Czech Technical University in Prague
Faculty of Nuclear Sciences and Physical Engineering
Department of Physical Electronics

Tight-focusing of short intense laser pulses in particle-in-cell simulations of laser-plasma interaction

(Master's thesis)

Author:	Bc. Petr Valenta
Supervisor:	doc. Ing. Ondřej Klimo, Ph.D.
Consultant:	Dr. Stefan Andreas Weber
Academic year:	2016/2017

Prohlášení/Declaration

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškerou použitou literaturu.

I hereby declare that I carried out this work independently, and only with the cited sources, literature and other professional sources.

V Praze dne/In Prague on ....

....
Bc. Petr Valenta

Název práce:	**Zaostření krátkého intenzivního laserového impulsu do velmi malého ohniska v částicových simulacích interakce s plazmatem**
Autor:	Bc. Petr Valenta
Druh práce:	Diplomová práce
Studijní program:	(N3913) Aplikace přírodních věd
Vedoucí práce: Obor:	(3901T065) Informatická fyzika
Vedoucí práce:	doc. Ing. Ondřej Klimo, Ph.D. Katedra fyzikální elektroniky, Fakulta jaderná a fyzikálně inženýrská, České vysoké učení technické v Praze
Konzultant:	Dr. Stefan Andreas Weber Projekt ELI-Beamlines, Fyzikální ústav Akademie věd České republiky, v. v. i.

Abstrakt

Content...

Klíčová slova: fokuzace, PIC simulace, laser-plazma interakce, optika

Title:	**Tight-focusing of short intense laser pulses in particle-in-cell simulations of laser-plasma interaction**
Author:	Bc. Petr Valenta
Type of work:	Master's thesis
Study programme:	(N3913) Aplications of Natural Sciences
Branch of study:	(3901T065) Computational Physics
Supervisor:	doc. Ing. Ondřej Klimo, Ph.D. Department of Physical Electronics, Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University in Prague
Consultant:	Dr. Stefan Andreas Weber Project ELI-Beamlines, Institute of Physics of the Czech Academy of Sciences

Abstract

Content...

Keywords: tight-focusing, PIC simulations, laser-plasma interaction, plasma optics

Contents

Introduction

Chapter 1

Electromagnetic field

Since the laser beam is nothing but the electromagnetic wave, the opening chapter has to be logically devoted to the fundamental physical aspects of the classical electromagnetic field theory based on the elegant Maxwell's equations. The reader can find a brief description of the microscopic as well as macroscopic variant of the Maxwell's equations and their general solutions exploiting electrodynamic potentials and Hertz vectors. A short part is devoted also to energy and momentum of the electromagnetic waves. In the last part, one finds the simplest mathematical description of the focused laser beam and conditions of its validity.

1.1 Maxwell's equations

The electromagnetic field is in general theory represented by two vectors, the intensity of the electric field $\mathbf{E}(\mathbf{r},t)$ and the magnetic induction $\mathbf{B}(\mathbf{r},t)$. These vectors are considered to be finite and continuous functions of position $\mathbf{r}$ and time t. The description of electromagnetic phenomena in classical electrodynamics is provided by the set of well-known Maxwell's equations. The microscopic variant for external sources in vacuum is formulated as follows,

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0}, \tag{1.1}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{1.2}$$

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = 0, \tag{1.3}$$

$$\nabla \times \mathbf{B} - \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} = \mu_0 \mathbf{J}, \tag{1.4}$$

where $\rho(\mathbf{r},t)$ is total electric charge density and $\mathbf{J}(\mathbf{r},t)$ is total electric current density, which is constituted by the motion of a charge. These distributions may be continuous as well as discrete. As might be seen from Maxwell's equations (1.1 - 1.4), the charge density is the source of the electric field, whilst the magnetic field is produced by the current density. The lack of symmetry in Maxwell's equations (1.2, 1.3 are homogeneous) is caused by the experimental absence of magnetic charges and currents. The universal constants appearing in the Maxwell's equations (1.1, 1.4) are the electric permittivity of vacuum ε_0 and the magnetic permeability of vacuum μ_0.

The first equation, 1.1, is Gauss's law for electric field in the differential form. It states that the flux of the electric field through any closed surface is proportional to the total charge inside. The second equation, 1.2, is Gauss's law for magnetic field. It expresses the fact that there are no magnetic monopoles, so the flux of magnetic field through any closed surface is always zero. The third equation, 1.3, is Faraday's law describing how the electric field is associated with a time varying magnetic field. And the last equation, 1.4, is Ampère's law with Maxwell's displacement current, which means that the time varying electric field causes the magnetic field. As a consequence, it predicts the existence of electromagnetic waves that can carry energy and momentum even in a free space.

To describe the effects of an electromagnetic field in the presence of macroscopic substances, the complicated distribution of charges and currents in matter over the atomic scale is not relevant. Thus one shall define a second set of auxiliary vectors that represent fields in which the material properties are already included in an average sense, the electric displacement $\mathbf{D}(\mathbf{r}, t)$ and the magnetic vector $\mathbf{H}(\mathbf{r}, t)$,

$$\mathbf{D} = \varepsilon_0 \mathbf{E} + \mathbf{P} = \varepsilon \mathbf{E}, \tag{1.5}$$

$$\mathbf{H} = \frac{\mathbf{B}}{\mu_0} - \mathbf{M} = \frac{\mathbf{B}}{\mu}, \tag{1.6}$$

where $\mathbf{P}(\mathbf{r}, t)$ and $\mathbf{M}(\mathbf{r}, t)$ are the vectors of polarization and magnetization, respectively. Note that the vectors of polarization and magnetization can be interpreted as a density of electric or magnetic dipole moment of the medium, therefore they are definitely associated with the state of a matter and vanish in vacuum. Similarly as in the case of free space, the factors ε and μ are called electric permittivity of medium and magnetic permeability of medium. In general case, ε and μ are tensors. The constitutive relations above (1.5, 1.6) hold only if the medium is homogeneous and isotropic. For the sake of simplicity, only such materials will be considered in the following text.

The macroscopic variant of Maxwell's equations is formulated as follows,

$$\nabla \cdot \mathbf{D} = \rho, \tag{1.7}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{1.8}$$

$$\nabla \times \mathbf{E} + \frac{\partial \mathbf{B}}{\partial t} = 0, \tag{1.9}$$

$$\nabla \times \mathbf{H} - \frac{\partial \mathbf{D}}{\partial t} = \mathbf{J}, \tag{1.10}$$

where $\rho(\mathbf{r}, t)$ and $\mathbf{J}(\mathbf{r}, t)$ now stand for only external electric charge and current density, respectively.

By combining the time derivative of the equation 1.7 with the divergence of the equation 1.10, one obtains the following relation between the electromagnetic field sources,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J} = 0. \tag{1.11}$$

The important result 1.11, which is frequently referred to as the equation of continuity, expresses nothing but the conservation of total electric charge in an isolated system. In other words, the time rate of change of the electric charge in any closed surface is balanced by the electric current flowing through the surface.

1.2 Electrodynamic potentials

The first-order partial differential Maxwell's equations can be effectively converted to a smaller number of second-order equations by introducing electrodynamic potentials. Hence, one can express the electric and magnetic field as follows,

$$\mathbf{E} = -\nabla \Phi - \frac{\partial \mathbf{A}}{\partial t}, \tag{1.12}$$

$$\mathbf{B} = \nabla \times \mathbf{A}, \tag{1.13}$$

where $\Phi(\mathbf{r}, t)$ is the scalar potential and $\mathbf{A}(\mathbf{r}, t)$ is the vector potential of the corresponding fields. One can clearly see that using the definitions 1.12, 1.13, six vector components are replaced by only four potential functions and two Maxwell's homogeneous equations (1.8, 1.9) are fulfilled identically.

However, by definitions 1.12, 1.13, $\Phi(\mathbf{r}, t)$ and $\mathbf{A}(\mathbf{r}, t)$ are not defined uniquely, thus an infinite number of potentials which lead to the same fields may be constructed. To avoid that, one has to impose a supplementary condition, for example

$$\nabla \cdot \mathbf{A} + \mu\varepsilon \frac{\partial \Phi}{\partial t} = 0. \tag{1.14}$$

The condition 1.14 is called the Lorenz gauge. Lorenz gauge is commonly used in electromagnetism because its independence of the coordinate system. Furthermore, it leads to the following uncoupled equations,

$$\Delta \Phi - \mu\varepsilon \frac{\partial^2 \Phi}{\partial t^2} = -\frac{\rho}{\varepsilon}, \tag{1.15}$$

$$\Delta \mathbf{A} - \mu\varepsilon \frac{\partial^2 \mathbf{A}}{\partial t^2} = -\mu \mathbf{J}, \tag{1.16}$$

that are in all respects equivalent to the Maxwell's equations and in many situations much simpler to solve.

Equations 1.15, 1.16 correspond to the inhomogeneous wave equations for scalar potential $\Phi(\mathbf{r}, t)$ and vector potential $\mathbf{A}(\mathbf{r}, t)$. Their general solutions are given by the following expressions,

$$\Phi(\mathbf{r}, t) = \frac{1}{4\pi\varepsilon} \int \frac{\rho(\mathbf{r}', t')}{\|\mathbf{r} - \mathbf{r}'\|} \mathrm{d}V, \tag{1.17}$$

$$\mathbf{A}(\mathbf{r}, t) = \frac{\mu}{4\pi} \int \frac{\mathbf{J}(\mathbf{r}', t')}{\|\mathbf{r} - \mathbf{r}'\|} \mathrm{d}V, \tag{1.18}$$

where $\mathrm{d}V$ is a volume element and $\|.\|$ stands for the standard Euclidean norm. Note that the solutions 1.17, 1.18 are dependent only on charge and current densities at position $\mathbf{r}'$ at so-called retarded time $t' = t - \sqrt{\mu\epsilon}\|\mathbf{r} - \mathbf{r}'\|$ which takes into account the finite velocity of the wave. In other words, the fields at the observation point $\mathbf{r}$ at the time t are proportional to the sum of all the electromagnetic waves that leave the source elements at point $\mathbf{r}'$ at the retarded time t'.

1.3 Hertz vectors

There exists also other possibilities how to express the electromagnetic field. Under ordinary conditions, an arbitrary electromagnetic field may be defined in terms of a single vector function. This may be helpful for solving of many problems of classical electromagnetic theory, particularly the wave propagation.

First, let us introduce the electric Hertz vector $\mathbf{\Pi_e}(\mathbf{r}, t)$ in terms of the scalar and vector potentials,

$$\Phi = -\nabla \cdot \mathbf{\Pi_e}, \tag{1.19}$$

$$\mathbf{A} = \mu\varepsilon \frac{\partial \mathbf{\Pi_e}}{\partial t}. \tag{1.20}$$

Note that the definitions 1.19, 1.20 are consistent with the Lorenz gauge condition 1.14. In the absence of magnetization, it might be easily shown that $\mathbf{J} = \partial \mathbf{P}/\partial t$ and the electric Hertz vector $\mathbf{\Pi_e}(\mathbf{r}, t)$ is governed by an inhomogeneous wave equation

$$\Delta \mathbf{\Pi_e} - \mu\varepsilon \frac{\partial^2 \mathbf{\Pi_e}}{\partial t^2} = -\frac{\mathbf{P}}{\varepsilon}. \tag{1.21}$$

The equation 1.21 is of the same type as the equations 1.15, 1.16 and has therefore the familiar general solution

$$\mathbf{\Pi_e}(\mathbf{r}, t) = \frac{1}{4\pi\varepsilon} \int \frac{\mathbf{P}(\mathbf{r}', t')}{\|\mathbf{r} - \mathbf{r}'\|} \mathrm{d}V. \tag{1.22}$$

As might be seen form 1.22, the fields derived from the electric Hertz vector $\mathbf{\Pi_e}(\mathbf{r}, t)$

can be interpreted as being due to a density distribution of electric dipoles. Every solution of 1.22 then uniquely determines the electromagnetic field through

$$\mathbf{E} = \nabla\left(\nabla \cdot \mathbf{\Pi_e}\right) - \mu\epsilon \frac{\partial^2 \mathbf{\Pi_e}}{\partial t^2}, \tag{1.23}$$

$$\mathbf{B} = \mu\varepsilon\left(\nabla \times \frac{\partial \mathbf{\Pi_e}}{\partial t}\right). \tag{1.24}$$

Second, one may introduce the magnetic Hertz vector $\mathbf{\Pi_m}\left(\mathbf{r}, t\right)$ in terms of the scalar and vector potentials by the following expressions,

$$\Phi = 0, \tag{1.25}$$

$$\mathbf{A} = \nabla \times \mathbf{\Pi_m}. \tag{1.26}$$

In the absence of polarization, $\mathbf{J} = \nabla \times \mathbf{M}$ and the magnetic Hertz vector $\mathbf{\Pi_m}\left(\mathbf{r}, t\right)$ defined by 1.25 and 1.26 fulfills an inhomogeneous wave equation

$$\Delta\mathbf{\Pi_m} - \mu\varepsilon \frac{\partial^2 \mathbf{\Pi_m}}{\partial t^2} = -\mu\mathbf{M}. \tag{1.27}$$

As for the previous cases, one may easily find the solution of 1.27,

$$\mathbf{\Pi_m}\left(\mathbf{r}, t\right) = \frac{\mu}{4\pi}\int \frac{\mathbf{M}\left(\mathbf{r}', t'\right)}{\|\mathbf{r} - \mathbf{r}'\|}\mathrm{d}V, \tag{1.28}$$

thus the fields derived from the magnetic Hertz vector $\mathbf{\Pi_m}\left(\mathbf{r}, t\right)$ may be imagined to be due to a density distribution of magnetic dipoles. Again, every solution of 1.28 uniquely determines the electromagnetic field via

$$\mathbf{E} = \nabla \times \frac{\partial \mathbf{\Pi_m}}{\partial t}, \tag{1.29}$$

$$\mathbf{B} = \nabla \times \left(\nabla \times \mathbf{\Pi_m}\right). \tag{1.30}$$

Note that the above derivations considered electric and magnetic Hertz vectors as a separate quantities. It is also possible, however, to introduce them together in the form of one six-vector [source].

1.4 Energy and momentum

To be able to describe the interaction of the electromagnetic field with matter, one has to know the energy distribution throughout the field as well as the momentum balance.

By scalar multiplications of 1.9 by $\mathbf{H}(\mathbf{r},t)$, of 1.10 by $\mathbf{E}(\mathbf{r},t)$, following subtraction of both obtained equations and using standard vector identities, one gets the expression

$$\mathbf{E}\cdot\frac{\partial\mathbf{D}}{\partial t}+\mathbf{H}\cdot\frac{\partial\mathbf{B}}{\partial t}+\nabla\cdot(\mathbf{E}\times\mathbf{H})=-\mathbf{E}\cdot\mathbf{J}. \tag{1.31}$$

The equation 1.31 can be rewritten in the form of conservation law,

$$\frac{\partial u}{\partial t}+\nabla\cdot\mathbf{S}=-\mathbf{E}\cdot\mathbf{J}, \tag{1.32}$$

where

$$u=\frac{1}{2}(\mathbf{E}\cdot\mathbf{D}+\mathbf{H}\cdot\mathbf{B}),\quad \mathbf{S}=\mathbf{E}\times\mathbf{H}. \tag{1.33}$$

The quantity $u(\mathbf{r},t)$ in 1.33 describes the total energy density in the field and $\mathbf{S}(\mathbf{r},t)$ is so-called Poynting vector which represents, in the equation 1.32, the energy flow of the field per unit area. Note that the Poynting vector points in the same direction as the vector of the wave propagation.

The important statement 1.32, also referred to as the Poynting theorem, expresses the energy balance for the electromagnetic field. In other words, the time rate of change of the field energy within a certain region and the energy flowing out of that region is balanced by the conversion of the electromagnetic energy into mechanical or heat energy.

Besides energy, the electromagnetic wave can carry also momentum. To derive a balance of linear momentum, one has to know the way how charges and currents interact with the electromagnetic field. This is described by the Lorentz force which density $\mathbf{f}_{\mathrm{L}}(\mathbf{r},t)$ is given by the following expression,

$$\mathbf{f}_{\mathrm{L}}=\rho\mathbf{E}+\mathbf{J}\times\mathbf{B}. \tag{1.34}$$

Thus the electric and magnetic fields can be regarded as a forces produced by distribution of charge and currents.

By replacing sources in 1.34 using macroscopic Maxwell's equations 1.7, 1.10, one may express the density of Lorentz force $\mathbf{f}_{\mathrm{L}}(\mathbf{r},t)$ entirely in terms of fields,

$$\mathbf{f}_{\mathrm{L}}=(\nabla\cdot\mathbf{D})\,\mathbf{E}+(\nabla\times\mathbf{H})\times\mathbf{B}-\frac{\partial\mathbf{D}}{\partial t}\times\mathbf{B}. \tag{1.35}$$

Note that the last term on the right hand side of 1.35 can be rewritten using the Poynting vector $\mathbf{S}(\mathbf{r},t)$ derived above,

$$\frac{\partial\mathbf{D}}{\partial t}\times\mathbf{B}=\varepsilon\mu\frac{\partial\mathbf{S}}{\partial t}+\mathbf{D}\times(\nabla\times\mathbf{E}). \tag{1.36}$$

By plugging 1.36 into 1.35 and employing basic vector calculus identities, one eventually gets the expression for the linear momentum balance in the form of conservation law,

$$\frac{\partial \mathbf{g}}{\partial t} + \nabla \cdot \mathbb{T} = -\mathbf{f}_{\mathrm{L}}, \tag{1.37}$$

where

$$\mathbf{g} = \mathbf{D} \times \mathbf{B}, \qquad T_{ij} = -E_i D_j - H_i B_j + \frac{1}{2} \left(\mathbf{E} \cdot \mathbf{D} + \mathbf{H} \cdot \mathbf{B} \right) \delta_{ij}. \tag{1.38}$$

The quantity $\mathbf{g}(\mathbf{r}, t)$ in 1.37 may be interpreted as the density of linear electromagnetic momentum and $\mathbb{T}(\mathbf{r}, t)$ is so-called Maxwell stress tensor which represents, in the equation 1.37, the momentum flow per unit area. The symbol δ_{ij} in 1.38 stands for the Kronecker delta.

The important statement 1.37, which expresses the linear momentum balance for electromagnetic field, may be used to calculate the electromagnetic forces that act on objects or particles within that field. Notice that the contribution of the electromagnetic field to energy and momentum is completely characterized by the fluxes of $\mathbf{S}(\mathbf{r}, t)$ and $\mathbb{T}(\mathbf{r}, t)$.

1.5 Electromagnetic waves and Gaussian beam

In this section, the simplest mathematical description of a focused laser beam based on approximations to the wave equation is deduced. Since in numerical codes it is a common practice to prescribe the laser beams by their propagation in free space, the set of the microscopic Maxwell's equations 1.1 - 1.4 will be exploited.

In the absence of external sources, it might be easily shown that the equations 1.1 - 1.4 may be alternatively formulated as an uncoupled homogeneous wave equations for electric field $\mathbf{E}(\mathbf{r}, t)$ and magnetic field $\mathbf{B}(\mathbf{r}, t)$,

$$\Delta \mathbf{E} - \frac{1}{c^2} \frac{\partial^2 \mathbf{E}}{\partial t^2} = 0, \tag{1.39}$$

$$\Delta \mathbf{B} - \frac{1}{c^2} \frac{\partial^2 \mathbf{B}}{\partial t^2} = 0, \tag{1.40}$$

where the universal constant $c = 1/\sqrt{\mu_0 \varepsilon_0}$ is the speed of light in vacuum, which leads to the essential fact, that the electromagnetic waves propagate in vacuum with the velocity of light c. However, the wave equations 1.39, 1.40 do not provide all the information about the electric and magnetic field of the wave. There are further constraints due to Maxwell's equations restricting the orientation and proportional magnitudes of the fields. From the set 1.1 - 1.4, it might be clearly seen that $\mathbf{E}(\mathbf{r}, t)$ and $\mathbf{B}(\mathbf{r}, t)$ must be mutually perpendicular to each other as well as to the direction of the wave propagation.

Without any loss of generality, consider the laser beam as a monochromatic electromagnetic wave propagating toward the positive direction of the z-axis. In many standard references [source], the description of such a wave is given by the evolution of a single electric field component linearly polarized along the x-axis of the Cartesian coordinate system (although the more proper way would be to use the vector potential [source]), therefore one has to look for the solution of the equation 1.39.

