possible speedup of a defined task is limited by its sequential part. A sequential part of only 10% would only allow for a speedup of 10. This shows the limits of executing a program in parallel.
However, this does not cover the entire truth. More hardware usually allows to increase the size of the simulation. This generally results in an increase of the parallel part of the program ρ compared to the sequential part $\sigma = 1-\rho$ reducing the relative sequential part σ with increasing parallelization n and therefore making larger problems more efficient. Assuming that the sequential execution time T_s is independent of the size of the problem to simulate, the speed-up can be calculated by:

$$s_n = \frac{T_1}{T_n} = \frac{T_s + n \cdot T_{pn}}{T_s + T_{pn}} = 1 + (n-1)\frac{\rho_n}{\sigma + \rho_n} \tag{B.4}$$

T_{pn} is the execution time of the parallel part when using n nodes. It can be increased by expanding the problem, while T_s stays constant. Therefore, the speed-up goes with:

$$\lim_{\sigma\to 0} s_n = n \tag{B.5}$$

This is the Gustafson's law [51].
While a fixed sized problem follows Amdahl's law (Eq. B.3) and might not profit from parallelization, the usual physics simulation can often be improved by increasing the simulated region or the number of particles making parallelization vital to keep the simulation time low.

B.2 Methods on Compute Clusters

Large scientific simulations are typically run on compute clusters. However, there is a variety of possibilities to use the available resources. First of all, a cluster consists of several computers, usually called nodes. Each computer has its own central processing unit (CPU) with its own random access memory (RAM) and is connected to the other computers of the cluster via a network, e. g. ethernet or InfiniBand. Each CPU might have several cores running in parallel and accessing the same RAM.
The most easy form of parallelization on a cluster is to write a sequential program and execute it with different input on several machines. Because this is extremely common, most clusters provide a program that takes care of starting programs on different nodes, called the *batch system.* For example the *hypnos* cluster at HZDR uses the *PBS* submit system while the GPU cluster *joker* at the TU Dresden used *SGE* and now uses *PBS pro.*

This batch, or job submission, system is often the only way to interact with the different nodes of a cluster because it schedules all jobs according to the available and requested hardware requirements. This ensures that the resources of the cluster are correctly assigned to certain jobs and are not blocked by other programs.
Another option for parallelization is to use several cores available on a CPU to run certain threads in parallel. The most easy form of this parallelization scheme is provided by the OpenMP standard [51]. It allows to tell the compiler, using so-called *pragmas* instruction in the source code, to parallelize certain instructions by creating threads. The advantage of OpenMP is that it hides the actual handling of the threads from the developer. Except for the *pragmas*, the source code still looks sequential making it more easy to check.
A more advanced method of parallelization includes the use of several nodes in parallel and sharing results between nodes while running. This can be achieved by using the message passing interface (MPI) [54]. The basic principle behind this standard is that tasks are split up into different processes with their own data. To exchange data, for example to communicate results, massages can be send and received. This simple principle allows to built complex programs running distributedly on several nodes of a cluster. One advantage of the MPI library is that in contrast to any specific network library, MPI processes can exchange messages by using the fastest method of communication available to processes. For example two MPI processes might run on the same node but on different cores of the same CPU. They would exchange messages over shared memory using the RAM of the CPU. On the other hand, on different nodes, processes can communicate using the fastest network available, which might not necessarily be the ethernet connection. The MPI library provides an abstract and easy to handle network, relieving the developer of the need to deal with technical details of network communication. It also provides a variety of useful functions for common types of data exchange, as for example broadcasting data to every node, collecting all data at a single node, summing up data of different nodes and many more. It is no problem to combine MPI with OpenMP allowing every MPI process to spawn its own parallel threads as for example the cluster based time domain code (chapter 3.2) or the sorting program for the Illuminaton data (chapter 5.2.4) do.

B.3 Massively Parallel Programming using graphic cards

Modern CPUs usually are node of several individual cores. These cores provide a full set of, generally, x86 instructions and can be considered processors of their own except that they access the same RAM. Since each core can han-

dle its own set of instructions, CPUs are called *multicore* devices. In contrast to that, graphic processing units (GPU) have far more individual processing units which however cannot all perform different instructions. Several of these *arithmetic and logical units* (ALU) share the same cache for instructions and controls. This *many-core* approach aims for a high throughput of parallel threads. [55]
Todays graphic cards provide a more than five times higher processing power than modern CPUs, as for example Intel's Sandy Bridge processors [22]. Additional, the GPUs memory bandwidth is about 4 times larger. This difference in performance makes GPUs more attractive for large simulations. To comply with the needs of scientists, NVIDIA developed the CUDA programming language to provide means to easily develop software running on CPUs and graphic cards aside from actual visualization applications [55] [22].
The basic idea behind CUDA is to extend the programming language C by methods specifically designed for NVIDIA's graphic cards. The user does not need to know many details about the architecture used because CUDA allows to abstractly describe the parallelization of the tasks. This abstraction provides the means of developing functions to be executed by several threads in parallel on the graphics card called *kernels*. To easily describe the parallelization of a kernel, a two dimensional *grid* containing several *blocks* each with a number of *threads* organized in a three dimensional grid is used. This scheme allows means for a more efficient handling of threads.
Threads of any blocks can run in parallel and have access to the same *global memory*. However, access to global memory is relatively slow. Threads of the same block also have the means of accessing a specific part of the memory only assigned to one block called *local memory*. It can be accessed faster and allows to efficiently share data between threads of a block. But shared memory is limited, making it necessary to combine only those threads to a block which profit from a faster alternative to share data.
The ALUs of a GPU are combined to several *streaming multiprocessors* (SM). Each SM provides a certain amount of shared memory accessible by all ALUs of the SM. A block of threads is executed on one SM. Not all threads of a block are executed simultaneously, 32 threads of a block, called *warp*, run at the same time. If a warp is stopped, finished or is waiting for a process to finish, another warp associated with the SM is executed. Because threads run as part of warps, it is ofter more efficient to parallelize the tasks in such a way that the number of threads in a block is a multiple of 32. A block is always just associated with one SM, but a SM can handle several blocks. A GPU consists of many SM, therefore only a parallelization with several blocks can use the hardware to its capacity.
For a detailed description of how to write CUDA code, please refer to [22] [55]. An alternative to CUDA is OpenCL, however developing using OpenCL code is at present still more difficult than using CUDA.

C | Particle statistics and coherent radiation

The radiation analyzer in PIConGPU uses the *coherent radiation method* in order to simulate coherent radiation correctly. As discussed in section 4.2.1, this algorithm leads to statistical fluctuation due to the random selection of macro-particles.
It is therefore crucial to investigate the statistical effects of the coherent radiation method when using a certain number of particles. While keeping the number of particles used for radiation calculation low results in fast simulation times, using too few particles will lead to results critically depending on the initial choice of particles.
Coherent radiation is very sensitive to the source distribution in phase space. If the average distance between two particles $\langle \Delta x \rangle$ is larger than the emitted wavelength λ_{rad}, incoherent radiation will be dominant [28]. Assuming Gaussian-distributed particles leads to the following criterion for incoherent radiation.