According to the previous assumptions, the solution is expected to be in the form of the following plane wave,

$$\mathbf{E}\left(\mathbf{r}_{\perp}, z, t\right) = \Re\ E_0 \Psi\left(\mathbf{r}_{\perp}, z\right) \mathrm{e}^{\mathrm{i}\left(k_z z - \omega t\right)} \hat{\mathbf{e}}_{\mathbf{x}}, \tag{1.41}$$

where $\mathbf{r}_{\perp} = (x, y)^{\mathrm{T}}$ is the vector of transverse Cartesian coordinates, symbol $\Re$ stands for the real part of the complex quantity, E_0 is a constant amplitude, $\Psi\left(\mathbf{r}_{\perp}, z\right)$ is the part of the wave function which is dependent only on the spatial coordinates, ω denotes the angular frequency, k_z is the z-component of the wave vector $\mathbf{k}\left(\omega\right)$, i denotes the imaginary unit and $\hat{\mathbf{e}}_{\mathbf{x}}$ is the unit vector pointing in the direction of the x-axis.

Direct substitution of expression 1.41 into the equation 1.39 yields the time-independent form of the scalar wave equation

$$\Delta \Psi\left(\mathbf{r}_{\perp}, z\right) + 2\mathrm{i} k_z \frac{\partial \Psi\left(\mathbf{r}_{\perp}, z\right)}{\partial z} = 0. \tag{1.42}$$

The equation 1.42 is called the Helmholtz equation. Note that it is sufficient to seek solutions to the equation 1.42 since the wave 1.41 is monochromatic.

It turned out, that the geometry of the focused laser beam can be expressed in terms of the laser wavelength λ and the following three parameters,

$$w_0, \qquad z_{\mathrm{R}} = \frac{k_z w_0^2}{2} = \frac{\pi w_0^2}{\lambda}, \qquad \Theta = \frac{w_0}{z_{\mathrm{R}}} = \frac{\lambda}{\pi w_0}. \tag{1.43}$$

The parameter w_0 in 1.43 is the beam waist, defined as a radius at which the laser intensity fall to $1/\mathrm{e}^2$ of its axial value at the focal spot. The second parameter, z_{R}, is so-called Rayleigh range which is a distance in the longitudinal direction from the focal spot to the point where the beam radius is $\sqrt{2}$ larger than the beam waist w_0. And the last parameter, Θ, is the divergence angle of the beam that represents the ratio of transverse and longitudinal extent.

Because of the symmetry about the longitudinal axis of the equation 1.42, the following calculations may be made simpler by introducing a dimensionless cylindrical coordinates that use the parameters 1.43,

$$\rho = \frac{\|\mathbf{r}_{\perp}\|}{w_0}, \qquad \zeta = \frac{z}{z_{\mathrm{R}}}. \tag{1.44}$$

After performing a transformation of coordinates, the Helmholtz equation 1.42 becomes

$$\frac{1}{\rho}\frac{\partial}{\partial \rho}\left(\rho \frac{\partial \Psi\left(\rho, \zeta\right)}{\partial \rho}\right) + 4\mathrm{i} \frac{\partial \Psi\left(\rho, \zeta\right)}{\partial \zeta} = -\Theta^2 \frac{\partial^2 \Psi\left(\rho, \zeta\right)}{\partial \zeta^2}. \tag{1.45}$$

In the following calculations, the beam divergence angle Θ is assumed to be small ($\Theta \ll 1$), thus it can be used as an expansion parameter for Ψ and the solution of 1.45 will always be consistent,

$$\Psi = \sum_{n=0}^{+\infty} \Theta^{2n} \Psi_{2n}. \tag{1.46}$$

Next, one shall insert 1.46 into 1.45 and collect the terms with the same power of Θ. Then the zeroth-order function Ψ_0 obeys the following equation,

$$\frac{1}{\rho}\frac{\partial}{\partial\rho}\left(\rho\frac{\partial\Psi_0(\rho,\zeta)}{\partial\rho}\right) + 4\mathrm{i}\frac{\partial\Psi_0(\rho,\zeta)}{\partial\zeta} = 0. \tag{1.47}$$

The equation 1.47, which is called the paraxial Helmholtz equation, is the starting point of traditional Gaussian beam theory. One can expect the solution of 1.47 in the form of a Gaussian function with a width varying along the longitudinal direction, thus

$$\Psi_0(\rho,\zeta) = h(\zeta)\,\mathrm{e}^{-f(\zeta)\rho^2}, \tag{1.48}$$

where $f(\zeta)$ and $h(\zeta)$ are unknown complex functions that have to satisfy a condition $f(0) = h(0) = 1$. After plugging 1.48 into 1.47, one gets the following equation,

$$-f(\zeta)\,h(\zeta) + \mathrm{i}\frac{\mathrm{d}h(\zeta)}{\mathrm{d}\zeta} + \rho^2 h(\zeta)\left(f(\zeta)^2 - \mathrm{i}\frac{\mathrm{d}f(\zeta)}{\mathrm{d}\zeta}\right) = 0. \tag{1.49}$$

Since the equation 1.49 has to hold for arbitrary value of ρ, one may find two independent equations that are equivalent to 1.49

$$\frac{1}{f(\zeta)^2}\frac{\mathrm{d}f(\zeta)}{\mathrm{d}\zeta} + \mathrm{i} = 0, \qquad \frac{1}{f(\zeta)\,h(\zeta)}\frac{\mathrm{d}h(\zeta)}{\mathrm{d}\zeta} + \mathrm{i} = 0. \tag{1.50}$$

It might be easily shown, that under specified conditions the solutions of equations 1.50 have to be

$$h(\zeta) = f(\zeta), \qquad f(\zeta) = \frac{1}{\sqrt{1+\zeta^2}}\mathrm{e}^{-\mathrm{i}\arctan\zeta}, \tag{1.51}$$

and therefore the complete expression for the zeroth-order wave function $\Psi_0(\rho,\zeta)$ is

$$\Psi_0(\rho,\zeta) = \frac{1}{\sqrt{1+\zeta^2}}\exp\left[-\frac{\rho^2}{1+\zeta^2} + \mathrm{i}\left(\frac{\rho^2\zeta}{1+\zeta^2} - \arctan\zeta\right)\right]. \tag{1.52}$$

In many situations, it is also useful to evaluate the expression 1.52 in terms of Cartesian coordinates, in which the zeroth-order wave function $\Psi_0(\mathbf{r}_\perp, z)$ is

$$\Psi_0(\mathbf{r}_\perp, z) = \frac{w_0}{w(z)}\exp\left[-\frac{{\mathbf{r}_\perp}^2}{w(z)^2} + \mathrm{i}\left(k_z\frac{{\mathbf{r}_\perp}^2}{2R(z)} - \varphi_{\mathrm{G}}(z)\right)\right], \tag{1.53}$$

where the parameters used to simplify the expression 1.53 are defined as

$$w\left(z\right)=w_0\sqrt{1+\left(\frac{z}{z_\mathrm{R}}\right)^2},\quad R\left(z\right)=z\left[1+\left(\frac{z_\mathrm{R}}{z}\right)^2\right],\quad \varphi_\mathrm{G}\left(z\right)=\arctan\left(\frac{z}{z_\mathrm{R}}\right). \tag{1.54}$$

One shall discuss the physical meaning of the three parameters 1.54. The function $w\left(z\right)$ represents the spot size parameter of the beam, that is the radius at which the laser intensity fall to $1/\mathrm{e}^2$ of its axial value at any position z along the beam propagation. Note that the minimum of the spot size $w(0)=w_0$, consequently the focal spot is stationary and located at the origin of a Cartesian coordinate system. The second parameter, $R\left(z\right)$ is known to be the radius of curvature of the beam's wavefront at any position z along the beam propagation. Note that $\lim_{z\to 0^\pm} R(z)=\pm\infty$, therefore the beam behaves like a plane wave at focus as required. The last parameter, $\varphi_\mathrm{G}\left(z\right)$, is the so-called Guoy phase of the beam at any position z along the beam propagation, which describes a phase shift in the wave as it passes through the focal spot.

Finally, by substituting 1.53 for $\Psi\left(\mathbf{r}_\perp, z\right)$ in 1.41 and taking the real part of that complex quantity, one obtains the electric field of the so-called paraxial Gaussian beam,

$$\mathbf{E}\left(\mathbf{r}_\perp, z, t\right)=E_0\frac{w_0}{w(z)}\exp\left(-\frac{{\mathbf{r}_\perp}^2}{w(z)^2}\right)\cos\left(\omega t-k_z\left(z+\frac{{\mathbf{r}_\perp}^2}{2R(z)}\right)+\varphi_\mathrm{G}\left(z\right)\right)\hat{\mathbf{e}}_\mathbf{x}. \tag{1.55}$$

Although given electric field 1.55 describes the main features of the focused laser beam, it might be clearly seen that it does not satisfy Gauss's law (1.1). The correct electric field cannot vary with the direction of its polarization or has to have at least two non-zero vector components. To fix that, one would have to solve the wave equation for the vector potential 1.16 and afterwards exploit the solution to deduce all components of the electric and magnetic fields.

In addition, since one assumed $\Theta \ll 1$, the solution 1.55 is not accurate for strongly diverging beams. Since the divergence angle is inversely proportional to the beam waist, the previous condition yields $w_0 \gg \lambda$. In other words, it means that 1.55 is not valid for tightly focused laser beams and the need may arise for higher-order corrections.

Chapter 2

Laser-plasma interaction

When a high-power laser pulse is focused onto the surface of a solid target, a high density plasma layer is produced almost immediately due to the presence of strong electromagnetic fields. The whole plasma region, which expands at sonic velocities, is dominated by laser-plasma interactions. The laser-plasma interactions offer an environment full of non-linear processes that take place during the propagation of the laser light through the plasma. In this chapter, a brief introduction to this field of research, which is rich both in physics and in applications, is provided.

2.1 Basic plasma parameters

A plasma, one of the four fundamental states of matter, is a quasi-neutral gas of charged and neutral particles which exhibits collective behavior [source]. It is necessary to closer explain some terms used in this definition.

By collective behavior one means motions that depend not only on local conditions but on the state of the plasma in remote regions as well. As charged particles move around, they can generate local concentrations of positive or negative charge, which give rise to electric fields. Motion of charges also generates currents, and hence magnetic fields. These fields affect the motion of other charged particles far away. Thus, the plasma gets a wide range of possible motions.

Quasi-neutrality describes the apparent charge neutrality of a plasma over large volumes, while at smaller scales, there can be charge imbalance, which may give rise to local electric fields. This fact can be expressed mathematically as

$$\sum_s q_s n_s \approx 0, \tag{2.1}$$

where q_s and n_s is, respectively, the charge and density of particles of species s. The index of summation is taken over all the particle species of given system.

One of the most important parameters, which allows to predict the behavior of plasmas more accurately, is the degree of its ionization. For a gas containing only single atomic species in thermodynamic equilibrium, the ionization can be clearly recognized from the Saha-Langmuir equation, which is most commonly written in the following form,

$$\frac{n_{k+1}}{n_k} = \frac{2}{n_e h^3} (2\pi m_e k_B T)^{\frac{3}{2}} \frac{g_{k+1}}{g_k} \exp\left(-\frac{\varepsilon_{k+1} - \varepsilon_k}{k_B T}\right). \tag{2.2}$$

Here n_k is the density of atoms in the k-th state of ionization, n_e is the electron density, m_e stands for the mass of electron, k_B is Boltzmann's constant, T is the gas temperature, h is Planck's constant, g_k is the degeneracy of the energy level for ions in the k-th state and ε_k is the ionization energy of the k-th level. From the equation (2.2), it may be clearly seen that the fully ionized plasmas exist only at high temperatures. That is the main reason why plasmas do not occur naturally on Earth (with a few exceptions).

A fundamental characteristics of the plasma behavior is its ability to shield out the electric potentials that are applied to it. Therefore, another important quantity λ_{Ds} which is called the Debye length of species s is established,

$$\lambda_{Ds} = \sqrt{\frac{\varepsilon_0 k_B T_s}{q_s^2 n_s}}. \tag{2.3}$$

The physical constant T_s denotes the temperature of the particles of species s. It often happens that a different species of particles in plasma have separate distributions with different temperatures, although each species can be in its own thermal equilibrium. The Debye length is a measure of the shielding distance or thickness of the sheath.

In plasma, each particle tries to gather its own shielding cloud. The previously mentioned concept of Debye shielding is valid only if there are enough particles in that cloud. Therefore, another important dimensionless number N_{Ds}, which is called plasma parameter of species s, is established. Definition of this parameter is given by the average number of particles of species s in a plasma contained within a sphere of radius of the Debye length, thus

$$N_{Ds} = \frac{4}{3}\pi n_s \lambda_{Ds}^3. \tag{2.4}$$

A typical harmonic electrostatic oscillations of charged particles in a response to a charge separation in plasma is described by the plasma frequency. Clearly, one may define the plasma frequency for particles of a general species s,

$$\omega_{ps} = \sqrt{\frac{q_s^2 n_s}{\varepsilon_0 m_s}}, \tag{2.5}$$

Note that as the ions are much heavier than electrons, they do not respond to the high frequency oscillations of the laser electric field. In many of the phenomena described in this chapter, the ions are often treated as an immobile, uniform, neutralizing background. However, if the frequency of external radiation source or the waves induced in plasmas is close to this frequency, the ion motion must also be included. An example may be a stimulated Brillouin scattering.

A typical charged particle in a plasma simultaneously undergo Coulomb collisions with all of the other particles in the plasma. The importance of collisions is contained in an expression called the collision frequency ν_c, which is defined as the inverse of the mean time that it takes for a particle to suffer a collision. Relatively accurate calculation of electron-ion collision frequency ν_{ei} can be obtained from the following relation,

$$\nu_{ei} = \frac{Ze^4 n_e}{4\pi\varepsilon_0^2 m_e^2 v^3} \ln\Lambda, \qquad \Lambda = \frac{\lambda_D}{b_0}. \tag{2.6}$$

The coefficient Z denotes the charge number, v is relative velocity of colliding particles and $\ln\Lambda$ is the so-called Coulomb logarithm. It is ratio of the Debye to Landau length. Landau length b_0 is the impact parameter at which the scattering angle in the center of mass frame is 90°. For many plasmas of interest Coulomb logarithm takes on values between $5-15$. In a plasma a Coulomb collision rarely results in a large deflection. The cumulative effect of the many random small angle collisions that it suffers, however, is often larger than the effect of the few large angle collisions. Notice that the collision frequency ν_{ei} is proportional to v^{-3}, therefore the effect of collisions in hot plasmas is usually weak.

In a constant and uniform magnetic field, one can find that a charged particle spirals in a helix about the line of force. This helix, however, defines a fundamental time unit and distance scale,

$$\omega_{cs} = \frac{|q_s|\|\mathbf{B}\|}{m_s}, \qquad r_{Ls} = \frac{v_\perp}{\omega_{cs}}. \tag{2.7}$$

These are called the cyclotron frequency ω_{cs} and the Larmor radius r_{Ls} of species s. Here $v_\perp$ is a positive constant denoting the speed in the plane perpendicular to $\mathbf{B}$.

2.2 Plasma description

Currently, there exist three different approaches to plasma physics. The particle theory, the kinetic theory and the hydrodynamic theory. Each approach has some advantages and limitations that stem from the simplified assumptions appropriate only for certain phenomena and time scales. Note that the coupling with the Maxwell's equations is usually straightforward, therefore the corresponding relations are not mentioned here. In the following three subsections, all the mentioned approaches are briefly discussed.

2.2.1 Particle theory

The time evolution of the system containing charged particles in plasma is influenced by the electromagnetic fields formed due to the motion of particles as well as the external fields (e.g. laser). The particle approach is based on solving the equations of motion for all the particles in the system. In this case, the motion of particles is governed by the Newton's equations with the Lorentz force,

$$m_s \frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = q_s \left(\mathbf{E} + \mathbf{v} \times \mathbf{B} \right). \tag{2.8}$$

The particle theory is useful for tracking a single particle motion in prescribed electromagnetic field. Obviously, the general solution for the system of charged particles using this approach can be complicated since the plasma typically consist of a large number of such particles interacting in a self-consistent fields. However, the full analysis for certain cases may be applied with the help of computing infrastructures and particle simulation codes.

2.2.2 Kinetic theory

The plasma kinetic theory takes into account the motion of all charged particles in the system as well. However, the evolution of such system is not described by the exact motion of the particles, but only via certain average properties. The kinetic theory is based on a set of equations for the distribution function $f_s(\mathbf{x}, \mathbf{v}, t)$ of particles of species s in plasma. The distribution function may be interpreted as a statistical description of a large number of interacting particles in the system. If collisions can be neglected (for example in hot plasmas), the distribution function is governed by the collisionless Vlasov equation,

$$\frac{\partial f_s}{\partial t} + \mathbf{v} \cdot \nabla f_s + \frac{q_s}{m_s} \left(\mathbf{E} + \mathbf{v} \times \mathbf{B} \right) \cdot \frac{\partial f_s}{\partial \mathbf{v}} = 0. \tag{2.9}$$

The Vlasov equation 2.9 is obtained only by making the assumption that the particle density is conserved, such that the time rate of change in a phase-space volume is equal to the flux of particles in or out of that volume. Due to its simplicity, the equation 2.9 is probably the most commonly used equation in the kinetic theory. However, the assumption to neglect collisions in a plasma is not valid generally. If it is necessary to take them into account, the collision term can be approximated using several methods [source].

2.2.3 Hydrodynamic theory

In spite of the fact that the hydrodynamic theory is the roughest approximation for the description of plasma, it is sufficiently accurate to describe the majority of observed phenomena. The velocity distribution of each species is assumed to be Maxwellian everywhere, so the dependent variables are functions of only space and time coordinates. In other words, the fluid equations are the first three moments of the Vlasov equation (2.9). These yield the following hydrodynamic equations for the density, momentum and the energy,

$$\frac{\partial n_s}{\partial t} + \nabla \cdot (n_s \mathbf{u}_s) = 0, \tag{2.10}$$

$$m_s n_s \left[\frac{\partial \mathbf{u}_s}{\partial t} + (\mathbf{u}_s \cdot \nabla) \mathbf{u}_s \right] + \nabla \cdot \mathbb{P}_s = q_s n_s (\mathbf{E} + \mathbf{u}_s \times \mathbf{B}), \tag{2.11}$$

$$\frac{\partial}{\partial t} \left(\frac{1}{2} n_s m_s u_s^2 + e_s \right) + \nabla \cdot \left(\frac{1}{2} n_s m_s u_s^2 \mathbf{u}_s + e_s \mathbf{u}_s + \mathbb{P}_s \mathbf{u}_s + \mathbf{Q}_s \right) = q_s n_s \mathbf{u}_s \cdot \mathbf{E}. \tag{2.12}$$

The zeroth-order moment (2.10) yields the continuity equation, where $\mathbf{u}_s(\mathbf{x}, t)$ is the velocity of the fluid of species s. This equation essentially states that the total number of particles is conserved. The first-order moment (2.11) leads to a momentum equation. Here $\mathbb{P}_s(\mathbf{x}, t)$ is the pressure tensor. This comes about by separating the particle velocity into the fluid and a thermal component of velocity. The thermal velocity then leads to the pressure term. Finally, the second-order moment (2.12) corresponds to the energy equation, where e_s is the density of the internal energy and $\mathbf{Q}_s$ describes the heat flux density.

It might be clearly seen that the moment equations form an infinite set. Thus, in order to solve these equations, a truncation is required. For the system of equations 2.10 - 2.12 to be complete, it has to be supplemented by the equation of state, which describes the relation between pressure and density in the plasma. However, the equations of state are well defined only in local thermodynamic equilibrium. Otherwise, the system cannot provide sufficiently exact description and thus the fluid equations 2.10 - 2.12 are not appropriate. Note that if the magnetic field is dominant in plasma, the set of hydrodynamic equations 2.10 - 2.12 may be effectively replaced by the equations of magnetohydrodynamics [source].

2.3 Electromagnetic waves in plasmas

In laser-plasma interaction, the knowledge how the laser beam propagates through plasma is essential. Therefore, a general properties of the electromagnetic wave propagation in plasmas are closer described in this section. One can find the characteristics of propagation in both, unmagnetized as well as magnetized plasma. Particularly, in the case of magnetized plasmas, the waves traveling parallel to and perpendicular to a constant magnetic field are discussed in greater detail.