$$\sigma_x \approx \langle \Delta x \rangle > \lambda_{\mathrm{rad}} \tag{C.1}$$

Investigating the statistical behavior of the simulated radiation using the coherent method requires to simulate different numbers of particles with a predefined spatial distribution and a clear spectral signature. Hence, a prototype version of the *Radiation Analyzer* developed without PIConGPU was combined with a GPU-based undulator trace generator. The trace generator uses Eq. 2.47 - 2.49 to calculate the particles' motion while allowing to set the starting position of every particle using a variety of random distributions. In all test cases an electron energy of $\gamma = 50$, an undulator length of $\lambda_u = 4\,\mu\mathrm{m}$ and an undulator parameter $K = 3.7 \cdot 10^{-4}$ were set. This leads to a frequency peak at $\omega_{\mathrm{rad}} = 2.35 \cdot 10^{18}\,\mathrm{Hz}$ (Eq. 2.51) or $\lambda_{\mathrm{rad}} = 0.8\,\mathrm{nm}$ for the electron's flight direction $\theta = 0$. Furthermore, only 10 undulator periods were considered, resulting in a broad peak (Eq. 2.52).
Gaussian-distributed particles were considered to investigate the statistical effects of the coherent radiation method. The standard deviation of the distribution in all three dimensions σ was varied between completely coherent radiation $\sigma = 10^{-3}\lambda_{\mathrm{rad}}$ and entirely incoherent radiation $\sigma = 10^{+7}\lambda_{\mathrm{rad}}$. The

number of particles changed from $N_{\text{particles}} = 128 = 2^7$ up to $N_{\text{particles}} = 65536 = 2^{16}$. The choice of particle numbers is due to running the code efficiently on the GPU architecture.
In order to reduce the effects of small scale interferences, a convolution with a normal distribution, which has a standard deviation of $\sigma = 10^{16}$ Hz or 0.5% of the plotted frequency range, was additionally performed on the frequency domain. Such a convolution can be done after the simulation and therefore permits a post-processing without having to rerun the simulation.
The resulting spectra for coherent radiation agree perfectly with the prediction and there are almost no differences between the original and the smoothed spectrum. This is even true for only a small number of test particles as shown in Fig. C.1. Here, the normal distribution of the particles are set to a standard deviation of $\sigma = 0.1 \cdot \lambda_{\text{rad}}$.

In contrast, simulating radiation in the incoherent range will lead to spec-

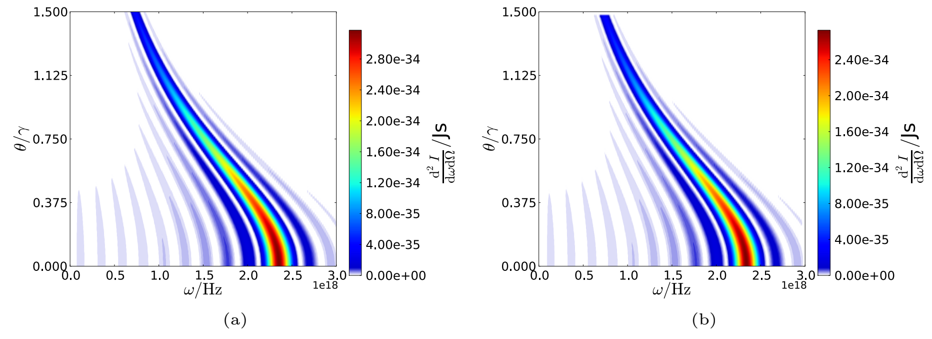


Figure C.1: Spectra from $N_{\text{p}} = 128$ particles, with a Gaussian distribution of $\sigma = 0.1 \cdot \lambda_{\text{rad}}$ in an undulator of $\lambda_u = 4\,\mu$m and $K = 3.7 \cdot 10^{-4}$ over 10 undulator periods. The electron in the undulator has an energy of $\gamma = 50$ (a) shows the original signal, while (b) shows a smoothed version.

tra with a noisy underground. The spectral structure can be identified but the noise, caused by the random particle selection, makes it hard to identify any substructures. This random noise causes striking differences between the original signal and the spectrum after convolution. However, in this incoherent range it does again not matter how many particles are considered. Higher particle numbers just lead to a slightly smoother spectrum after convolution. Fig. C.2 shows the results of simulating $N_{\text{p}} = 2^{16}$ particles with a standard deviation of $\sigma = 10^3 \cdot \lambda_{\text{rad}}$. The smoothed spectrum shows a clear structure, while for the original spectrum the signal-to-noise ratio is too high to identify any substructures on the main spectral peak. It needs to be emphasized that the energy scale in both plots differ because the convolution reduces the size of any single peak in favor of an average spectra. It is important to notice that particle distributions with a density close to

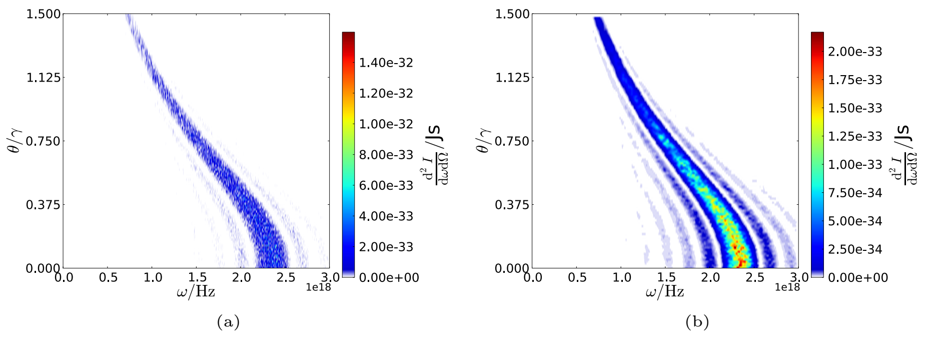


Figure C.2: Spectra from $N_\mathrm{p} = 2^{16}$ particles, with a Gaussian distribution of $\sigma = 10^3 \cdot \lambda_\mathrm{rad}$ in an undulator of $\lambda_u = 4\,\mu\mathrm{m}$ and $K = 3.7 \cdot 10^{-4}$ over 10 undulator periods. The electron has an energy of $\gamma = 50$ (a) shows the original signal, while (b) shows a smoothed version.

the critical distance of $\Delta x \approx \lambda_\mathrm{rad}$ destroy the spectral structure. This is not a numerical problem, but a physical effect. When particles are a distance equivalent to the radiated wavelength apart, their emitted fields cancel out. Numerically, this cancellation strongly depends on the random choice of particles as can be seen in Fig. C.3 for $N_\mathrm{p} = 128$ and $N_\mathrm{p} = 65536$ particles. It is important to notice that, even if the random interference effects are reduced with an increasing number of particles, a signal at low-frequencies will remain. Hence, the *Radiation Analyzer* can be set up to not calculate the low frequency part of a spectrum thereby saving simulation time.

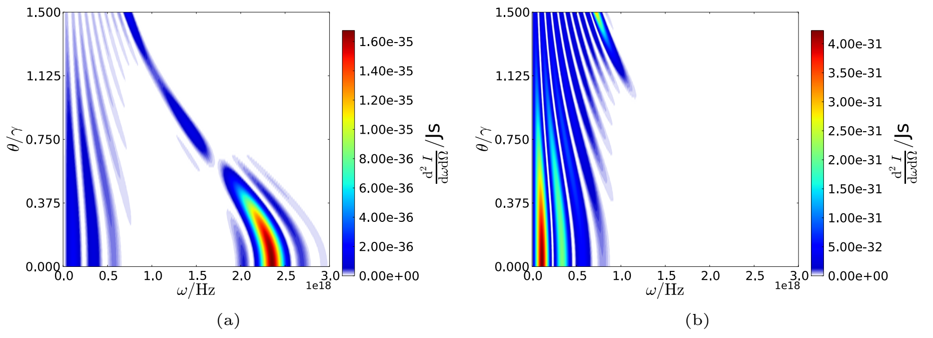


Figure C.3: Critical effects of coherent radiation: The simulated particles are normal distributed with $\sigma = \lambda_\mathrm{rad}$. No smoothing has been applied. (a) shows the emission of $N_\mathrm{p} = 128$ particles. The main form of the undulator spectra is still visible, but the cancellation is random. (b) shows the radiation of $N_\mathrm{p} = 65536$ particles. The spectrum does not show any features of the typical undulator spectrum anymore. Both spectra have a strong low frequency signal. (undulator: $\lambda_u = 4\,\mu\mathrm{m}$ and $K = 3.7 \cdot 10^{-4}$, electron: $\gamma = 50$)

Another effect occurs when considering very large standard deviations σ compared to the radiation wavelength λ_{rad}. Around a spread of $\sigma = 8\,\mathrm{cm} = 10^7 \lambda_{\mathrm{rad}}$, weak side peaks appear (Fig. C.4).