2.3.1 Unmagnetized plasmas

To derive the dispersion relation of electromagnetic wave propagating through unmagnetized plasmas, the hydrodynamic approach is exploited. Since one assume plasma response to a high frequency field, the ions are treated as a stationary, neutralizing background. Thermal motion of particles is also ignored, thus the pressure term in 2.11 can be neglected. According to the previous assumptions, one shall solve the following set of equations,

$$\frac{\partial n_e}{\partial t} + \nabla \cdot (n_e \mathbf{u}_e) = 0, \tag{2.13}$$

$$m_e n_e \left[\frac{\partial \mathbf{u}_e}{\partial t} + (\mathbf{u}_e \cdot \nabla) \mathbf{u}_e \right] = -e n_e \mathbf{E}. \tag{2.14}$$

One also needs the wave equation for the electric field 1.39. However, it is now necessary to include the current density due to motion of charged particles in plasma. The hydrodynamic equations are coupled with the Maxwell's equations via $\mathbf{J} = q_s n_s \mathbf{u}_s$, thus the wave equation yields the following form,

$$\Delta \mathbf{E} - \frac{1}{c^2} \frac{\partial^2 \mathbf{E}}{\partial t^2} = -\mu_0 e n_e \frac{\partial \mathbf{u}_e}{\partial t}, \tag{2.15}$$

The system of equations above will be linearized using the methods of perturbation theory. Consider a small perturbations from the stationary state,

$$n_e = n_{e0} + \delta n_e, \quad \mathbf{u}_e = \mathbf{u}_{e0} + \delta \mathbf{u}_e, \quad \mathbf{E} = \mathbf{E}_0 + \delta \mathbf{E}, \tag{2.16}$$

where $\mathbf{u}_{e0}$ and $\mathbf{E}_0$ are obviously identically equal to zero vector. After substituting perturbed quantities 2.16 into initial system of equations and performing Fourier transform one obtains

$$\delta n_e = \mathrm{i} \frac{n_{e0}}{\omega} \mathbf{k} \cdot \delta \mathbf{u}_e, \tag{2.17}$$

$$\delta \mathbf{u}_e = -\mathrm{i} \frac{e}{\omega m_e} \delta \mathbf{E}, \tag{2.18}$$

$$\delta \mathbf{E} = \mathrm{i} \frac{\omega\, e\, n_{e0}}{\varepsilon_0 \left(\omega^2 - c^2 k^2\right)} \delta \mathbf{u}_e. \tag{2.19}$$

Note that the equation for density perturbation 2.17 may be ignored. Eliminating $\delta \mathbf{u}_e$ from the equation 2.19 one gets the equation for perturbation of the electric field,

$$\left(\omega^2 - \omega_{pe}^2 - c^2 k^2\right) \delta \mathbf{E} = 0. \tag{2.20}$$

The dispersion relation can be immediately obtained from the equation 2.20. To describe how the electromagnetic waves propagates through given medium, it is useful to introduce the index of refraction $N(\omega) = ck(\omega)/\omega$. Consequently, the dispersion relation may be rewritten as

$$N^2 = 1 - \left(\frac{\omega_{pe}}{\omega}\right)^2. \tag{2.21}$$

The important properties of the waves are distinguished by their cut-offs ($N \to 0$) and resonances ($N \to \infty$). In the vicinity of the resonance there is a total absorption, at a cut-off frequency there is a total reflection of incident waves.

In the case of the electromagnetic wave propagating through unmagnetized plasma, it might be clearly seen that there are no resonances. On the other hand, the equation 2.21 exhibits cut-off and the corresponding frequency (including ions) is given by the following expression,

$$\omega = \sqrt{\omega_{pe}^2 + \omega_{pi}^2} \tag{2.22}$$

The condition 2.22 occurs at the so-called critical plasma density $n_c\left(\omega\right)$. Note that the electromagnetic wave with frequency ω passing through plasma with densities larger than $n_c\left(\omega\right)$ is exponentially damped.

2.3.2 Magnetized plasmas

Next, consider the electromagnetic wave propagating through magnetized plasmas. To derive the dispersion relation, one may use similar treatment as in the previous subsection. The only difference is, that it is now necessary to include the effect of magnetic field. Thus, one shall solve the following set of equations,

$$\frac{\partial n_e}{\partial t} + \nabla \cdot \left(n_e \mathbf{u}_e\right) = 0, \tag{2.23}$$

$$m_e n_e \left[\frac{\partial \mathbf{u}_e}{\partial t} + \left(\mathbf{u}_e \cdot \nabla\right) \mathbf{u}_e\right] = -e n_e \left(\mathbf{E} + \mathbf{u}_e \times \mathbf{B}\,\right). \tag{2.24}$$

To obtain the wave equations for the oscillating electric and magnetic field, Faraday's law (1.3) and Ampere's law (1.4) are needed. The system of equations above will be again linearized using a small perturbations from the stationary state,

$$n_e = n_{e0} + \delta n_e, \quad \mathbf{u}_e = \mathbf{u}_{e0} + \delta \mathbf{u}_e, \quad \mathbf{B} = \mathbf{B}_0 + \delta \mathbf{B}, \quad \mathbf{E} = \mathbf{E}_0 + \delta \mathbf{E}, \tag{2.25}$$

As in the previous case, $\mathbf{u}_{e0}$ and $\mathbf{E}_0$ are identically equal to zero vector. After substituting perturbed quantities (2.26) into initial system of equations and performing Fourier transform one obtains

$$\delta n_e = \mathrm{i}\frac{n_{e0}}{\omega}\mathbf{k} \cdot \delta \mathbf{u}_e. \tag{2.26}$$

$$\delta \mathbf{u}_e = -\mathrm{i}\frac{e}{m_e \omega}\delta \mathbf{E} - \mathrm{i}\frac{e}{m_e \omega}\delta \mathbf{u}_e \times \mathbf{B}_0, \tag{2.27}$$

$$\delta \mathbf{B} = \frac{1}{\omega}\mathbf{k} \times \delta \mathbf{E}, \tag{2.28}$$

$$\delta \mathbf{E} = -\frac{1}{\varepsilon_0 \mu_0 \omega}\mathbf{k} \times \delta \mathbf{B} + \mathrm{i}\frac{e n_0}{\varepsilon_0 \omega}\delta \mathbf{u}_e. \tag{2.29}$$

Since the perturbation of density is present only in one equation, one may ignore the equation 2.26. Eliminating $\delta\mathbf{B}$ and $\delta\mathbf{u}_e$ from the equation 2.29 one gets the equation for perturbation of the electric field,

$$\left(\omega^2-\omega_{pe}^2-c^2k^2\right)\delta\mathbf{E}+\mathrm{i}\frac{\omega_{ce}}{\omega}\left(\omega^2-c^2k^2\right)\delta\mathbf{E}\times\mathbf{e}_B+ \\ +c^2\left(\mathbf{k}\cdot\delta\mathbf{E}\right)\mathbf{k}+\mathrm{i}\frac{\omega_{ce}}{\omega}c^2\left(\mathbf{k}\cdot\delta\mathbf{E}\right)\mathbf{k}\times\hat{\mathbf{e}}_B=0. \tag{2.30}$$

Here $\hat{\mathbf{e}}_B=\mathbf{B}_0/B_0$ is a unit vector in the direction of the magnetic field.

If one choose, without the loss of generality, the coordinate system where $\mathbf{B}_0=(0,0,B_0)$ and $\mathbf{k}=(k\sin\alpha,0,k\cos\alpha)$, one obtains the equation $\mathbb{M}\cdot\delta\mathbf{E}=0$ with the matrix

$$\mathbb{M}=\begin{pmatrix}\omega^2-\omega_{pe}^2-c^2k^2\cos^2\alpha & \mathrm{i}\frac{\omega_{ce}}{\omega}\left(\omega^2-c^2k^2\right) & c^2k^2\cos\alpha\sin\alpha \\ -\mathrm{i}\frac{\omega_{ce}}{\omega}\left(\omega^2-c^2k^2\cos^2\alpha\right) & \omega^2-\omega_{pe}^2-c^2k^2 & -\mathrm{i}\frac{\omega_{ce}}{\omega}c^2k^2\cos\alpha\sin\alpha \\ c^2k^2\cos\alpha\sin\alpha & 0 & \omega^2-\omega_{pe}^2-c^2k^2\sin^2\alpha\end{pmatrix}. \tag{2.31}$$

The system of equations has non-trivial solution if and only if $\det(\mathbb{M})=0$. This condition leads to the desired dispersion relation for an arbitrary angle α,

$$\left[\left(\omega^2-\omega_{pe}^2-c^2k^2\cos^2\alpha\right)\left(\omega^2-\omega_{pe}^2-c^2k^2\alpha\right)-\left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2-c^2k^2\right)\left(\omega^2-c^2k^2\cos^2\alpha\right)\right] \\ \left(\omega^2-\omega_{pe}^2-c^2k^2\sin^2\alpha\right)-c^4k^4\cos^2\alpha\sin^2\alpha\left[\left(\omega^2-\omega_{pe}^2-c^2k^2\right)-\left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2-c^2k^2\right)\right]=0. \tag{2.32}$$

Now, it is necessary to find dispersion relations for the two simplest cases, propagation along and perpendicular to the magnetic field. For the waves propagating along $\mathbf{B_0}$ is $\alpha=0$ and the dispersion relation 2.32 gets relatively simple form,

$$\left(\omega^2-\omega_{pe}^2\right)\left[\left(\omega^2-\omega_{pe}^2-c^2k^2\right)^2-\left(\frac{\omega_{ce}}{\omega}\right)^2\left(\omega^2-c^2k^2\right)^2\right]=0. \tag{2.33}$$

The equation 2.33 has three solutions. The first describes plasma oscillations at frequency $\omega=\omega_{pe}$. The second and third solutions give right-handed (R) and left-handed (L) circularly polarized waves,

$$N_R^2=1-\frac{\left(\omega_{pe}/\omega\right)^2}{1-\omega_{ce}/\omega},\qquad N_L^2=1-\frac{\left(\omega_{pe}/\omega\right)^2}{1+\omega_{ce}/\omega}. \tag{2.34}$$

In a similar manner, for the waves propagating perpendicular to $\mathbf{B_0}$ is $\alpha=\pi/2$ and the

Wave	Cut-offs	Resonances
R	$\omega_R = \frac{1}{2}\omega_{ce} + \frac{1}{2}\sqrt{\omega_{ce}^2 + 4\omega_{pe}^2}$	$\omega_{ce} = \frac{eB_0}{m_e}$
L	$\omega_L = -\frac{1}{2}\omega_{ce} + \frac{1}{2}\sqrt{\omega_{ce}^2 + 4\omega_{pe}^2}$	$\omega_{ci} = \frac{ZeB_0}{m_i}$
O	$\omega = \sqrt{\omega_{pe}^2 + \omega_{pi}^2}$	-
X	$\omega = \omega_R, \quad \omega = \omega_L$	$\omega_{lh} = \sqrt{\omega_{ce}\,\omega_{ci}}, \quad \omega_{uh} = \sqrt{\omega_{pe}^2 + \omega_{ce}^2}$

Table 1: Summary of cut-offs and resonances for all the principal waves. Note that ω_{lh} and ω_{uh} are so-called lower hybrid frequency and upper hybrid frequency, respectively.

dispersion relation 2.32 has the following form,

$$\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right)\left[\left(\omega^2 - \omega_{pe}^2\right)\left(\omega^2 - \omega_{pe}^2 - c^2k^2\right) - \omega_{ce}^2\left(\omega^2 - c^2k^2\right)\right] = 0. \tag{2.35}$$

The equation 2.35 has two solutions, which give ordinary (O) and extraordinary (X) waves,

$$N_O^2 = 1 - \left(\frac{\omega_{pe}}{\omega}\right)^2, \qquad N_X^2 = 1 - \left(\frac{\omega_{pe}}{\omega}\right)^2 \frac{1 - \left(\omega_{pe}/\omega\right)^2}{1 - \left(\omega_{pe}/\omega\right)^2 - \left(\omega_{ce}/\omega\right)^2}. \tag{2.36}$$

The ordinary wave corresponds to a linearly polarized wave with electric field lying along the magnetic field direction, so that the motion remains unaffected. The extraordinary wave has the electric fields that are perpendicular to magnetic field, but with components both perpendicular and parallel to the wave vector. Note that all of the cut-offs and resonances of waves propagating through magnetized plasmas (including ions) are listed in the table 1.

2.4 Ponderomotive force

The ponderomotive force is probably the most important quantity that describes the interaction of high intensity laser pulses with plasma. It represents the gradient of the laser electric field that pushes charged particles in plasma into the regions of a lower field amplitude and it consequently leads to a wide range of non-linear phenomena that make the plasma inherently unstable. The ponderomotive force is involved e. g. in filamentation and self-focusing of laser beam, in the formation of cavities and solitons in the plasma profile or various parametric instabilities.

Note that since the mass of the ions is much higher than the electron mass, the ponderomotive force acting on ions is in most cases negligible. However, the ponderomotive force exerted on the electrons may be consequently transmitted to the ions by the electric field which is created due to the charge separation in plasma.

In the following two subsections, one can find the derivation of the ponderomotive force for the non-relativistic as well as the relativistic case. The derivation of the non-relativistic formula for ponderomotive force is easy to understand and clearly depicts its main characteristics that are valid even for the relativistic case. However, one has to be aware that the non-relativistic approximation breaks down if the laser intensity exceeds the values around 10^{18} W/cm^2. This may be the case of short or tightly focused laser beams. For this reason, the relativistic case is briefly discussed as well in the second subsection.

2.4.1 Non-relativistic case

The ponderomotive force can be derived from the motion of single particle in prescribed electromagnetic field. Thus one may exploit the Newton's equation of motion with the Lorentz force (2.8). As the laser propagates through plasma, one may expect the oscillation of charged particles in a high-frequency electromagnetic field. For the mathematical description, one may exploit the so-called oscillation-center approximation in this case. It consist of splitting the radius vector $\mathbf{r}$ and velocity $\mathbf{v}$ of an arbitrary charged particle into two parts,

$$\mathbf{r} = \mathbf{r}_0 + \delta\mathbf{r}, \quad \mathbf{v} = \mathbf{v}_0 + \delta\mathbf{v}, \tag{2.37}$$

where the index 0 denotes the quantities at the center of oscillation and $\delta\mathbf{x} = \mathbf{x} - \mathbf{x}_0$ stands for the oscillating vector.

Next, one may exploit the Taylor series to expand both, the electric field $\mathbf{E}(\mathbf{r}, t)$ and magnetic field $\mathbf{B}(\mathbf{r}, t)$ around the oscillation center $\mathbf{r}_0$,

$$\mathbf{E}(\mathbf{r}_0 + \delta\mathbf{r}, t) \cong \mathbf{E}(\mathbf{r}_0, t) + (\delta\mathbf{r} \cdot \nabla)\mathbf{E}(\mathbf{r}_0, t), \tag{2.38}$$

$$\mathbf{B}(\mathbf{r}_0 + \delta\mathbf{r}, t) \cong \mathbf{B}(\mathbf{r}_0, t) + (\delta\mathbf{r} \cdot \nabla)\mathbf{B}(\mathbf{r}_0, t). \tag{2.39}$$

Note that taking only the zeroth and first-order terms of the fields is sufficiently accurate approximation for the description of non-relativistic ponderomotive force.

By substituting 2.37, 2.38 and 2.39 into 2.8 and collecting the terms of the same order, one obtains following two equations,

$$m_s \frac{\mathrm{d}^2\delta\mathbf{r}}{\mathrm{d}t^2} = q_s\left[\mathbf{E}(\mathbf{r}_0, t) + \mathbf{v}_0 \times \mathbf{B}(\mathbf{r}_0, t)\right], \tag{2.40}$$

$$m_s \frac{\mathrm{d}^2\mathbf{r}_0}{\mathrm{d}t^2} = q_s\left[(\delta\mathbf{r} \cdot \nabla)\mathbf{E}(\mathbf{r}_0, t) + \delta\mathbf{v} \times \mathbf{B}(\mathbf{r}_0, t)\right]. \tag{2.41}$$

To solve equations 2.40 and 2.41, assume further the laser fields in the form of standing monochromatic wave,

$$\mathbf{E}(\mathbf{r}_0, t) = \mathbf{E}_0(\mathbf{r}_0)\cos\omega t, \quad \mathbf{B}(\mathbf{r}_0, t) = -\frac{1}{\omega}\nabla \times \mathbf{E}_0(\mathbf{r}_0)\sin\omega t. \tag{2.42}$$

Note that the magnetic field in 2.42 has been obtained by the integration of Faraday's law (1.3). For the non-relativistic case, the term $\mathbf{v}_0 \times \mathbf{B}(\mathbf{r}_0, t)$ is small in comparison with the term $\mathbf{E}(\mathbf{r}_0, t)$ in the equation 2.40, thus it can be neglected. Taking into account the previous considerations, the solution of 2.40 yields the oscillating quantities,

$$\delta\mathbf{r} = -\frac{q_s}{m_s\omega^2}\mathbf{E}_0(\mathbf{r}_0)\cos\omega t, \quad \delta\mathbf{v} = \frac{q_s}{m_s\omega}\mathbf{E}_0(\mathbf{r}_0)\sin\omega t. \tag{2.43}$$

Since the integration constants can be considered as a slowly varying time functions, they have been included in $\mathbf{r}_0$.

By inserting 2.43 into 2.41 one gets immediately the following formula,

$$m_s\frac{\mathrm{d}^2\mathbf{r}_0}{\mathrm{d}t^2} = -\frac{q_s^2}{m_s\omega^2}\left[(\mathbf{E}_0(\mathbf{r}_0)\cdot\nabla)\mathbf{E}_0(\mathbf{r}_0)\cos^2\omega t + \mathbf{E}_0(\mathbf{r}_0)\times(\nabla\times\mathbf{E}_0(\mathbf{r}_0))\sin^2\omega t\right]. \tag{2.44}$$

To get the quasi-stationary ponderomotive force, one shall compute the time average values over the laser period $T = 2\pi/\omega$. Thus by performing the time average of 2.44 (note that $\langle\sin^2\omega t\rangle_T = \langle\cos^2\omega t\rangle_T = 1/2$) one gets

$$m_s\left\langle\frac{\mathrm{d}^2\mathbf{r}_0}{\mathrm{d}t^2}\right\rangle_T = -\frac{q_s^2}{2m_s\omega^2}\left[(\mathbf{E}_0(\mathbf{r}_0)\cdot\nabla)\mathbf{E}_0(\mathbf{r}_0) + \mathbf{E}_0(\mathbf{r}_0)\times(\nabla\times\mathbf{E}_0(\mathbf{r}_0))\right]. \tag{2.45}$$

The first term on the right-hand side of 2.45 contributes to the transverse component of the ponderomotive force, whilst its longitudinal component originates from the second term. Utilizing standard identities of vector calculus, one finally gets the resulting quasi-stationary ponderomotive force $\mathbf{F}_\mathrm{p}(\mathbf{r})$ in the non-relativistic case,

$$\mathbf{F}_\mathrm{p} = -\nabla\Phi_\mathrm{p}, \quad \Phi_\mathrm{p} = \frac{q_s^2}{4m_s\omega^2}E_0^2, \tag{2.46}$$

where $\Phi_\mathrm{p}(\mathbf{r})$ is the corresponding ponderomotive potential, which may be interpreted as a cycle-averaged oscillation energy.

2.4.2 Relativistic case

The non-relativistic case is based on neglecting the term $\mathbf{v}\times\mathbf{B}$ in the Newton's equation of motion (2.8). However, this approximation is not valid for particles moving with velocities close to the velocity of light c. The derivation of the relativistic ponderomotive force presented in this subsection follows [source].

A system containing a charged particle in an arbitrary electromagnetic field can be rigorously described using Lagrangian mechanics. The relativistic Lagrangian $L(\mathbf{r}, \mathbf{v}, t)$ for this case is given by the following expression,

$$L = -\frac{m_s c^2}{\gamma} - q_s (\Phi - \mathbf{v} \cdot \mathbf{A}), \quad \gamma = \left(1 - \frac{v^2}{c}\right)^{-\frac{1}{2}}. \tag{2.47}$$

For the traveling monochromatic wave, the Lagrangian $L(\mathbf{r}, \mathbf{v}, t)$ in 2.47 can be transformed to $L(\varphi)$ using the wave phase $\varphi = \mathbf{k} \cdot \mathbf{r} - \omega t$. The reason of doing this is that the transformed Lagrangian can be used to obtain the average over the wave period of motion,

$$\mathcal{L}_0(\varphi) = \frac{1}{2\pi} \int\limits_{\varphi}^{\varphi+2\pi} \mathcal{L}(\varphi') \, \mathrm{d}\varphi', \quad \mathcal{L}(\varphi) = L(\varphi) \left(\frac{\mathrm{d}\varphi}{\mathrm{d}t}\right)^{-1}. \tag{2.48}$$

Note that the Langrangian 2.48 depends only on the quantities at the center of oscillation $\mathbf{r}_0$ and $\mathbf{v}_0$ defined through φ. Then the motion of the oscillation center is governed by the Lagrange equations,

$$\frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial L_0}{\partial \mathbf{v}_0}\right) - \frac{\partial L_0}{\partial \mathbf{r}_0} = 0, \quad L_0(\varphi) = \mathcal{L}_0(\varphi) \left(\frac{\mathrm{d}\varphi}{\mathrm{d}t}\right). \tag{2.49}$$

The complete proof of assertions above can be found in [source].