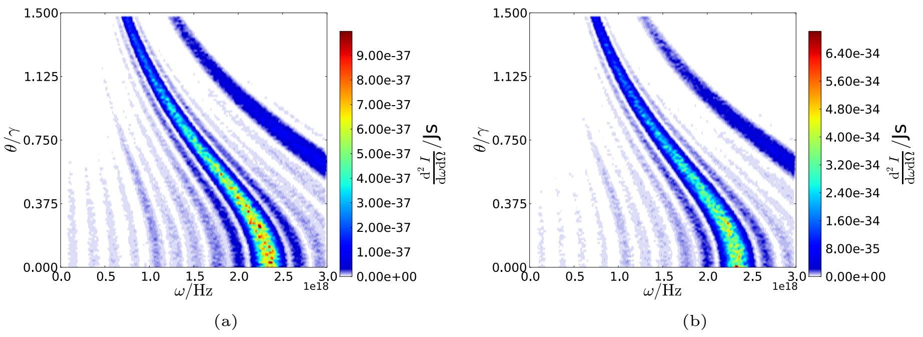


Figure C.4: Simulating very spread-out particle distributions $\sigma = 10^7 \cdot \lambda_{\mathrm{rad}}$ leads to side peaks. The effect seems stronger for $N_{\mathrm{p}} = 128$ particles (a) than for $N_{\mathrm{p}} = 65536$ particles (b). Both pictures are convoluted with a normal distribution. (undulator: $\lambda_u = 4\,\mu\mathrm{m}$ and $K = 3.7 \cdot 10^{-4}$, electron: $\gamma = 50$)

These side peaks do not disappear for larger numbers of particles N_{p}. The cause for this effect is still unknown. But it is a less relevant case because usually particles distributions of several centimeters are less common in laser plasma physics.

Bibliography

[1] H. Wiedemann, *Particle accelerator physics I: basic principles and linear beam dynamics*, vol. 1. Springer Verlag, 1999.

[2] A. Grudiev, S. Calatroni, and W. Wuensch, “New local field quantity describing the high gradient limit of accelerating structures,” *Physical Review Special Topics-Accelerators and Beams*, vol. 12, no. 10, p. 102001, 2009.

[3] T. Feder, “Accelerator school travels university circuit,” *Physics Today*, vol. 63, p. 20, February 2010.

[4] P. Gibbon, *Short pulse laser interactions with matter.* Imperial College Press London, 2005.

[5] E. Esarey, C. Schroeder, and W. Leemans, “Physics of laser-driven plasma-based electron accelerators,” *Reviews of Modern Physics*, vol. 81, no. 3, p. 1229, 2009.

[6] J. D. Jackson, *Classical Electrodynamics.* John Wiley and Sons, Inc., 1999.

[7] R. Feynman, R. Leighton, and M. Sands, *The Feynman Lectures on Physics.* Addison Wesley, 1977.

[8] K. Wille, “Synchrotron radiation.” Lecture Notes, 2008.

[9] T. Heinzl, D. Seipt, and B. Kämpfer, “Beam-shape effects in nonlinear compton and thomson scattering,” *Physical Review A*, vol. 81, no. 2, p. 022125, 2010.

[10] D. Seipt and B. Kämpfer, “Nonlinear compton scattering of ultrashort intense laser pulses,” *Physical Review A*, vol. 83, no. 2, p. 022101, 2011.

[11] U. Mohideen and at. al., “Interaction of free electrons with an intense focused laser pulse in gaussian and conical axion geometries,” *Journal of the Optical Society of America*, vol. 9, pp. 2190–2195, December 1992.