The relativistic ponderomotive force in the oscillation center system can be easily found from the Lagrange equations 2.49,

$$\mathbf{F}_{\mathrm{p}} \equiv \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial L_0}{\partial \mathbf{v}_0}\right) = -c^2 \nabla m_{\mathrm{eff}}, \tag{2.50}$$

where the so-called effective mass m_{eff} is the space and time dependent quantity that has been introduced as follows,

$$m_{\mathrm{eff}} = -\mathcal{L}_0 \frac{\gamma_0}{c^2} \left(\frac{\mathrm{d}\varphi}{\mathrm{d}t}\right), \quad \gamma_0 = \left(1 - \frac{v_0^2}{c}\right)^{-\frac{1}{2}}. \tag{2.51}$$

Note that the assertion 2.51 holds for any electromagnetic field in vacuum in which it is possible to define the oscillation center. In an ordinary conditions, the effective mass m_{eff} can be rewritten as

$$m_{\mathrm{eff}} = m_s \sqrt{1 + \frac{q_s A^2}{\alpha m_s^2 c^2}} \tag{2.52}$$

with $\alpha = 1$ for circular and $\alpha = 2$ for linear polarization of the electromagnetic wave.

2.5 Self-induced transparency (SIT)

As already mentioned in the section 2.3, the electromagnetic wave with frequency ω cannot propagate through plasma with densities greater than the critical density $n_c(\omega)$. However, in the case of intense laser beam, one has to perform a relativistic generalization of the dispersion relation, thus the corresponding cut-off frequency is given by the following formula,

$$\omega = \sqrt{\frac{\omega_{pe}^2 + \omega_{pi}^2}{\gamma}}, \quad \gamma = \left(1 - \frac{v^2}{c^2}\right)^{-\frac{1}{2}}. \tag{2.53}$$

Therefore, one may immediately see that in the relativistic case, the laser beam actually propagates up to densities $n_c'(\omega) = n_c(\omega)\gamma > n_c(\omega)$. This increase of the effective critical density is called relativistic self-induced transparency (SIT) [source].

Note that in the case of real laser pulse, the ability to penetrate into the overdense plasma depends on the local amplitude. Therefore, there are logically some parts of laser beam that may propagate, while the others cannot. Consequently, both the reflected and transmitted laser intensity profiles reveal temporal chopping that leads to the significant modifications of the pulse shape.

Furthermore, one has to be aware that the plasma does not remain homogeneous during the laser-plasma interaction. As mentioned in the section 2.4, the ponderomotive force pushes charged particles in plasma into the regions of a lower field amplitude creating a charge separation layer. Consequently, the plasma density rapidly increases in these regions. Therefore, it is necessary to resolve the propagation of the electromagnetic wave self-consistently with the modification of plasma density profile.

Note that the electromagnetic wave propagating in a relativistically transparent target may be a subject of strong instabilities that lead to an additional plasma heating. Thus this way, the energy of electromagnetic wave may be also absorbed. Furthermore, in the SIT regime, the acceleration of charged particles due to the ponderomotive force may take place in the whole volume of the target.

At the end of this section, an approach to SIT based on a steady state solutions that follows [source] is presented. Consider a laser pulse to be a monochromatic plane wave at normal incidence and with circular polarization. Then the thresholds for SIT in the case of two limit cases, a semi-infinite plasma and an ultrathin plasma slab of thickness l, are given by following two formulas,

$$a_0 \cong \frac{3^{3/2}}{2^3}\left(\frac{n_0}{n_c}\right)^2 \quad \text{for} \quad n = n_0 H(x), \tag{2.54}$$

$$a_0 \cong \frac{\pi l}{\lambda}\frac{n_0}{n_c} \quad \text{for} \quad n = n_0 l \delta(x), \tag{2.55}$$

where $H(x)$ stands for a Heaviside step function and $\delta(x)$ is Dirac delta function. The threshold values are expressed in terms of laser dimensionless potential $a_0 = eE_0/m_e\omega c$. If the conditions 2.54 and 2.55 are fulfilled, laser beam penetrates plasma as a non-linear wave.

Note that the model presented above is based on several simplifications (e.g. the pulse profile or electron oscillations and heating are not included). In general case, the SIT dynamics is obviously much more complicated.

2.6 Laser absorption and electron heating mechanisms

Absorption of laser energy during the laser-plasma interaction is an important issue to be investigated. The energy of laser pulses can be absorbed in plasma by various different non-linear mechanisms depending on the laser intensity. For the laser intensities considered within this work (above 10^{16} W/cm^2), the plasma is predominantly heated by collisionless absorption mechanisms. These mechanisms are generally characterized by the fact that the energy is transfered to a relatively small part of so-called hot electrons. In the following three subsections, the main mechanisms of collisionless plasma heating are briefly described.

2.6.1 Resonance absorption

The energy of laser beam may be absorbed by resonance absorption in the plasma. It is a linear process in which an incident laser wave is partially absorbed by conversion into an electron wave at the critical density of plasma $n_c(\omega)$.

Resonance absorption takes place when a linearly p-polarized laser pulse is obliquely incident on a plasma with an inhomogeneous density profile. A component of the electric laser field perpendicular to the target surface then resonantly excites an electron plasma wave also along the plasma density gradient, thus a part of the laser wave energy is transferred into the electrostatic energy of the electron plasma wave. This wave propagates into the underdense plasma and is damped either by collision or collisionless absorption mechanisms. Consequently, energy is further converted into the thermal energy which heats the plasma and may possibly produce hot electrons.

Particularly, resonance absorption is the main absorption process for the laser beams of higher intensities and longer wavelengths. The efficiency of the resonance absorption may also be higher in plasma with high temperature, low critical density, or short scale length of the density profile.

2.6.2 Brunel's vaccum heating

Similarly as for the resonance absorption, Brunel's vaccum heating takes place when a linearly p-polarized laser pulse is obliquely incident on a plasma. In this case, however, the incident angle must be relatively large. Moreover, the plasma density profile has to be steep, so the amplitude of the oscillating electrons driven by the electric laser field is larger than the density scale length. Consequently, the energy of laser pulse carried by electrons is transfered into mechanical or heat energy in the overdense plasma region.

The energy absorbed via Brunel's vacuum heating is carried by hot electrons in the bunches ejected once per laser period. To give a physical picture, in the first half of laser cycle, the electrons are pushed inside the plasma gaining only a small amount of energies because the electric laser field is shielded. On the other hand, in the second half of laser cycle, the electrons gain very high energies while they are ejected into vacuum.

The trajectory of charged particles is strongly influenced by the time of their expulsion. Furthermore, a self-consistent electric field may be generated if a large amount of charged particles is ejected at the same time. The majority of the charged particles, however, returns into the plasma without restoring forces behind the skin layer due to the oscillating component of the laser field and self-consistent electric field. Note that since the charged particles are accelerated by different phases of the laser field, the distribution of such particles can be in most cases considered as a Maxwellian [source].

2.6.3 Relativistic $\mathbf{J} \times \mathbf{B}$ heating

For the relativistic laser beams, the high-frequency $\mathbf{v} \times \mathbf{B}$ component of the Lorentz force becomes important and may contribute to plasma heating. Similarly to the Brunel's vacuum heating, discussed in the previous subsection, $\mathbf{J} \times \mathbf{B}$ heating requires very steep plasma density gradients. On the contrary, this heating scenario works for any polarization apart from circular and is most efficient for the laser pulses normally incident onto the target surface, so there is no oscillating component of the electric field perpendicular to plasma.

In the case of $\mathbf{J} \times \mathbf{B}$ heating, the absorbed energy is carried by bunches of hot electrons that are ejected twice per laser period. This fact may help to distinguish between the effects of Brunel's and $\mathbf{J} \times \mathbf{B}$ heating. Again, the ejected electrons are pushed back into the plasma by self-consistent electric field generated by charged particles without restoring forces behind the skin layer.

2.7 Mechanisms of laser-driven ion acceleration

Because of the relatively large ion mass, currently achievable laser intensities are not strong enough to accelerate protons or heavier ions directly to sufficiently high energies. However, ions typically respond on slowly varying electric fields in plasma arising from the strong charge separations induced by various phenomena that take place during the interaction of intense laser beam with matter.

As one may see later, ions can be either accelerated in the vicinity of the laser focal spot at the front side of the target as well as in the vicinity of the target-vacuum boundary at the rear side. In the last two subsections of this chapter, the two main mechanisms for ion acceleration in laser-plasma interactions are briefly described.

2.7.1 Radiation pressure acceleration (RPA)

Radiation pressure acceleration (RPA) stands for the mechanism in which the ions are accelerated from the target front side in the vicinity of the laser focal spot. The acceleration is driven by the ponderomotive force (see the section 2.4) which expels the electrons into the regions of a lower laser intensities and consequently generates a strong electrostatic fields as a result of a charge separation. In the case of intense laser beams, the radiation pressure is strong enough to push an overdense target inwards whilst changing the shape of its surface and correspondingly the density profile. This process is commonly named hole boring.

For a plane, monochromatic wave at a normal incidence onto a target at rest, the balance between the electrostatic pressure and the radiation pressure at the target surface can be expressed as follows,

$$\frac{1}{2}\varepsilon_0 E_{\mathrm{es}}^2 = \frac{(1+R-T)}{c} I, \tag{2.56}$$

where $\mathbf{E}_{\mathrm{es}}(\mathbf{r}, t)$ is the electrostatic field generated by the charge separation, R and T are the reflection and transmission coefficients of the target, respectively, and I is the intensity of incident laser pulse.

The formula 2.56 determines the extension of a charge depletion layer, which is established at the front side of the target. Ions in the depletion layer are accelerated by the electrostatic field $\mathbf{E}_{\mathrm{es}}(\mathbf{r}, t)$, which amplitude can be obtained by solving the Poisson equation. The maximum energy, that the ions may gain, is then estimated as follows [source],

$$\mathcal{E}_{max} = \frac{Z m_e c^2 a_0^2}{m_i \gamma}, \quad \gamma = \sqrt{1 + \frac{a_0^2}{2}}. \tag{2.57}$$

The RPA regime starts to dominate for a laser beams with peak intensity higher than 10^{21} W/cm^2. Also, RPA could be more efficient for circularly polarized laser beams normally incident onto a target, because of their ability to suppress the effects of electron heating mechanisms mentioned in the previous section. Note that the RPA mechanism typically produce an ion beam of a large divergence because the critical density interface where the charge separation occurs is curved by the shape of the laser beam.

2.7.2 Target normal sheath acceleration (TNSA)

The interaction of intense laser beam with matter typically produce a population of hot electrons that may pass through the target and propagate beyond its rear side. This leads to a generation of a strong electrostatic potential which accelerates the ionized particles in the direction normal to the target rear surface. Therefore, this mechanism is commonly known as a target normal sheath acceleration (TNSA). TNSA can be also observed on the target front, because the majority of expelled electrons is often attracted back to the target and may even reach the front side.

The approximation for TNSA mechanism can be provided either by the static or dynamic model of the sheath. In the following, a dynamic model based on a free isothermal expansion of electron-ion plasma into vacuum is briefly described.

To find a non-linear solution that corresponds to the isothermal rarefaction wave, one may exploit the hydrodynamic equations for an unique ion component,

$$\frac{\partial n_i}{\partial t} + \nabla \cdot (n_i \mathbf{u}_i) = 0, \tag{2.58}$$

$$\frac{\partial \mathbf{u}_i}{\partial t} + (\mathbf{u}_i \cdot \nabla)\, \mathbf{u}_i = \frac{Ze}{m_i}\mathbf{E}. \tag{2.59}$$

Assume that at time $t = 0$, the electron density satisfies the Boltzmann distribution and the ion density occupies a half space with an infinitely sharp boundary, thus

$$n_e = n_{e0} \exp\left(\frac{e\Phi}{k_B T_e}\right), \quad n_i = n_{i0} H(x). \tag{2.60}$$

Assume further that the plasma is locally neutral on the scale length larger than the Debye radius, therefore $n_e = Zn_i$. However, one shall be aware that the condition for the local neutrality does not imply that there is no electrostatic field. The electrostatic field may arise due to the sources coming from the regions where the condition for quasi-neutrality does not hold. Therefore, the electric field $\mathbf{E}(\mathbf{r}, t)$ in the equation 2.59 may be replaced by the electrostatic field $\mathbf{E}_{es}(\mathbf{r}, t) = -\nabla\Phi(\mathbf{r}, t)$, which one can obtain by taking the gradient of the electron density 2.60.

After replacing the electric field $\mathbf{E}(\mathbf{r}, t)$ in the equation 2.59 one gets

$$\frac{\partial \mathbf{u}_i}{\partial t} + (\mathbf{u}_i \cdot \nabla)\, \mathbf{u}_i = -\frac{c_s^2}{n_i}\nabla n_i, \quad c_s = \sqrt{\frac{Zk_B T_e}{m_i}}, \tag{2.61}$$

where $c_s(\mathbf{r}, t)$ is the ion-acoustic velocity. By performing the Fourier transforms of the equations 2.58 and 2.61, one easily obtain the dispersion relation of the ion-acoustic waves. However, the system has also a non-linear self-similar solution which describes the rarefaction wave. Thus, define a self-similar variable in 1D geometry as $\xi = x/t$. The ion fluid density and velocity then become $n_i(x, t) = N(\xi)$ and $u_i(x, t) = V(\xi)$, respectively. After transformation, the system of equations 2.58 and 2.61 yields

$$\frac{1}{N}\frac{\mathrm{d}N}{\mathrm{d}\xi} = -\frac{V - \xi}{c_s}, \tag{2.62}$$

$$\frac{\mathrm{d}V}{\mathrm{d}\xi} = -\frac{1}{N}\frac{\mathrm{d}N}{\mathrm{d}\xi}(V - \xi). \tag{2.63}$$

Now, the set of equations 2.62, 2.63 can be solved easily. The solution in the original coordinates can be found below,

$$n_i = n_{i0} \exp\left(-\frac{x}{c_s t} - 1\right), \quad u_i = c_s + \frac{x}{t}. \tag{2.64}$$

Note that the solution 2.64 is valid only for $x > -c_s t$ where $u_i > 0$. The condition $x = -c_s t$ describes the rarefaction front propagating backwards into the target at the speed c_s. Finally, by plugging the solution 2.64 into the equation 2.59, one may find the expression for the electric field at the ion front $x = -c_s t$,

$$E_x = \frac{2E_0}{\omega_{pi} t}, \quad E_0 = c_s \sqrt{\frac{n_i m_i}{\varepsilon_0}}. \tag{2.65}$$

The apparent drawback of formula 2.65 is that it has a singularity at time $t = 0$. However, one may find a simple expression for the electric field at $t = 0$ by integration of the Poisson's equation from $x = 0$ to $x = +\infty$,

$$E_x = \sqrt{\frac{2}{e}} E_0. \tag{2.66}$$

A very precise interpolation of the electric field at the ion front, which is valid for any time, has been already found [source],

$$E_x \cong \frac{2E_0}{\sqrt{2e + \omega_{pi}^2 t^2}}. \tag{2.67}$$

As a consequence of formula 2.67, one may find relatively accurate prediction of the corresponding ion front velocity by solving the equation of motion 2.59 with the electric field 2.67,

$$u_i \cong 2c_s \ln\left[\frac{\omega_{pi} t}{\sqrt{2e}} + \left(\frac{\omega_{pi}^2 t^2}{2e} + 1\right)^{\frac{1}{2}}\right]. \tag{2.68}$$

The next drawback of the model is that the derived formula for maximum velocity of ions 2.68 diverges logarithmically with time, therefore the system accelerates the ions infinitely. This result leads from the isothermal assumption. Nevertheless, one might overcome this drawback by assuming the finite thickness of plasma or by introducing a maximum acceleration time constant at which the ion acceleration stops [source].

Chapter 3

Particle-in-cell (PIC)

This chapter is devoted to numerical simulations, particularly to the particle-in-cell (PIC) method, which represents one of the most popular numerical algorithms in plasma physics. There can be found the mathematical background of this method, description of the individual steps of the computational cycle as well as the stability conditions. The last section provides a brief overview of the particle-in-cell code EPOCH, which has been used for simulations within this work.

3.1 Mathematical derivation

Laser-plasma interaction involves a collective behavior of particles and electromagnetic fields which is in general case complex and strongly non-linear problem to solve. The investigation of such systems cannot be usually carried out only through the theoretical and experimental work. Since a large amount of simultaneous interactions with many degrees of freedom may be present there, analytical modeling seems to be impractical. On the other hand, many of the significant details of laser-plasma interaction may be extremely difficult or even impossible to obtain experimentally. Hence, for the further progress in this field of research, other tools and techniques are required [10].

Numerical simulations help researchers to develop models covering a wide range of physical scenarios as well as to investigate their properties. With the advent of powerful computational systems, numerical simulations now play an important role as an essential tool in developing theoretical models and understanding experimental results in various fields of modern science [12]. These so-called computer experiments are often faster and much cheaper than a single real experiment in laboratory [8]. In addition, one numerical code can solve a broad range of tasks only by modification of its initial and boundary conditions.

Nowadays, it is clear that detailed understanding of the physical mechanisms in the laser-plasma interaction can be achieved only through the combination of theory, experiment and simulation. Development of parallel algorithms that are able to provide sufficiently exact solutions in reasonable time, however, belongs to the most challenging fields of modern science.

The PIC method refers to a technique that has been used to solve a certain class of partial differential equations. The method has been proposed in the mid-fifties and it has early gained a great popularity in the field of plasma physics. It is based on the kinetic description approach, thus the evolution of the system is conducted in principle via the motion of the charged particles.

However, real systems are often extremely large in terms of the number of particles they contain. In order to make simulations computationally tractable, so-called macro-particles are used. The macro-particle is a finite-size computational particle representing a group of physical particles that are near each other in the phase space. Notice that it is allowed to rescale the number of particles, because the Lorentz force depends only on the charge to mass ratio, which is invariant to this transformation. Thus, the macro-particle will follow the same trajectory as the corresponding real particles would [9].

Although this approach significantly reduces the number of computational particles in simulation, the binary interactions for every pair of a system cannot be taken into account. The cost would scales quadratically as the number of particles increases and that makes the computational effort unmanageable in the case of larger systems. Fortunately, many of the phenomena occur in high-temperature plasmas where the collective behavior dominates and the collisional effects are very weak (see 2.6), thus one can neglect them. Otherwise, one has to exploit other techniques to estimate the collisional effects [11].

In the collisionless plasma, the phase space particle distribution function $f_s\left(\mathbf{r},\mathbf{v},t\right)$ for a given species s is governed by the Vlasov equation (2.9). The mathematical formulation of the PIC method is obtained by assuming that the distribution function $f_s\left(\mathbf{r},\mathbf{v},t\right)$ is given by the sum of distribution functions for macro-particles $f_{p,s}\left(\mathbf{r},\mathbf{v},t\right)$,

$$f_s = \sum_p f_{p,s}. \tag{3.1}$$

Index p denotes hereafter the quantities attributable to macro-particles. The distribution function for each macro-particle is further assumed to be

$$f_{p,s}\left(\mathbf{r},\mathbf{v},t\right) = w_p S_r\left(\mathbf{r}-\mathbf{r}_p\left(t\right)\right) S_v\left(\mathbf{v}-\mathbf{v}_p\left(t\right)\right), \tag{3.2}$$

where w_p is a weight depending on the number of physical particles represented by each macro-particle, and $S_r\left(\mathbf{r}-\mathbf{r}_p\left(t\right)\right)$, $S_v\left(\mathbf{v}-\mathbf{v}_p\left(t\right)\right)$ are the so-called shape functions for the spatial and velocity coordinate, respectively.

The shape functions cannot be chosen arbitrarily. They have to fulfill several special properties. Let S_ξ be the shape function of the phase space coordinate ξ. Then:

1. The support of the shape function is compact, $\exists R > 0$, $\operatorname{supp} S_\xi \subset (-R, R)$.

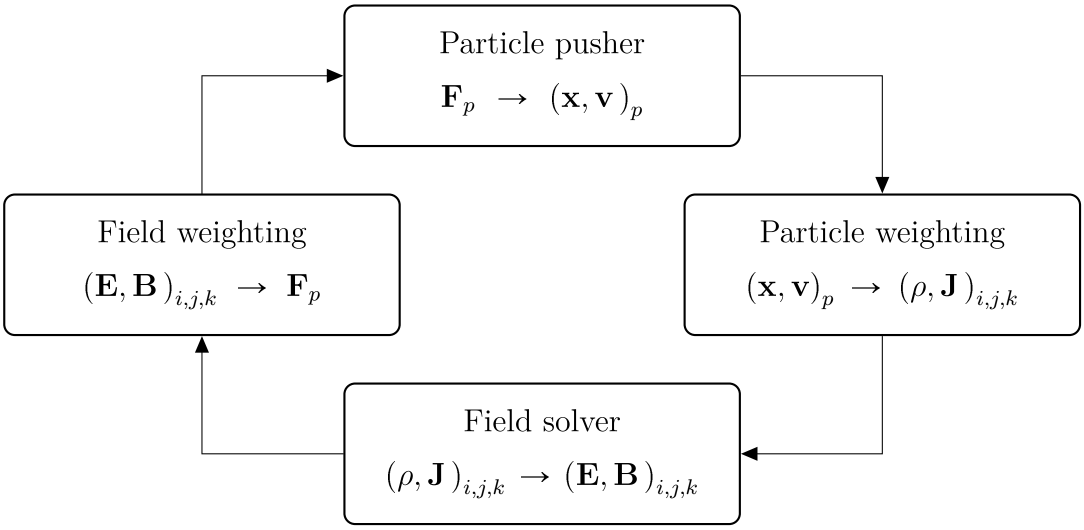

Figure 1: Computational cycle of the particle-in-cell method

2. Integral of the shape function is unitary, $\int\limits_{-\infty}^{+\infty} S_\xi(\xi)\,\mathrm{d}\xi = 1$.
3. The shape function is symmetrical, $S_\xi(\xi) = S_\xi(-\xi)$.

While these restrictive conditions still offer a wide range of options, the standard PIC method is essentially determined by the choice of the shape function in the velocity coordinate as a Dirac delta function and in the spatial coordinate as a m-th order b-spline basis function b_m. Note that the choice of the shape functions has strong impact on the stability and accuracy of the simulation. Higher-order basis functions results in less numerical noise and reduces non-physical phenomena in simulations, obviously at the cost of increased computational time.

Substituting the discretization 3.1 into the Vlasov equation 2.9 and taking into account the properties of shape functions mentioned above, one obtains the set of equations of motion for macro-particle p,

$$\frac{\mathrm{d}\mathbf{r}_p}{\mathrm{d}t} = \mathbf{v}_p, \qquad \frac{\mathrm{d}\mathbf{v}_p}{\mathrm{d}t} = \frac{\mathbf{F}_p}{m_s}, \tag{3.3}$$

where $\mathbf{F}_p(t)$ represents a spatial average of the force acting on the macro-particle,

$$\mathbf{F}_p = q_s\left(\mathbf{E}_p + \mathbf{v}_p \times \mathbf{B}_p\right). \tag{3.4}$$

The fields at the macro-particle position $\mathbf{E}_p(t)$ and $\mathbf{B}_p(t)$ in 3.4 are given by the spatial shape function, which implies some form of interpolation. This will be closer discussed in the fourth section of this chapter.

The PIC method combines Lagrangian and Eulerian frame of reference. It means that the macro-particles are tracked in continuous phase space, whereas the electromagnetic fields are calculated only on stationary grid points. Therefore, it is necessary to perform the discretization of the spatial coordinates $\mathbf{r} \to \mathbf{r}_{ijk}$ where $(i, j, k) \in \mathbb{Z}^3$ are grid indices. It is also necessary to perform discretization of the temporal coordinate $t \to t^n$, where $n \in \mathbb{N}$ is the time level index. The algorithm will be outlined for a three-dimensional equidistant rectangular grid, thus $\mathbf{r}_{ijk} = (i\Delta x, j\Delta y, k\Delta z)$, where $\Delta x, \Delta y, \Delta z$ are the spatial steps in each direction and $t^n = n\Delta t$, where Δt is the time step. Each quantity $A(\mathbf{r}_{ijk}, t^n)$ will be hereafter denoted as A^n_{ijk}.

The computational cycle of the PIC method is shown in Figure 1. Individual steps are closer described in several following sections. The influence of the choice of the time and spatial step on the stability and accuracy of the PIC method will be demonstrated as well.

3.2 Particle pusher

As one could already see from 3.3, the motion of macro-particles in simulation is governed by the Newton equations with spatially averaged Lorentz force. Since the particles can reach velocities near the velocity of light, it is necessary to perform relativistic generalization,

$$\mathbf{u}_p = \gamma\,\mathbf{v}_p, \qquad \gamma = \sqrt{1 + \left(\frac{\mathbf{u}_p}{c}\right)^2}. \tag{3.5}$$

Assume that the electric and magnetic fields are interpolated from the grid points to the particles at the time level t^n. Then the equations of motion to be integrated are

$$\frac{\mathrm{d}\mathbf{r}_p}{\mathrm{d}t} = \frac{\mathbf{u}_p}{\gamma}, \qquad \frac{\mathrm{d}\mathbf{u}_p}{\mathrm{d}t} = \frac{q_s}{m_s}\left(\mathbf{E}_p + \frac{\mathbf{u}_p}{\gamma} \times \mathbf{B}_p\right). \tag{3.6}$$

To discretize the equations of motion 3.6, a time-centered leap-frog scheme is used. One obtains

$$\frac{\mathbf{r}_p^{n+1} - \mathbf{r}_p^{n}}{\Delta t} = \frac{\mathbf{u}_p^{n+1/2}}{\gamma^{\,n+1/2}}, \tag{3.7}$$

$$\frac{\mathbf{u}_p^{n+1/2} - \mathbf{u}_p^{n-1/2}}{\Delta t} = \frac{q_s}{m_s}\left(\mathbf{E}_p^n + \frac{\mathbf{u}_p^{n+1/2} + \mathbf{u}_p^{n-1/2}}{2\gamma^{\,n}} \times \mathbf{B}_p^n\right). \tag{3.8}$$

Although these equations appear to be very simple, the solution is the most time-consuming part of the simulation, because they must be solved for every single macro-particle at each time step. Standard approach for particle pushing in plasma simulation PIC codes involves elegant Boris method, which completely separates the effect of electric and magnetic fields [4]. Substitute

$$\mathbf{u}_p^{n-1/2} = \mathbf{u}_p^- - \frac{q_s\mathbf{E}_p^n}{m_s}\frac{\Delta t}{2}, \qquad \mathbf{u}_p^{n+1/2} = \mathbf{u}_p^+ + \frac{q_s\mathbf{E}_p^n}{m_s}\frac{\Delta t}{2} \tag{3.9}$$

into the equation 3.8, then the electric field cancels entirely,

$$\frac{\mathbf{u}_p^+ - \mathbf{u}_p^-}{\Delta t} = \frac{q_s}{2\gamma^n m_s}\left(\mathbf{u}_p^+ + \mathbf{u}_p^-\right) \times \mathbf{B}_p^n. \tag{3.10}$$

The equation 3.10 describes a rotation of the vector $\mathbf{u}_p^-$ to $\mathbf{u}_p^+$ in one simulation time step Δt. The angle θ between the vector $\mathbf{u}_p^-$ and $\mathbf{u}_p^+$ is expected to be $\theta = \omega_c \Delta t$.

To implement the Boris method, first add half the electric impulse $\mathbf{E}_p^n$ to $\mathbf{u}_p^{n-1/2}$, then perform the full rotation by the angle θ, and finally, add another half the electric impulse $\mathbf{E}_p^n$. From the basic geometry Boris derived following steps to obtain $\mathbf{u}_p^{n+1/2}$. From the first of the equations 3.9, express the vector $\mathbf{u}_p^-$,

$$\mathbf{u}_p^- = \mathbf{u}_p^{n-1/2} + \frac{q_s \mathbf{E}_p^n}{m_s}\frac{\Delta t}{2} \tag{3.11}$$

and construct an auxiliary vector $\mathbf{u}_p'$, which is simultaneously perpendicular to $\left(\mathbf{u}_p^+ - \mathbf{u}_p^-\right)$ and to $\mathbf{B}_p^n$,

$$\mathbf{u}_p' = \mathbf{u}_p^- + \mathbf{u}_p^- \times \mathbf{t}, \qquad \mathbf{t} = \frac{q_s \mathbf{B}_p^n}{\gamma^n m_s}\frac{\Delta t}{2}. \tag{3.12}$$

The vector $\mathbf{t}$ has to be logically parallel to $\mathbf{B}_p^n$ with the length of $\tan(\theta/2) \approx \theta/2$ for small angles. Next, exploit the fact that the vector $(\mathbf{u}_p' \times \mathbf{B}_p^n)$ is parallel to $\left(\mathbf{u}_p^+ - \mathbf{u}_p^-\right)$ and express the vector $\mathbf{u}_p^+$,

$$\mathbf{u}_p^+ = \mathbf{u}_p^- + \mathbf{u}_p' \times \mathbf{s}, \qquad \mathbf{s} = \frac{2\mathbf{t}}{1+t^2}. \tag{3.13}$$

Here the vector $\mathbf{s}$ is parallel to $\mathbf{B}_p^n$ and its length has to fulfill the condition $\|\mathbf{u}_p^+\| = \|\mathbf{u}_p^-\|$. The transition from $\mathbf{u}_p^-$ to $\mathbf{u}_p^+$ can be written more clearly using the matrix,

$$\mathbf{u}_p^+ = \mathbb{M}\,\mathbf{u}_p^-, \tag{3.14}$$

where

$$\mathbb{M} = \begin{pmatrix} 1 - s_2 t_2 - s_3 t_3 & s_2 t_1 + s_3 & s_3 t_1 - s_2 \\ s_1 t_2 - s_3 & 1 - s_1 t_1 - s_3 t_3 & s_3 t_2 + s_1 \\ s_1 t_3 + s_2 & s_2 t_3 - s_1 & 1 - s_1 t_1 - s_2 t_2 \end{pmatrix}. \tag{3.15}$$

Finally, substitute the vector $\mathbf{u}_p^+$ into the second from the equations 3.9,

$$\mathbf{u}_p^{n+1/2} = \mathbf{u}_p^+ + \frac{q_s \mathbf{E}_p^n}{m_s}\frac{\Delta t}{2}. \tag{3.16}$$

The position of macro-particle particle is then advanced according to

$$\mathbf{r}_p^{n+1} = \mathbf{r}_p^{n} + \frac{\mathbf{u}_p^{n+1/2}}{\gamma^{n+1/2}}\Delta t, \qquad \gamma^{n+1/2} = \sqrt{1 + \left(\frac{\mathbf{u}_p^{n+1/2}}{c}\right)^2}. \tag{3.17}$$

3.3 Field solver

As already mentioned, the behavior of the time varying electromagnetic field in free space is governed by the microscopic variant of Maxwell equations (1.1 - 1.4). A typical numerical technique to resolve the Maxwell's equations with respect to time is finite-difference time-domain method (FDTD), because it is probably the simplest technique in terms of the implementation.

The vector components of the fields $\mathbf{E}_{ijk}^n$ and $\mathbf{B}_{ijk}^n$ are spatially staggered about rectangular cells of the computational grid,

$$\mathbf{E}_{ijk}^n \rightarrow \left[(E_x)_{i,\,j+1/2,\,k+1/2}^n , (E_y)_{i+1/2,\,j,\,k+1/2}^n , (E_z)_{i+1/2,\,j+1/2,\,k}^n \right], \tag{3.18}$$

$$\mathbf{B}_{ijk}^n \rightarrow \left[(B_x)_{i+1/2,\,j,\,k}^n , (B_y)_{i,\,j+1/2,\,k}^n , (B_z)_{i,\,j,\,k+1/2}^n \right]. \tag{3.19}$$

This scheme, which has proven to be very robust, is now known as a Yee lattice [15]. The illustration of a standard Cartesian Yee cell used for FDTD is shown in Figure 2. Components of the current density $\mathbf{J}_{ijk}^n$ are defined in the same way as the components of $\mathbf{E}_{ijk}^n$, charge density ρ_{ijk}^n is defined in the middle of the cell,

$$\mathbf{J}_{ijk}^n \rightarrow \left[(J_x)_{i,\,j+1/2,\,k+1/2}^n , (J_y)_{i+1/2,\,j,\,k+1/2}^n , (J_z)_{i+1/2,\,j+1/2,\,k}^n \right], \tag{3.20}$$

$$\rho_{ijk}^n \rightarrow \rho_{i+1/2,\,j+1/2,\,k+1/2}^n. \tag{3.21}$$

For marching in time a leap-frog scheme is used, thus discretized Maxwell's equations 1.1 - 1.4 have the following form,

$$\nabla^+ \cdot \mathbf{E}_{ijk}^n = \frac{\rho_{ijk}^n}{\varepsilon_0}, \tag{3.22}$$

$$\nabla^- \cdot \mathbf{B}_{ijk}^{n+1/2} = 0, \tag{3.23}$$

$$\frac{1}{c^2}\frac{\mathbf{E}_{ijk}^{n+1} - \mathbf{E}_{ijk}^n}{\Delta t} = \nabla^+ \times \mathbf{B}_{ijk}^{n+1/2} - \mu_0 \mathbf{J}_{ijk}^{n+1/2}, \tag{3.24}$$

$$\frac{\mathbf{B}_{ijk}^{n+1/2} - \mathbf{B}_{ijk}^{n-1/2}}{\Delta t} = -\nabla^- \times \mathbf{E}_{ijk}^n. \tag{3.25}$$

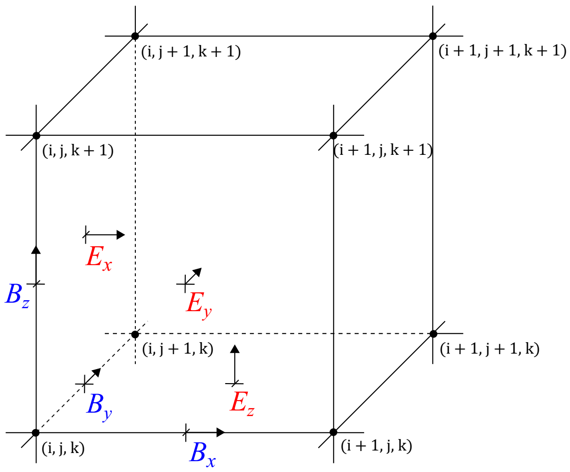

Figure 2: Standard Cartesian Yee cell used for FDTD method

Notice that this scheme achieves second-order accuracy in both, space and time. Discrete operators (∇^+) and (∇^-) used in 3.22 - 3.25 act on a scalar field f_{ijk} as follows,

$$\nabla^+ f_{ijk} = \left(\frac{f_{i+1,\,j,\,k} - f_{i,\,j,\,k}}{\Delta x}, \frac{f_{i,\,j+1,\,k} - f_{i,\,j,\,k}}{\Delta y}, \frac{f_{i,\,j,\,k+1} - f_{i,\,j,\,k}}{\Delta z} \right), \tag{3.26}$$

$$\nabla^- f_{ijk} = \left(\frac{f_{i,\,j,\,k} - f_{i-1,\,j,\,k}}{\Delta x}, \frac{f_{i,\,j,\,k} - f_{i,\,j-1,\,k}}{\Delta y}, \frac{f_{i,\,j,\,k} - f_{i,\,j,\,k-1}}{\Delta z} \right). \tag{3.27}$$

These operators have the following properties,

$$\nabla^- \cdot \nabla^- \times = \nabla^+ \cdot \nabla^+ \times = 0, \qquad \nabla^- \cdot \nabla^+ = \nabla^+ \cdot \nabla^- = \Delta^\pm. \tag{3.28}$$

Symbol $\Delta^\pm$ stands for the discrete Laplace operator in central differences,

$$\Delta^\pm f_{i,j,k} = \frac{f_{i-1,j,k} + 2f_{i,j,k} + f_{i+1,j,k}}{\Delta x^2} + \frac{f_{i,j-1,k} + 2f_{i,j,k} + f_{i,j+1,k}}{\Delta y^2} + \frac{f_{i,j,k-1} + 2f_{i,j,k} + f_{i,j,k+1}}{\Delta z^2}. \tag{3.29}$$

Before trying to find the solution of discretized Maxwell equations 3.22 - 3.25, one must realize that this system of equations is not independent. In the three-dimensional case, there are eight first-order differential equations, but only six unknown vector components. Acting on the equations 3.24 and 3.25 by operators $(\nabla^- \cdot)$ and $(\nabla^+ \cdot)$, respectively, one obtains

$$\frac{\nabla^- \cdot \mathbf{B}_{ijk}^{n+1/2} - \nabla^- \cdot \mathbf{B}_{ijk}^{n-1/2}}{\Delta t} = 0, \tag{3.30}$$

$$\frac{\rho_{ijk}^{n+1} - \rho_{ijk}^{n}}{\Delta t} + \nabla^+ \cdot \mathbf{J}_{ijk}^{n+1/2} = 0. \tag{3.31}$$

It means that it is possible to solve only the equations 3.24 and 3.25, while the divergence equations 3.22, 3.23 can be considered as the initial conditions. Note that in this case, the continuity equation in the finite differences (3.31) has to be fulfilled.

3.4 Particle and field weighting

In order to solve the Maxwell's equations, as shown in the previous section of this chapter, one has to know the source terms produced by the motion of the charged particles. In other words, it is necessary to assign charge and current densities from the continuous macro-particle positions to the discrete grid points. This simulation step is usually referred to as particle weighting and it involves some form of interpolation.

According to the kinetic theory, charge density $\rho(\mathbf{x}, t)$ and current density $\mathbf{J}(\mathbf{x}, t)$ are given by the following integrals over the velocity space,

$$\rho(\mathbf{x}, t) = \sum_s q_s \int f_s(\mathbf{x}, \mathbf{v}, t)\,\mathrm{d}\mathbf{v}, \qquad \mathbf{J}(\mathbf{x}, t) = \sum_s q_s \int f_s(\mathbf{x}, \mathbf{v}, t)\,\mathbf{v}\,\mathrm{d}\mathbf{v}. \tag{3.32}$$

After discretization of 3.32 using macro-particles and exploiting the properties of the shape functions, one gets immediately

$$\rho^n_{ijk} = \sum_p q_p S_r\left(\mathbf{r}_{ijk} - \mathbf{r}^n_p\right), \qquad \mathbf{J}^n_{ijk} = \sum_p q_p \mathbf{v}^n_p S_r\left(\mathbf{r}_{ijk} - \mathbf{r}^n_p\right), \tag{3.33}$$

where $q_p = q_s w_p$. However, using the formulas 3.33 for charge and current deposition in PIC codes may violate the discrete continuity equation (3.31) and in turn cause errors in Gauss's law (3.22). In this case, one would have to solve the Poisson's equation for the correction of the electric field at every simulation time step or use a numerical scheme that satisfies the continuity equation exactly. These schemes are referred to as a charge conservation methods ([14], [5], [source]).

Similarly, to advance macro-particle positions, as shown in the second section of this chapter, one has to know the force acting on them. Hence, it is necessary to assign electric and magnetic fields that are calculated at the discrete grid points to the continuous macro-particle positions. This simulation step is usually referred to as field weighting.

By analogy to the particle weighting, one may exploit the shape functions to calculate the spatial averages of the all electric and magnetic field components,

$$\mathbf{E}^n_p = \sum_{ijk} \mathbf{E}^n_{ijk} S_r\left(\mathbf{r}_{ijk} - \mathbf{r}^n_p\right), \qquad \mathbf{B}^n_p = \sum_{ijk} \mathbf{B}^n_{ijk} S_r\left(\mathbf{r}_{ijk} - \mathbf{r}^n_p\right). \tag{3.34}$$

Note that it is recommended to use the same weighting for both, particles and fields, in order to eliminate a self-force and ensure the conservation of momentum [6].

3.5 Stability and accuracy

The stability and accuracy of the standard PIC method is directly dependent on the size of the spatial and temporal simulation steps. In order to find correct parameters, one has to know the absolute accuracy and corresponding stability conditions.

The effect of the spatial grid is to smooth the interaction forces and to couple plasma perturbations to perturbations at other wavelengths, called aliases. It may lead to non-physical instabilities and numerical heating. To avoid these effects, the spatial step needs to resolve the Debye length (see 2.3). Thus, it is desirable to fulfill the following condition,

$$\Delta x, \Delta y, \Delta z \leqslant \lambda_D. \tag{3.35}$$

In the general electromagnetic case, the time step has to satisfy the Courant–Fridrichs–Levy (CFL) condition [10],

$$C = c^2 \Delta t^2 \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2} \right), \tag{3.36}$$

where the dimensionless number $C \leqslant 1$ is called the CFL number. This condition limits the range of motion of all objects in the simulation during one time step. It ensures, that these particles would not cross more than one cell in one simulation time step. When this condition is violated, the growth of non-physical effects can be very rapid.