[12] D. Seipt and B. Kämpfer, “Scaling law for the photon spectral density in the nonlinear thomson-compton scattering,” *Physical Review Special Topics-Accelerators and Beams*, vol. 14, no. 4, p. 040704, 2011.

[13] E. Sarachik and G. Schappert, “Classical theory of the scattering of intense laser radiation by free electrons,” *Physical Review D*, vol. 1, no. 10, p. 2738, 1970.

[14] E. Esarey, S. Ride, and P. Sprangle, “Nonlinear Thomson scattering of intense laser pulses from beams and plasmas,” *Physical Review E*, vol. 48, p. 3003, October 1993.

[15] A. Beerlink, S. Thutupalli, M. Mell, M. Bartels, P. Cloetens, S. Herminghaus, and T. Salditt, “X-ray propagation imaging of a lipid bilayer in solution,” *Soft Matter*, vol. 8, no. 17, pp. 4595–4601, 2012.

[16] T. Yamada, M. Yuri, H. Onuki, and S. Ishizaka, “Development of a circularly polarizing microscope with a polarizing undulator,” *Review of scientific instruments*, vol. 66, no. 2, pp. 1493–1495, 1995.

[17] J. Clarke, *The science and technology of undulators and wigglers.* Oxford University Press, 2004.

[18] A. Debus, *Brilliant radiation sources by laser-plasma accelerators and optical undulators.* PhD thesis, Technische Universität Dresden, 2012. `http://www.hzdr.de/db/Cms?pOid=37269` - accessed 29.11.2012.

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

[20] G. Strang, *Wissenschaftliches Rechnen.* Springer, 2010.

[21] D. V. Myrnyy, “A simple and efficient fft implementation in c++.” `http://www.drdobbs.com/cpp/a-simple-and-efficient-fft-implementatio/199500857`, May 2007. accessed 10.10.2012.

[22] “NVIDIA CUDA C Programming Guide Version 4.2.” `http://docs.nvidia.com/cuda/cuda-c-programming-guide/`, 2012. accessed 29.11.2012.

[23] H. Burau, R. Widera, W. Hönig, G. Juckeland, A. Debus, T. Kluge, U. Schramm, T. Cowan, R. Sauerbrey, and M. Bussmann, “PIConGPU: a fully relativistic particle-in-cell code for a GPU cluster,” *Plasma Science, IEEE Transactions on*, vol. 38, no. 10, pp. 2831–2839, 2010.

[24] G. Amdahl, “Validity of the single processor approach to achieving large scale computing capabilities,” in *Proceedings of the April 18-20, 1967, spring joint computer conference*, pp. 483–485, ACM, 1967.

[25] F. Marvasti, *Nonuniform sampling: theory and practice*. Springer, 2001.

[26] P. Ehrenfest, "Welche Züge der Lichtquantenhypothese spielen in der Theorie der Wärmestrahlung eine wesentliche Rolle?," *Annalen der Physik*, vol. 341, no. 11, pp. 91–118, 1911.

[27] J. W. Goodman, *Introduction to Fourier Optics*, vol. 3. Roberts & Co Publ, 2005.

[28] A. Khachatryan, F. Van Goor, and K. Boller, "Coherent and incoherent radiation from a channel-guided laser wakefield accelerator," *New Journal of Physics*, vol. 10, no. 8, p. 083043, 2008.

[29] C. Huang, T. J. T. Kwan, and B. E. Carlsten, "Two dimensional model for coherent synchrotron radiation." Los Alamos National Laboratory - to be submitted.

[30] P. Goldreich and D. Keeley, "Coherent synchrotron radiation," *The Astrophysical Journal*, vol. 170, p. 463, 1971.

[31] Y. Shibata, K. Ishi, T. Ohsaka, H. Mishiro, T. Takahashi, M. Ikezawa, Y. Kondo, T. Nakazato, M. Oyamada, N. Niimura, *et al.*, "Coherent synchrotron radiation at submillimeter and millimeter wavelengths," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 301, no. 1, pp. 161–166, 1991.