The leap-frog scheme, used to solve the field equations and equations of motion, is second-order accurate in both, time and space. In addition, this scheme is explicit and time-reversible. An extensive study of PIC method can be found in [source].

3.6 Extendable PIC Open Collaboration (EPOCH)

The abbreviation EPOCH refers to an Extendable PIC Open Collaboration project [1]. EPOCH is a multi-dimensional, relativistic, electromagnetic code designed for plasma physics simulations based on the PIC method. The code, which has been developed at University of Warwick, is written in FORTRAN and parallelized using MPI library. EPOCH is able to cover physical processes that may take place at ultra-high laser intensities, such as barrier suppression ionization, quantum electrodynamics emission and pair production.

The main features include dynamic load balancing option for making optimal use of all processors when run in parallel, allowing restart on an arbitrary number of processors. The setup of EPOCH is controlled through a customizable input deck. An input deck is a text file which can be used to set simulation parameters for EPOCH without necessity to edit or recompile the source code. Most aspects of a simulation can be controlled, such as the number of grid points in the simulation domain, the initial distribution of particles and the initial electromagnetic field configuration. In addition, EPOCH has been written to add more modern features and to structure the code in such a way that the future expansion of the code may be made as easily as possible. The entire core of the code uses SI units.

By default, EPOCH uses triangular particle shape functions with the peak located at the position of computational particle and a width of two cells, which provides relatively clean and fast solution. However, user can select higher order particle shape functions based on a spline interpolation by enabling compile-time option in the makefile.

The electromagnetic field solver uses a FDTD scheme with second order of accuracy. The field components are spatially staggered on a standard Cartesian Yee cell. The solver is directly based on the scheme derived by Hartmut Ruhl [7]. The particle pusher is relativistic, Birdsall and Landon type [4] and uses Villasenor and Buneman current weighting [14].

EPOCH offers several types of boundary conditions for fields and particles, such as periodic, transmissive, reflecting and also Convolutional Perfectly Matched Layer (CPML) boundary conditions. Laser beams can be attached to arbitrary boundary via special boundary condition as well.

As a side project within this work, the code EPOCH has been instrumented to enable in situ diagnostics and visualization of the electromagnetic fields using ParaView Catalyst [source]. The increasing demands of the simulations need more data to be stored on a disk and analysed. However, the capabilities of computing environment which is responsible for transferring the data and communication have not grown up as rapid as the computational power. Dumping and processing of all the data calculated during the simulation would take too much time, so in practice this usually means that they are stored only at several time steps or at much coarser resolution than the original data. The rest is just discarded and the significant part of information may be potentially lost.

In situ visualization describes techniques where data can be visualized in real-time as it is generated during a simulation and without it being stored on a storage resource. By coupling the visualization and simulation, the data transfer bottleneck can be overcome. Furthermore, this approach allows scientists to monitor and interact with a running simulation, allowing for its parameters to be modified and allowing to immediately view the effects of these changes.

While a simulation is running, a user can see the size of the datasets that a simulation produces. But none of this data is physically stored on a storage system. The computationally expensive operations are carried out using ParaView's graphical interface. So, the user can select data structures and analyze them in the same way as in post-processing. But there is one difference, the simulation is in progress so a user can observe the data as it is being generated. With Catalyst, it is also possible to pause the simulation or specify a break-point at a selected time step. This can be helpful if a user expects some interesting behavior of investigated phenomena or for identifying regions where numerical instability arises. For the implementation details, see appendix B.

The main goal of this work has been to implement a solution that would enable to simulate tightly focused laser beams using simulation code EPOCH. This will be closer described in the following chapter.

Chapter 4

Tight-focusing of laser pulses

The investigation of laser-matter interaction also involves exploring of specific themes of the ultra-relativistic regime, which requires extremely high intensities of the external field. These intensities, that are experimentally inaccessible at present, could be potentially achieved by tight-focusing and that would allow a broad spectrum of many multidisciplinary applications.

As mentioned in the previous chapter, various aspects of electromagnetic interaction are usually studied using sophisticated numerical simulation codes. Vast majority of these codes, however, use a paraxial approximation (closer described in chapter 1) to prescribe the laser fields at the boundaries, and afterwards, a field solver guides the beam across the simulation domain. As already mentioned, the paraxial approximation is valid only if the angular spectrum of laser pulse is sufficiently narrow, therefore it is not possible to simulate tightly focused laser beams using this approach. As might be seen later, paraxial approximation in this case leads to a distorted field profiles which have strong impact on the results of laser-matter interaction.

Several interesting solutions, how to simulate strongly focused beams, have been already proposed [source]. Within this work, a simple and efficient algorithm for a Maxwell consistent calculation of the electromagnetic fields at the boundaries of the computational domain [source] (also called laser boundary conditions) has been used and implemented into the PIC code EPOCH [source]. Note, that this algorithm is able to describe laser beams with arbitrary shape.

4.1 Laser boundary conditions

In this section, another mathematical description of a focused laser beam based on a rigorous solution of the wave equation 1.39 is presented. All the following calculations are reproduced from the excellent work of Illia Thiele et al. [source].

Assume that the laser beam propagates in vacuum without external sources along the z-axis of the Cartesian coordinate system. The wave equation 1.39 in temporal Fourier space has the following form,

$$\Delta\hat{\mathbf{E}}\left(\mathbf{r},\omega\right)+\frac{\omega^{2}}{c^{2}}\hat{\mathbf{E}}\left(\mathbf{r},\omega\right)=0, \tag{4.1}$$

where the hat symbol placed on the top of a variable denotes the Fourier transform with respect to time. Next, one shall perform a spatial Fourier transform of 4.1 with respect to transverse coordinates $\mathbf{r}_\perp$ only,

$$\left(-k_x^2-k_y^2+\frac{\partial^2}{\partial z^2}\right)\bar{\mathbf{E}}\left(k_x,k_y,z,\omega\right)+\frac{\omega^2}{c^2}\bar{\mathbf{E}}\left(k_x,k_y,z,\omega\right)=0, \tag{4.2}$$

where the bar symbol placed on the top of a variable denotes the Fourier transform with respect to time and spatial transverse coordinates. The equation 4.2 can be simplified as follows,

$$k_z^2\left(\mathbf{k}_\perp,\omega\right)\bar{\mathbf{E}}(\mathbf{k}_\perp,z,\omega)+\frac{\partial^2}{\partial z^2}\bar{\mathbf{E}}\left(\mathbf{k}_\perp,z,\omega\right)=0. \tag{4.3}$$

where $k_z\left(\mathbf{k}_\perp,\omega\right)=\sqrt{-\mathbf{k}_\perp^2+\omega^2/c^2}$ and $\mathbf{k}_\perp=(k_x,k_y)^{\mathrm{T}}$. The fundamental solution of the equation 4.3 consists of the forward (+) and backward (−) propagating waves,

$$\bar{\mathbf{E}}^{\pm}\left(\mathbf{k}_\perp,z,\omega\right)=\bar{\mathbf{E}}_0^{\pm}\left(\mathbf{k}_\perp,\omega\right)\mathrm{e}^{\pm\mathrm{i}k_z(\mathbf{k}_\perp,\omega)(z-z_0)}. \tag{4.4}$$

Where $\bar{\mathbf{E}}_0^{\pm}\left(\mathbf{k}_\perp,\omega\right)$ is the electric laser field at some plane $z=z_0$. It might be clearly seen, that only two out of six vector components of the electric and magnetic fields are independent, therefore one may prescribe for example the transverse components $\bar{\mathbf{E}}_{0,\perp}^{\pm}\left(\mathbf{k}_\perp,\omega\right)$ at the plane $z=z_0$ and all other components can be derived from the Maxwell's equations 1.1, 1.3,

$$\bar{\mathbf{E}}_\perp^{\pm}\left(\mathbf{k}_\perp,z,\omega\right)=\bar{\mathbf{E}}_{0,\perp}^{\pm}\left(\mathbf{k}_\perp,\omega\right)\mathrm{e}^{\pm\mathrm{i}k_z(\mathbf{k}_\perp,\omega)(z-z_0)}, \tag{4.5}$$

$$\bar{E}_z^{\pm}\left(\mathbf{k}_\perp,z,\omega\right)=\mp\frac{\mathbf{k}_\perp\cdot\bar{\mathbf{E}}_\perp^{\pm}(\mathbf{k}_\perp,z,\omega)}{k_z\left(\mathbf{k}_\perp,\omega\right)}, \tag{4.6}$$

$$\bar{\mathbf{B}}_\perp^{\pm}\left(\mathbf{k}_\perp,z,\omega\right)=\frac{1}{\omega k_z\left(\mathbf{k}_\perp,\omega\right)}\mathbb{R}^{\pm}\left(\mathbf{k}_\perp,\omega\right)\bar{\mathbf{E}}^{\pm}\left(\mathbf{k}_\perp,z,\omega\right), \tag{4.7}$$

where

$$\mathbb{R}^{\pm}\left(\mathbf{k}_\perp,\omega\right)=\begin{pmatrix}\mp k_xk_y & \mp\left[k_z^2\left(\mathbf{k}_\perp,\omega\right)+k_y^2\right] & 0\\ \pm\left[k_z^2\left(\mathbf{k}_\perp,\omega\right)+k_x^2\right] & \pm k_xk_y & 0\\ -k_yk_z\left(\mathbf{k}_\perp,\omega\right) & -k_xk_z\left(\mathbf{k}_\perp,\omega\right) & 0\end{pmatrix}. \tag{4.8}$$

Analogically, one could solve the wave equation for the magnetic field 1.40, prescribe two transverse components of $\bar{\mathbf{B}}_{0,\perp}^{\pm}\left(\mathbf{k}_\perp,\omega\right)$ at the plane $z=z_0$ and afterwards calculate all other fields using Maxwell's equations 1.2, 1.4. The complete proof, that the fields 4.5 - 4.7

are consistent with the Maxwell's equations in vacuum 1.1 - 1.4 can be found in the original paper [source].

Note that for $k_\perp^2 > \omega^2/c^2$, $k_z(\mathbf{k}_\perp, \omega)$ becomes imaginary and equation 4.4 describes evanescent waves that are unphysical in free space. Thus the Fourier spectrum of laser waves has to be filtered in the transverse Fourier space. On the other hand, if the spatial Fourier spectrum contains only components with $k_\perp^2 \ll \omega^2/c^2$, then $k_z(\mathbf{k}_\perp, \omega)$ can be approximated using the first few terms of a Taylor series,

$$k_z(\mathbf{k}_\perp, \omega) \approx \frac{|\omega|}{c} - \frac{c}{2|\omega|} k_\perp^2. \tag{4.9}$$

Note that by plugging 4.9 into equations 4.5 - 4.7 one gets the paraxial approximation.

In the last part of this section, the practical algorithm for implementation of the boundary conditions based on the previously derived solution of Maxwell's equations is presented. Assume that the laser beam propagates in a forward direction along the z-axis. In the beginning, it is necessary to prescribe the electric laser field $\mathbf{E}_{0,\perp}(\mathbf{r}_\perp, t)$ in the plane $\mathcal{P}$ at $z = z_0$. Note, that it can be defined by arbitrary function of space and time. The goal is then to find the fields $\mathbf{E}_\mathrm{B}(\mathbf{r}_\perp, t)$ and $\mathbf{B}_\mathrm{B}(\mathbf{r}_\perp, t)$ at the corresponding boundary $z = z_\mathrm{B}$.

Consider that the transverse part of simulation domain is made of equidistant rectangular grid described by x^i, y^j, where $i, j \in \{1, \ldots, N_{x,y}\}$, and the grid steps δx, δy. The simulation time t^n, where $n \in \{1, \ldots, N_t\}$, is also divided into equidistant time steps of size δt.

The algorithm allows to calculate fields $\mathbf{E}_\mathrm{B}^{ij}(t)$ and $\mathbf{B}_\mathrm{B}^{ij}(t)$ for any given time t from the interval $\left[t^1 - \frac{z_\mathrm{B} - z_0}{c}, t^{N_t} - \frac{z_\mathrm{B} - z_0}{c}\right]$. In order to preserve clarity, the algorithm below is given in the exact form as in the original paper [source].

1. Calculate $\hat{\mathbf{E}}_{0,\perp}^{ijn}$ via discrete Fourier transforms in time:

$$\omega^n = \frac{2\pi}{N_t \delta t}\left(-\frac{N_t}{2} + n\right), \tag{4.10}$$

$$\hat{\mathbf{E}}_{0,\perp}^{ijn} = \frac{\delta t}{2\pi} \sum_{l=1}^{N_t} \mathbf{E}_{0,\perp}^{ijl} \mathrm{e}^{\mathrm{i}\omega^n t^l}, \quad n \in \{1, \ldots, N_t\}. \tag{4.11}$$

2. Calculate $\bar{\mathbf{E}}_{0,\perp}^{ijn}$ via two-dimensional discrete Fourier transforms in transverse space:

$$k_x^i = \frac{2\pi}{N_x \delta x}\left(-\frac{N_x}{2} + i\right), \quad k_y^j = \frac{2\pi}{N_y \delta y}\left(-\frac{N_y}{2} + j\right), \tag{4.12}$$

$$\bar{\mathbf{E}}_{0,\perp}^{ijn} = \frac{\delta x \delta y}{(2\pi)^2} \sum_{l,m=1}^{N_x, N_y} \hat{\mathbf{E}}_{0,\perp}^{lmn} \mathrm{e}^{-\mathrm{i}\left(k_x^i x^l + k_y^j y^m\right)}, \quad i, j \in \{1, \ldots, N_{x,y}\}. \tag{4.13}$$

3. Calculate transverse electric field components at the boundary $z = z_{\mathrm{B}}$:

$$k_z^{ijn} = \Re\sqrt{\frac{(\omega^n)^2}{c^2} - (k_x^i)^2 - (k_y^j)^2}, \tag{4.14}$$

$$\bar{\mathbf{E}}_{\mathrm{B},\perp}^{ijn} = \begin{cases} \bar{\mathbf{E}}_{0,\perp}^{ijn} \mathrm{e}^{\mathrm{i}k_z^{ijn}(z_{\mathrm{B}}-z_0)} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{4.15}$$

4. Calculate longitudinal electric field components at the boundary $z = z_{\mathrm{B}}$:

$$\bar{E}_{\mathrm{B},z}^{ijn} = \begin{cases} -\frac{k_x^i \bar{E}_{\mathrm{B},x}^{ijn} + k_y^j \bar{E}_{\mathrm{B},y}^{ijn}}{k_z^{ijn}} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{4.16}$$

5. Calculate the magnetic field at the boundary $z = z_{\mathrm{B}}$:

$$\mathbb{R}^{ijn} = \begin{pmatrix} -k_x^i k_y^j & (k_x^i)^2 - (\omega^n)^2/c^2 \\ (\omega^n)^2/c^2 - (k_y^j)^2 & k_x^i k_y^j \\ -k_y^j k_z^{ijn} & k_x^i k_z^{ijn} \end{pmatrix}, \tag{4.17}$$

$$\bar{\mathbf{B}}_{\mathrm{B}}^{ijn} = \begin{cases} (\omega^n k_z^{ijn})^{-1} \mathbb{R}^{ijn} \bar{\mathbf{E}}_{\mathrm{B},\perp}^{ijn} & \text{for } k_z^{ijn} > 0 \\ 0 & \text{for } k_z^{ijn} = 0 \end{cases}. \tag{4.18}$$

6. Calculate $\hat{\mathbf{E}}_{\mathrm{B}}^{ijn}$, $\hat{\mathbf{B}}_{\mathrm{B}}^{ijn}$ via two-dimensional inverse discrete Fourier transforms:

$$\hat{\mathbf{E}}_{\mathrm{B}}^{ijn} = \frac{(2\pi)^2}{N_x N_y \delta x \delta y} \sum_{l,m=1}^{Nx,Ny} \bar{\mathbf{E}}_{\mathrm{B}}^{lmn} \mathrm{e}^{\mathrm{i}(k_x^l x^i + k_y^m y^j)}, \tag{4.19}$$

$$\hat{\mathbf{B}}_{\mathrm{B}}^{ijn} = \frac{(2\pi)^2}{N_x N_y \delta x \delta y} \sum_{l,m=1}^{Nx,Ny} \bar{\mathbf{B}}_{\mathrm{B}}^{lmn} \mathrm{e}^{\mathrm{i}(k_x^l x^i + k_y^m y^j)}. \tag{4.20}$$

7. Calculate $\mathbf{E}_{\mathrm{B}}^{ij}(t)$, $\mathbf{B}_{\mathrm{B}}^{ij}(t)$ for any given time $t \in \left[t^1 - \frac{z_{\mathrm{B}}-z_0}{c}, t^{N_t} - \frac{z_{\mathrm{B}}-z_0}{c}\right]$.

$$\mathbf{E}_{\mathrm{B}}^{ij}(t) = \frac{2\pi}{N_t \delta t} \sum_{n=1}^{N_t} \hat{\mathbf{E}}_{\mathrm{B}}^{ijn} \mathrm{e}^{-\mathrm{i}\omega^n t}, \tag{4.21}$$

$$\mathbf{B}_{\mathrm{B}}^{ij}(t) = \frac{2\pi}{N_t \delta t} \sum_{n=1}^{N_t} \hat{\mathbf{B}}_{\mathrm{B}}^{ijn} \mathrm{e}^{-\mathrm{i}\omega^n t}. \tag{4.22}$$

4.2 Implementation

One of the main goals of this work has been to implement the algorithm mentioned in the previous section, to evaluate its correctness in several test simulations and finally, to exploit resulting implementation for simulations of tightly focused Gaussian beams in laser-matter interaction. The main requirement on implementation has been easy to use with 2D version of particle-in-cell simulation code EPOCH [source]. For this reason, several possible solutions has been taken into account.

The final decision has been to create a static library, which will be able to compute desired quantities and will provide functions for communication with the main simulation code. The essential advantage is that it could be basically linked with any laser-plasma simulation code. Also, since it is necessary to call only two additional functions, the instrumentation will be fast, easy and the main simulation code will not be excessively disturbed. Furthermore, the implementation itself come with the CMake [source] support, which simplify the compilation process using platform and compiler independent configuration files.

The library has been written in C++ language and is object oriented so the algorithm can be easily extended to three dimensional geometry. In order to speedup the whole underlying computation, the algorithm has been parallelized using hybrid techniques. The time domain has been decomposed into the stripes corresponding to individual computational processes, the communication between these processes is ensured by MPI library. Furthermore, the computationally most expensive cycles are parallelized using OpenMP implementation of multi-threading. Later on, the speedup and parallel scaling performance will be briefly discussed.

Fourier transforms form the core of the computational process and their performance is crucial for the overall performance of the code. For this reason, many currently available libraries have been considered. Eventually, the Fourier transforms in the algorithm can be computed using FFTW [source] library, Intel® MKL [source] library or it is also possible to directly evaluate the formulas without using any additional library. The user specifies his option before the compilation. Regarding both libraries, a threaded versions of 1D in-place complex fast Fourier transforms have been used throughout the code. According to several measures, there is no significant difference between the speed of both implementations.

One potential bottleneck could happen during the computation of spatial Fourier transforms since the arrays with spatial data are decomposed into different processes. The cluster versions of functions performing the Fourier transforms have been tested, however they did not bring any significant speedup. The reason is as follows. They require to have the global array in memory and use its own distribution which involves overlapping. Since the size of global arrays is usually not so large and since it is necessary to perform a lot of different Fourier transforms, the majority of computational time is spent rather for communication, mainly if many of computational cores are used.

This issue has been solved by gathering the data on master process, performing the Fourier transforms in space by only one processor and scattering the data back to corresponding processes. This is the reason why the code does not scale well, however, the time to compute all desired quantities is in most cases negligible in comparison with the time required by main simulation cycle. Nevertheless, this issue could be improved in future.

Since it is necessary to compute the whole time evolution of the laser field at boundary for each grid point before the simulation starts, the resulting amount of data can be significantly large and does not have to fit in a computer memory. Thus, it is inevitable to dump the data into a file, which will be then accessed by the main simulation code. Due to the performance purposes, each computational process stores its data into a shared file with corresponding offset and in binary coding. Therefore, the output operations are as fast as possible and save the storage resources. Library then provide a function which allows to seek an arbitrary position in a file. This function is then called each time step of the main simulation loop to fill the laser source arrays with all the relevant data. This way of accessing data does not cause any significant slowdown or memory overhead.

The EPOCH [source] code require only transverse components of laser electric field, all other quantities are computed by the FDTD solver. The implementation of the library allows fully connection with EPOCH [source]. In practice, if the user wants to simulate tight-focusing, is is necessary to enable the corresponding flag as a compile-time option and then to specify all required parameters in the input file. The code then automatically computes all necessary data. It works generally regardless the number of lasers in the simulation or boundaries that they are attached to.

The current version of library does not work for obliquely incident laser pulses, because in this case one cannot exploit the advantage of an efficient computation with fast Fourier transforms. However, the code allows to compute the laser fields at boundary by evaluating Fourier integrals directly, so it could be easily extended. Second, it is at the moment possible to simulate only Gaussian laser pulses. However, the user can easily prescribe its own shape and position of the beam in focal plane by modifying corresponding part of the code.

Several most important data structures, functions and methods that form the core of the library for tight-focusing can be seen in appendix B.

4.3 Evaluation

To evaluate the correctness of the algorithm presented in the previous section of this chapter as well as to demonstrate the drawbacks of the paraxial approximation, several test simulations in 2D geometry have been performed. In the following text, a two limit cases are presented. The first pair of simulations employs tightly focused Gaussian laser beam with the size at focus comparable with the center laser wavelength, whilst the second one shows the case of the Gaussian beam with the size at focus one order of magnitude larger than the center laser wavelength, where both approaches should return identical results. Note, that all the simulations have been computed using 2D version of PIC code EPOCH [source] instrumented with library for tight-focusing.

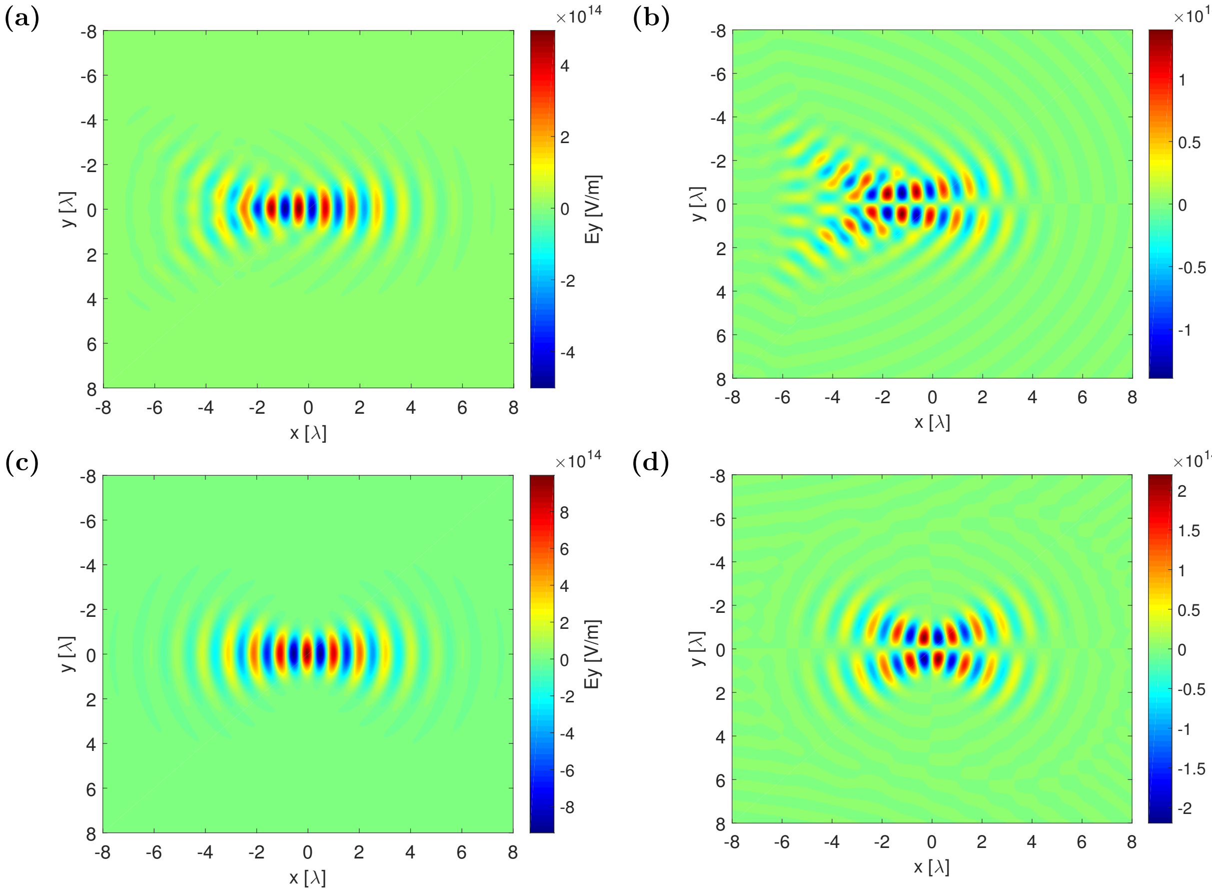

Figure 3: Transverse (E_y) and longitudinal (E_x) electric laser field components captured at the time step of their maximal intensity at the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. In the case of paraxial approximation, both components reveal strong distortions and asymmetry, their focal spot is located about 1λ closer to the left boundary than specified and the corresponding amplitude is significantly lower. The laser has been attached to the left hand side boundary.

First, have a look at the simulation of a tightly focused Gaussian beam. The p-polarized laser pulse with center wavelength $\lambda = 1\,\mu$m propagates from left hand side boundary to the right. Its duration has been chosen to $\tau = 20\,$fs in FWHM and amplitude $E_0 = 1 \cdot 10^{15}\,$V/m. The beam waist $w_0 = 0.7\,\mu$m is shorter than the laser wavelength, which implies that non-negligible parts of $\bar{\mathbf{E}}_{0,\perp}(k_x, \omega)$ are evanescent. The focus is located at a distance $x = 8\,\mu$m from the boundary that the laser is attached to.

The size of the simulation domain is $16\lambda \times 48\lambda$, with 100 cells per laser wavelength in both directions, thus $\Delta x = \Delta y = \lambda/100 = 10\,$nm. The one simulation time step is according to CFL condition $\Delta t = 0.95\sqrt{2}\lambda/100c \approx 0.05\,$fs, the whole simulation time is then $t = 150\,$fs.

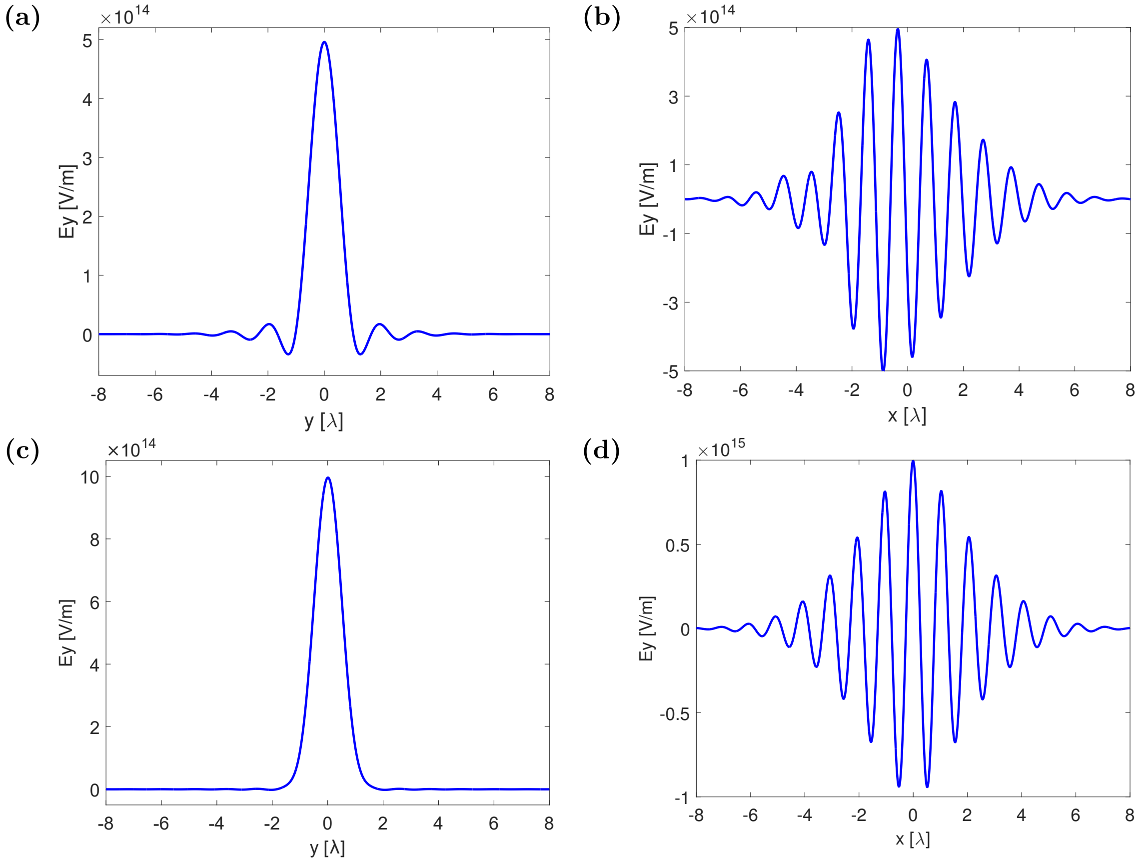

Figure 4: Transverse **(a)**, **(c)** and longitudinal **(b)**, **(d)** slices of the transverse electric laser field (E_y) at the time step when it reaches maximal intensity at the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. In the case of paraxial approximation, one can clearly see strong side-wings in the spatial beam profile **(a)** as well as the asymmetry of the field in the longitudinal line-out **(b)**.

The pulse propagates in vacuum in order to get rid of all effects that could be potentially caused by plasma. All the simulation parameters can be found in the attached input files in the appendix A.

In the following paragraph, the results of the first simulation are discussed in a more detail. Fig. 3 shows transverse and longitudinal electric field components at their maximal intensity at focus for both cases, laser beam propagating under the paraxial approximation (Fig. 3 - a, b) and according to the approach consistent with Maxwell equations (Fig. 3 - c, d). In the case of paraxial approximation, one can clearly see strong distortions and asymmetry in the shape of both electric field components. In addition, the focus location is shifted about 1 μm closer to the left boundary and the corresponding amplitude at focus is less than half the required value. In contrast, the fields produced by the simulation using

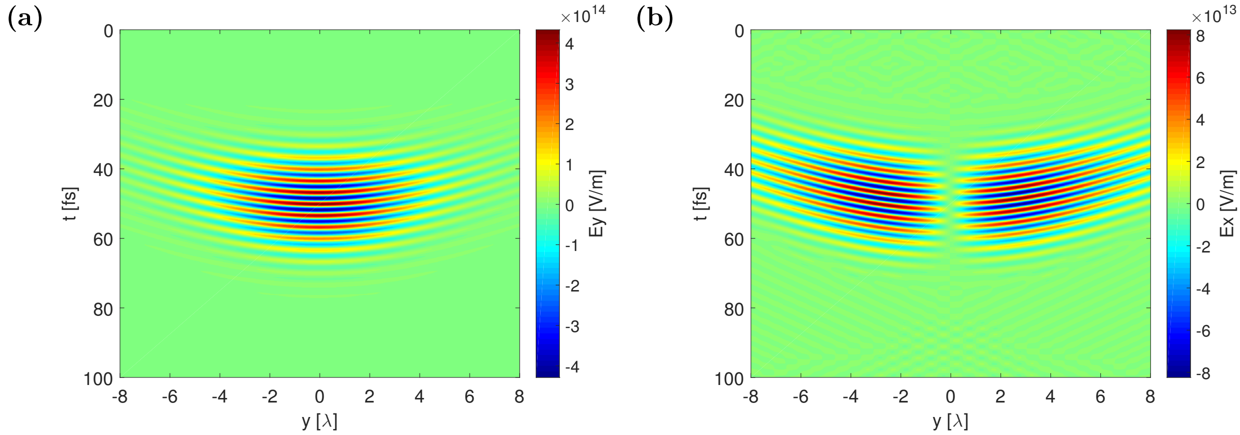

Figure 5: The time evolution of transverse (E_y) **(a)** and longitudinal (E_x) **(b)** electric laser field components at the boundary that the laser is attached to. Both components has been calculated according to the Maxwell consistent approach.

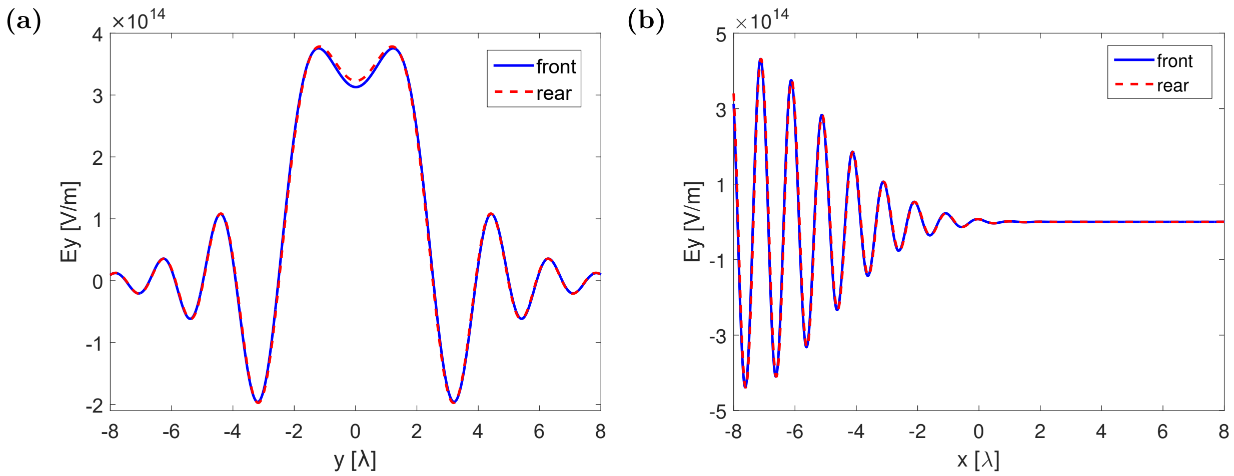

Figure 6: Transverse **(a)** and longitudinal **(b)** slice of the transverse electric laser field (E_y) when it reaches its maximal intensity at the front (blue) and rear (red) boundary. The results come from the simulation where the Maxwell consistent approach for laser propagation has been used. For better comparison, the field at the rear boundary in **(b)** has been horizontally flipped. The exact match between the field shapes at a different time steps of simulation proves the correctness of the laser beam propagation.

Maxwell consistent calculation of laser fields at boundary are symmetric with respect to the focal spot and without any distortions. Furthermore, the focus location as well as the amplitude fulfills the initial requirements precisely.

Fig. 4 shows transverse and longitudinal slices of transverse electric field component at focus for the case of laser beam propagating under the paraxial approximation (Fig. 4 - a,

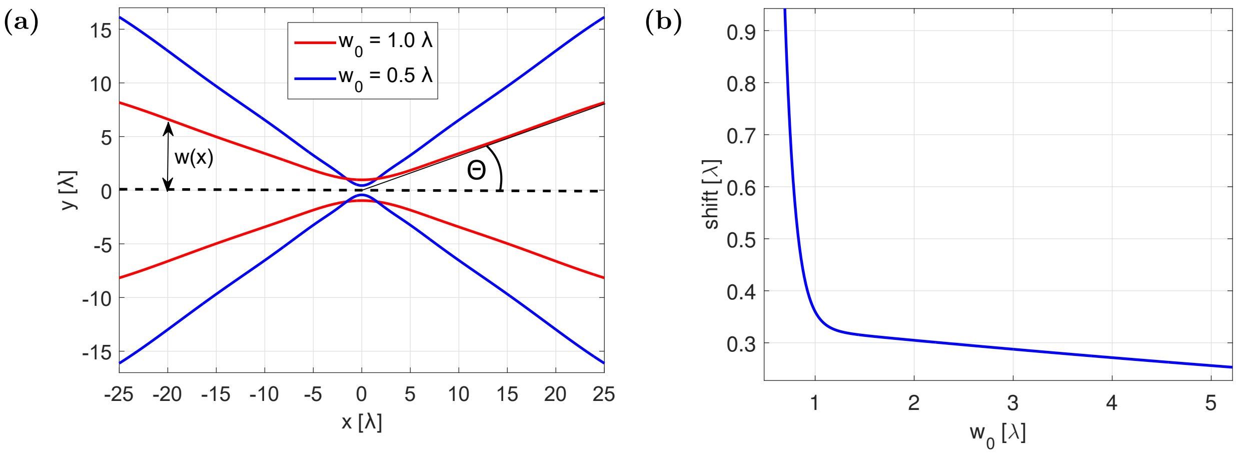

Figure 7: (a) Graph of the spot size parameter $w(x)$ of the beams with $w_0 = 1\lambda$ and 0.5λ calculated using Maxwell consistent approach. From the plotted lines, one can roughly estimate the beam divergence angle Θ. The divergence angles for both beams are surprisingly in a good accordance with the formula for divergence angles of corresponding Gaussian beams. **(b)** Dependency of the absolute focal point shift on the beam waist w_0 for the laser beam propagating under the paraxial approximation. One can see sharp increase of the error for $w_0 < \lambda$. In both cases, values have been dumped at discrete time intervals and interpolated.

b) as well as for the case where the beam propagation has been resolved within the Maxwell consistent approach (Fig. 4 - c, d). For the case of paraxial approximation, one can clearly see the asymmetry of the field shape in the longitudinal slice (Fig. 4 - b), which consequently leads to a decrease of the amplitude at focus and to the strong side-wings in the spatial beam profile, as might be better seen from the transverse slice (Fig. 4 - a). On the other hand, Maxwell consistent approach calculates fields of perfect symmetry with respect to the focal spot (Fig. 4 - c) and no side-wings or distortions are present (Fig. 4 - d).

In Fig. 5 one can examine the time evolution of transverse (Fig. 5 - a) and longitudinal (Fig. 5 - b) electric field components at boundary as computed using Maxwell consistent approach. Note, that one have to chose carefully the transverse size of the domain, since the beam width at boundary may be much larger than at focus because of a diffraction. To estimate the beam width at boundary, one has to know the beam divergence angle Θ. In Fig. 7 - a, one can find a graph of the spot size parameter w as calculated using the Maxwell consistent approach for the beams focused to $w_0 = 1\ \lambda$ and $0.5\ \lambda$. From the plotted lines, one can roughly estimate the beam divergence angle Θ. For the beam focused to $w_0 = 1\ \lambda$, the beam divergence angle has been around $\Theta = 18.2°$ which is almost identical value as for the Gaussian beam of the same parameters. For the beam with $w_0 = 0.5\ \lambda$ the divergence has been estimated to $\Theta = 32.9°$, while the Gaussian beam with the same size at focus has divergence $\Theta = 36.5°$. Consequently, in spite of the fact that the divergence of the tightly focused beams calculated using the Maxwell consistent approach is a little bit lower, both

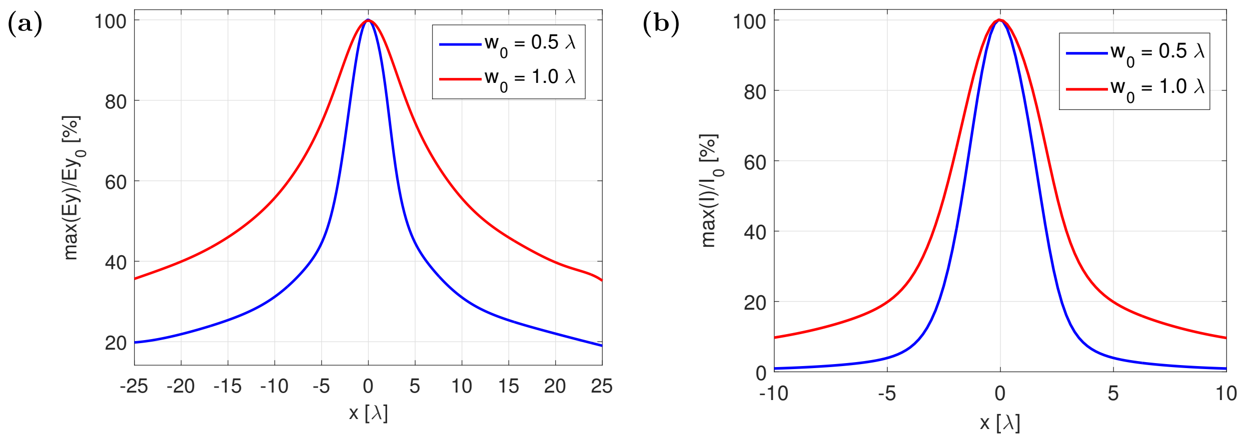

Figure 8: (a) Graph of the transverse (E_y) electric laser field amplitude with respect to the distance from focal spot. The values on vertical axis are given in a percentage of the amplitude at focus $E_{0,y}$. **(b)** Graph of the maximal instantaneous laser intensity with respect to the distance from focal spot. The values on vertical axis are given in a percentage of the laser intensity at focus I_0. In both cases, values have been computed using the Maxwell consistent approach, dumped at discrete time intervals and interpolated.

approaches are in quite a good accordance regarding this parameter.