[32] J. van Paradijs, M. Pettini, N. Tanvir, J. Bloom, H. Pedersen, H. Noerdgaard-Nielsen, M. Linden-Voernle, J. Melnick, G. van de Steene, M. Bremer, *et al.*, "Discovery of transient optical emission from the error box of the gamma-ray burst of february 28, 1997," *Nature*, vol. 386, pp. 686–689, 1997.

[33] W. Leemans, B. Nagler, A. Gonsalves, C. Toth, K. Nakamura, C. Geddes, E. Esarey, C. Schroeder, and S. Hooker, "GeV electron beams from a centimetre-scale accelerator," *Nature Physics*, vol. 2, no. 10, pp. 696–699, 2006.

[34] A. Jochmann, A. Irman, U. Lehnert, J. P. Couperus, M. Kuntzsch, A. Wagner, A. D. Debus, H. P. Schlenvoight, T. E. Cowan, R. Sauerbrey, U. Schramm, S. Trotsenko, and K. D. Ledingham, "Picosecond narrow bandwidth x-rays from laser-thomson-backscattering." Helmholtz-Zentrum Dresden-Rossendorf - to be submitted.

[35] F. Hartemann, H. Baldis, A. Kerman, A. Le Foll, N. Luhmann Jr, and B. Rupp, "Three-dimensional theory of emittance in compton scattering and x-ray protein crystallography," *Physical Review E*, vol. 64, no. 1, p. 016501, 2001.

[36] M. De Loos and S. Van der Geer, "General particle tracer: a new 3d code for accelerator and beam line design," in *Proc. 6th Europ Particle Accelerator Conference (EPAC)*, 1996.

[37] C. Thomas, J. Botman, C. Van der Geer, and M. Couprie, "Simulation of laser-electron beam interaction in the optical klystron of a free electron laser," in *Proc. of the Particle Accelerator Conference (EPAC)*, 2000.

[38] E. Esarey, B. Shadwick, P. Catravas, and W. Leemans, "Synchrotron radiation from electron beams in plasma-focusing channels," *Physical Review E*, vol. 65, no. 5, p. 056505, 2002.

[39] K. Phuoc, S. Corde, R. Shah, F. Albert, R. Fitour, J. Rousseau, F. Burgy, B. Mercier, and A. Rousse, "Imaging electron trajectories in a laser-wakefield cavity using betatron x-ray radiation," *Physical review letters*, vol. 97, no. 22, p. 225002, 2006.

[40] A. Rousse, K. Phuoc, R. Shah, A. Pukhov, E. Lefebvre, V. Malka, S. Kiselev, F. Burgy, J. Rousseau, D. Umstadter, *et al.*, "Production of a kev x-ray beam from synchrotron radiation in relativistic laser-plasma interaction," *Physical review letters*, vol. 93, no. 13, p. 135005, 2004.

[41] M. Geissler, *Interaction of High Power Laser Pulses with Plasmas.* PhD thesis, Max Planck Institut für Quantenoptik, 2000.

[42] H. W. Meuer, E. Strohmaier, J. Dongarra, and H. Simon, "Top500 List - November 2012." `http://www.top500.org/list/2012/11/`. accessed 13.11.2012.

[43] Jülich Supercomputing Centre (JSC), "Juelich Dedicated GPU Environment (JuDGE)." `http://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUDGE/JUDGE_node.html`. accessed 13.11.2012.

[44] J. Koga, T. Esirkepov, and S. Bulanov, "Nonlinear thomson scattering in the strong radiation damping regime," *Physics of plasmas*, vol. 12, p. 093106, 2005.

[45] D. Griffiths and R. College, *Introduction to electrodynamics*, vol. 3. prentice Hall New Jersey, 1999.

[46] D. Griffiths, T. Proctor, and D. Schroeter, "Abraham–Lorentz versus Landau–Lifshitz," *American Journal of Physics*, vol. 78, p. 391, 2010.

[47] T. Butz, *Fouriertransformation für Fußgänger.* Teubner Verlag, 5 ed., 2007.

[48] J. Cooley and J. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, pp. 297–301, 1965.

[49] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical recipes 2nd edition: The art of scientific computing.* Cambridge University Press, 1997.

[50] D. Vandevoorde and N. Josuttis, *C++ templates: the Complete Guide.* Addison-Wesley Professional, 2003.

[51] S. Hoffmann and R. Lienhart, *OpenMP: Eine Einführung in die parallele Programmierung mit C/C++.* Springer, 2009.

[52] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, pp. 114–117, April 1965.

[53] M. Flynn, "Some computer organizations and their effectiveness," *Computers, IEEE Transactions on*, vol. 100, no. 9, pp. 948–960, 1972.

[54] T. Rauber and G. Rünger, *Parallele Programmierung.* Springer, 2007.

[55] D. Kirk and W. Wen-mei, *Programming massively parallel processors: a hands-on approach.* Morgan Kaufmann, 2010.

Danksagung

Nach einem Jahr Arbeit ist es endlich geschafft. Die Diplomarbeit in ihrer jetzigen Form wäre nicht ohne die tatkräftige Unterstützung von Kollegen, Freunden und der Familie zustande gekommen.
Ich möchte mich ganz herzlich bei Professor Ulrich Schramm für die Möglichkeit bedanken, die Diplomarbeit am Institut für Strahlenphysik des HZDR anfertigen zu dürfen.
Nicht genug bedanken kann ich mich bei meinem Betreuer Dr. Michael Bussmann. Die Faszination mit der er über Physik spricht, lässt auch nach den anstrengenden Arbeitstagen die Freude an der Forschung wieder aufblühen. Er war mit unermüdlichen Einsatz immer für uns Studenten da, wenn wir Hilfe benötigten.
Ähnlich wertvoll war die Zusammenarbeit mit Dr. Alexander Debus, meinem inoffiziellen Betreuer. In langen Diskussion hat er mir viel erklärt und mit unzähligen Ratschlägen stets zur Seite gestanden. Sein unermüdlicher Einsatz, nicht nur beim Korrekturlesen, war eine große Hilfe. Vielen Dank Alex.
René Widera möchte ich dafür danken, dass er nie aufgegeben hat, einen besseren Programmierer aus mir zu machen. Er ließ mich in der faszinierenden Welt der Hochleistungsrechner nie allein und war immer ansprechbar für Fragen.
Besonderem Dank gebührt auch Klaus Steiniger. Da er mir gegenüber saß, musste er die meisten meiner Fragen ertragen. Trotz des ständigen Bombardements war er nie müde mir zu helfen oder über Probleme zu diskutieren.
Viel Unterstützung habe ich auch durch die anderen Studenten in unserem Büro erfahren, als da wären Omid Zarini, Leon Buikstra, Axel Hübl, Heiko Burau, Anton Helm und Malte Zacharias. Es war schön mit euch, den Spaß an der Arbeit teilen zu können.
Danken möchte ich auch Dr. Arie Irman und Axel Jochmann. Die Diskussion mit ihnen über die Emittanzmessungen an der ELBE waren sehr lehrreich.
Der rege Austausch mit Dr. Chengkun Huang erlaubte eine einzigartige Möglichkeit meine Arbeit zu verifizieren. Dafür möchte ich mich bei ihm bedanken.
Mit der Abgabe dieser Arbeit endet auch meine Zeit als Student. Es war schön diese wunderbare Zeit mit vielen Freunden zu erleben. Insbesondere möchte ich Johannes Petzoldt, Philip Dießner, Eric Jehnes, Fabian Patrovsky, Sascha Müller, Jacob Grasemann und Alexander Köhler danken.
Unzählig viel verdanke ich meinen Eltern Annette und Guntram Pausch. Sie haben mich immer unterstützt und mich motiviert wenn es nötig war.

Euch allen, vielen Dank für die Unterstützung!

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Richard Pausch
Dresden, Dezember 2012