To evaluate a correctness of the beam propagation using Maxwell consistent approach, several criteria has been defined. The correctness of the amplitude and beam waist as well as the right focus location has already been verified. Additional criteria has been set on a beam symmetry. In Fig. 6, one can find a comparison of the transverse electric laser field component when it achieves its maximal intensity at front and rear boundary. One can clearly see the exact match between the field shapes at different time steps of the simulation in transverse (Fig. 6 - a) and longitudinal (Fig. 6 - b) slices. Moreover, all the aforementioned criteria has been fulfilled also in other simulations with different input parameters that are not presented here. Consequently, these observations prove the correctness of the propagation at least for the tightly focused Gaussian laser beams.

In Fig. 7 - b, one may find the estimate of the absolute focal point shift with respect to the beam waist. The interpolated values come from the simulations where the laser beam propagation had been resolved using paraxial approximation. One can clearly see the sharp increase of the error for $w_0 < \lambda$. On the other hand, for $w_0 = 5\lambda$ the shift is around 0.2λ which is generally considered to be acceptable.

In Fig. 8 - a, one can see the amplitude of the transverse electric laser field with respect to the distance from the focal spot calculated using Maxwell consistent approach. This could be particularly useful for experimenters since it is not always easy to focus the beam onto the target surface perfectly. Sometimes, one might be more interested in the laser intensity rather than the amplitude of the electric field. For this reason, Fig. 8 - b shows the instantaneous maximal laser intensity in the narrower spatial interval calculated from the values of the

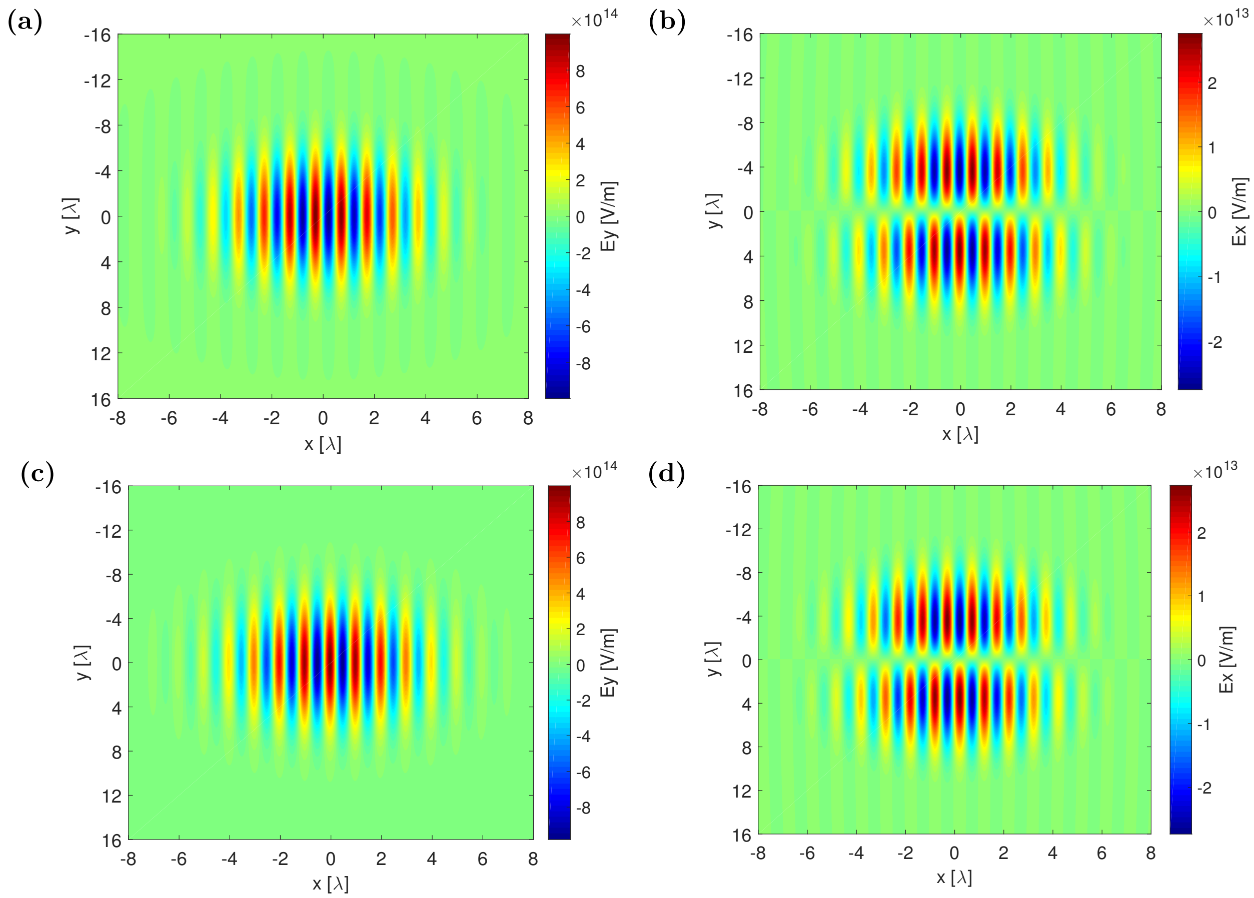

Figure 9: Transverse (E_y) and longitudinal (E_x) electric laser field components captured at the time step of their maximal intensity at the focal spot. The cases **(a)**, **(b)** correspond to the laser pulse propagating under the paraxial approximation, whilst **(c)**, **(d)** come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. The size of the focus has been chosen to be one order of the magnitude larger than the center laser wavelength. One can clearly see, that there is no significant difference between the shapes of the electric field components.

electric field amplitude.

For the second simulation, all the input parameters remained the same except the beam waist. Now, the parameter $w_0 = 5\,\mu$m, which is about the limit case for the beam propagating under the paraxial approximation. Thus, one expects that the simulation results should be almost identical.

Similarly as in Fig. 3, Fig. 9 shows again transverse and longitudinal electric field components at their maximal intensity at focus for both cases, laser beam propagating under the paraxial approximation (Fig. 9 - a, b) and according to the approach consistent with Maxwell equations (Fig. 9 - c, d). Here, one cannot register any difference between the re-

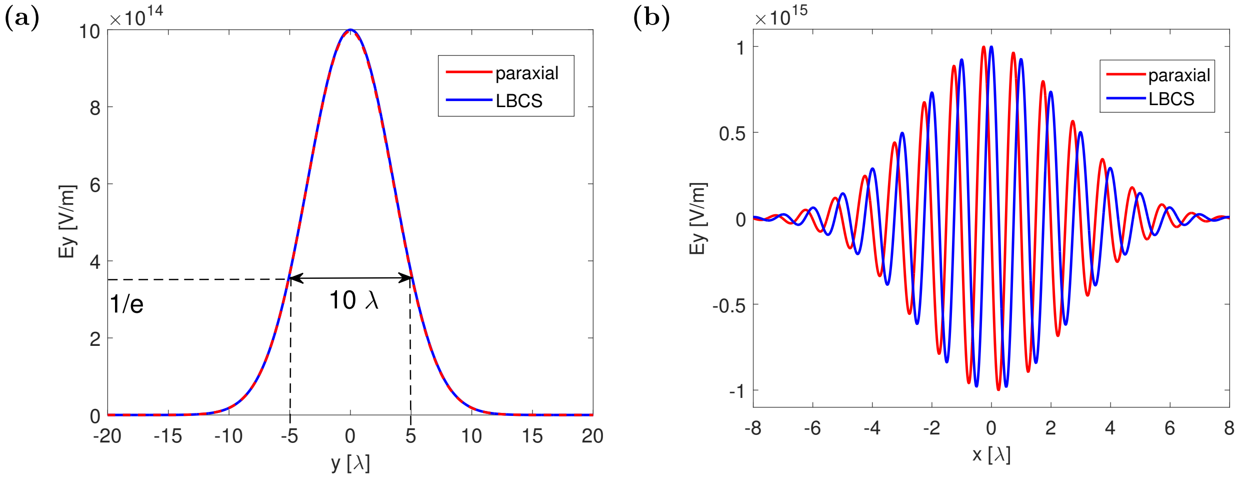

Figure 10: Transverse **(a)** and longitudinal **(b)** slices of the transverse electric laser field (E_y) at the time step when it reaches maximal intensity at focus. Red lines correspond to the laser pulse propagating under the paraxial approximation, whilst blue lines come from the simulation where the beam propagation has been resolved within the Maxwell consistent approach. The size of the focus has been chosen to be one order of magnitude larger than the center laser wavelength. In the case of paraxial approximation, the focus is slightly shifted closer to the left boundary **(b)**, otherwise the size of the focus as well as the amplitude is correct for both cases **(a)**.

sults corresponding to both approaches. Also, the transverse slice of the transverse electric laser field component at focus (Fig. 10 - a) shows the correct shape and amplitude for both cases. The longitudinal slice (Fig. 10 - b), however, points out the fact that the location of the focal spot is still a little bit shifted closer to the left hand side boundary. Nevertheless, this difference could be in practice neglected. At the end of the day, for the Gaussian beams propagating under the paraxial approximation, the beam diameter at focus should be at least one order of magnitude larger than the center laser wavelength.

In conclusion, one should be aware that the propagation of tightly focused laser pulses cannot be described by paraxial approximation. Above, it has been shown that for the beams focused to a spot with the size comparable to a center laser wavelength, paraxial approximation leads to a shifted location of the focus, asymmetric laser field profiles with distortions and lower amplitude. These deviations are far from negligible and have without doubt a strong impact on the laser-matter or laser-plasma interaction results. On the other hand, the propagation of tightly focused Gaussian laser beams prescribed at boundaries according to the Maxwell consistent approach has been proven to be correct.

Also, it has been verified that the paraxial approximation can be safely used when the beam size at focus is about one order of magnitude larger than the center laser wavelength, thus $w_0 \geqslant 5\lambda$. In this case, the laser fields are symmetric with respect to the focal point, the laser intensity fulfills specified requirements precisely, only the location of the focus is shifted about 0.2λ closer to the boundary on which the laser is attached to. However, in this case, the simulations generally produce physically relevant results, therefor the effect of the focal point shift can be in practice neglected.

4.4 Overview of experimental methods

In this section, basic approaches how to focus a laser beam in experiments are discussed. Before providing a brief overview of currently used methods, it is necessary to introduce several most fundamental parameters that describe any optical system. First, one defines a numerical aperture of an optical system NA, which is a dimensionless number characterizing the range of angles over which the system can accept or emit light. Second characteristic of any optical system is a focal length f describing a distance between the center of the aperture and the point in space over which collimated light rays are brought to a focus. Finally, one defines a f-number $f/\#$ as a ratio of the focal length f to the size of the aperture D. The f-number is dimensionless and stands for a quantitative measure of a speed of the optical system.

In experiments, focusing of laser beams is usually achieved by means of an optical system, such as a lens or a curved mirror. However, when dealing with a short pulses, lenses are not preferred because they may affect the beam by frequency-dependent effects, such as chromatic aberration and other nonlinear optical distortions. On the other hand, reflective optics are generally able to produce a diffraction limited image without chromatic effects.

An off-axis parabolic mirror is a frequently used tool to focus an incoming collimated beam. It is made by cutting out a small section from a full parabolic mirror and thus it allows to deviate the beam path off the optical axis. Therefore, the focal point is at more accessible location and the target is not blocking the incoming collimated laser beam as in the case of a complete parabolic mirror. Obviously, the off-axis parabolic mirror is able to work reversibly, so it can take the light coming from a point source and produce a collimated beam. These physical properties make the off-axis parabolic mirror a valuable tool for many different optical purposes.

However, as soon as the laser intensity exceeds 10^{13} W/cm^2, the surface of any material becomes strongly ionized. Therefore, in order to keep the energy density on the optical components below the damage threshold, the beam diameter has to be increased. This approach inherently imposes limits upon the geometrical characteristics of the focused beam such as the size of the focus. The focal spot can be alternatively decreased by implementing a small f-number focusing optic. However, since their focal length is inevitably very short and therefore they must be placed in close proximity to the interaction region, additional care has to be taken to protect the optical components because they can be easily damaged from debris induced by the exploded target flow. Note that such optics are typically very expensive.

In the case of tight-focusing, the conventional solid state optics seems to be inappropriate. Nevertheless, many of the drawbacks mentioned above might be overcome by using a plasma-based optics. As already mentioned in the chapter 2, when the laser pulse with

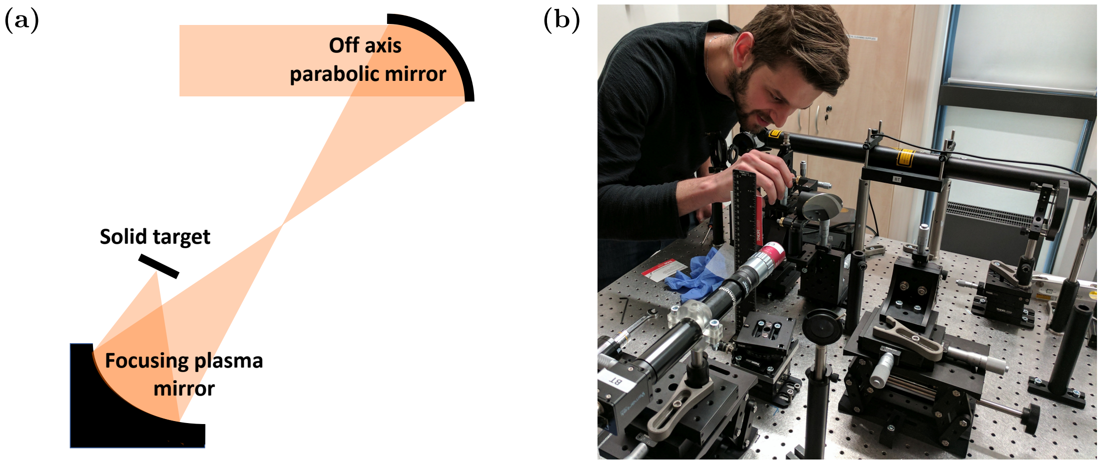

Figure 11: (a) Schematic diagram showing the operation of the focusing plasma mirror where the incoming laser beam is focused by a conventional off-axis parabolic mirror. The plasma mirror focuses the beam onto a target surface. **(b)** Photo capturing a very long and frustrating process of aligning an off-axis parabolic mirror.

intensity higher than 10^{16} W/cm^2 hits a solid target, a thin dense plasma layer is immediately formed on its surface. Under certain conditions, the plasma becomes dense enough and the reflectivity of otherwise transparent target strongly increases. The incident laser light is reflected at the critical plasma density, thus the dense plasma acts as a mirror. These optical components are known as plasma mirrors.

Plasma mirrors are able to operate under much higher energy density than the conventional solid state optics, therefore the optics aperture diameter can be more than one order of magnitude smaller. Consequently, plasma mirrors can be mass-produced at much lower cost, which is crucial since they are single-use and thus have to be replaced after each shot. In addition, the concerns about damage from target debris are naturally eliminated.

To induce focusing, the surface of plasma mirror has to be curved similarly as in conventional optics. However, since the distance between the mirror and the interaction region can be significantly reduced, the corresponding f-number can be extremely small enabling to focus an incident laser beam to a spot size smaller than the center laser wavelength. In addition, plasma mirrors are attractive also due to their ability to improve the temporal and spatial contrast ratio of the laser pulse, which is crucial for many applications of the laser-matter interaction.

The diagram in Fig. 11 schematically shows an experimental configuration for tight-focusing. A conventional off-axis parabolic mirror is aligned in a such way that the focus coincides with the one focal position of the ellipsoidal focusing plasma mirror. The ellipsoidal shape is chosen because it possesses no spherical aberration and

Chapter 5

Simulation results

Zeroth set (4 simulations with const. E = 2.8306e4 J, traget 2 micron, 2000 ppc): Laser:

- wavelength: $\lambda = 0.8\ \mu m$
- const. energy: E = 2.8306e4 J (corresponds to I = 1e20 W/cm2 for $w_0 = 1.0\ \mu m$)
- duration: t = 30 fs (in FWHM)
- beam waist in focus: $w_0 = 0.5, 1.0, 2.0, 4.0\ \mu m$
- focus distance from boundary: $x_B - x_0 = 8\ \mu m$
- polarization: P
- boundary: left

Domain:

- x min: 0 μm
- x max: 15 μm
- y min: -20 μm
- y max: 20 μm
- N_x: 1875 cells ($\delta x = \lambda/100 = 8$ nm)
- N_y: 5000 cells ($\delta y = \lambda/100 = 8$ nm)
- time step: $\delta t = 1/(\sqrt{2}c)\lambda/100 \approx 0.05$ fs
- simulation time: $\tau = 200$ fs

Target:

- x min: 8 μm
- x max: 10 μm
- y min: -15 μm
- y max: 15 μm
- electrons: 2000 ppc
- protons: 100 ppc
- density: 100 critical
- temperature: 100 eV

First set (4 simulations with const. I = 1e20 W/cm2, traget 2 micron, 2000 ppc): Laser:

- wavelength: $\lambda = 0.8$ μm
- const. intensity: I = 1e20 W/cm2
- duration: t = 30 fs (in FWHM)
- beam waist in focus: $w_0 = 0.5$, 1.0, 2.0, 4.0 μm
- focus distance from boundary: $x_\mathrm{B} - x_0 = 8$ μm
- polarization: P
- boundary: left

Domain:

- x min: 0 μm
- x max: 15 μm
- y min: -20 μm
- y max: 20 μm
- N_x: 1875 cells ($\delta x = \lambda/100 = 8$ nm)
- N_y: 5000 cells ($\delta y = \lambda/100 = 8$ nm)
- time step: $\delta t = 1/(\sqrt{2}c)\lambda/100 \approx 0.05$ fs

- simulation time: τ = 200 fs

Target:

- x min: 8 μm
- x max: 10 μm
- y min: -15 μm
- y max: 15 μm
- electrons: 2000 ppc
- protons: 100 ppc
- density: 100 critical
- temperature: 100 eV

Seconds set (4 simulations with const. I = 1e21, 1000 ppc): Laser:

- wavelength: $\lambda = 0.8$ μm
- const intensity: I = 1e21 W/cm2
- duration: t = 30 fs (in FWHM)
- beam waist in focus: $w_0 = 0.5, 1.0, 2.0, 4.0$ μm
- focus distance from boundary: $x_{\mathrm{B}} - x_0 = 8$ μm
- polarization: P
- boundary: left

Domain:

- x min: 0 μm
- x max: 15 μm
- y min: -20 μm
- y max: 20 μm
- N_x: 1875 cells ($\delta x = \lambda/100 = 8$ nm)
- N_y: 5000 cells ($\delta y = \lambda/100 = 8$ nm)

- time step: $\delta t = 1/(\sqrt{2}c)\lambda/100 \approx 0.05$ fs
- simulation time: $\tau = 200$ fs

Target:

- x min: 8 μm
- x max: 10 μm
- y min: -15 μm
- y max: 15 μm
- electrons: 1000 ppc
- protons: 100 ppc
- density: 100 critical
- temperature: 100 eV

Third set (2 simulations with I = 1e21 W/cm2, thiner target (0.25 micron), 2000 ppc): Laser:

- wavelength: $\lambda = 0.8$ μm
- const intensity: I = 1e21 W/cm2
- duration: t = 30 fs (in FWHM)
- beam waist in focus: $w_0 = 0.5$, 2.0 μm
- focus distance from boundary: $x_\mathrm{B} - x_0 = 8$ μm
- polarization: P
- boundary: left

Domain:

- x min: 0 μm
- x max: 15 μm
- y min: -20 μm
- y max: 20 μm
- N_x: 1875 cells ($\delta x = \lambda/100 = 8$ nm)

- N_y: 5000 cells ($\delta y = \lambda/100 = 8$ nm)
- time step: $\delta t = 1/(\sqrt{2}c)\lambda/100 \approx 0.05$ fs
- simulation time: $\tau = 200$ fs

Target:

- x min: 8 μm
- x max: 8.25 μm
- y min: -15 μm
- y max: 15 μm
- electrons: 2000 ppc
- protons: 100 ppc
- density: 100 critical
- temperature: 100 eV

w_0 [μm]	Total absorption [%]		
	const. E = $2.83 \cdot 10^4$ J	const. I = 10^{20} W/cm^2	const. I = 10^{21} W/cm^2
0.5	20.12	16.38	34.41
1.0	9.61	9.61	25.21
2.0	5.27	8.26	24.98
4.0	3.50	8.29	24.38

Table 2: Total absorption, target thickness: 2.0 micron

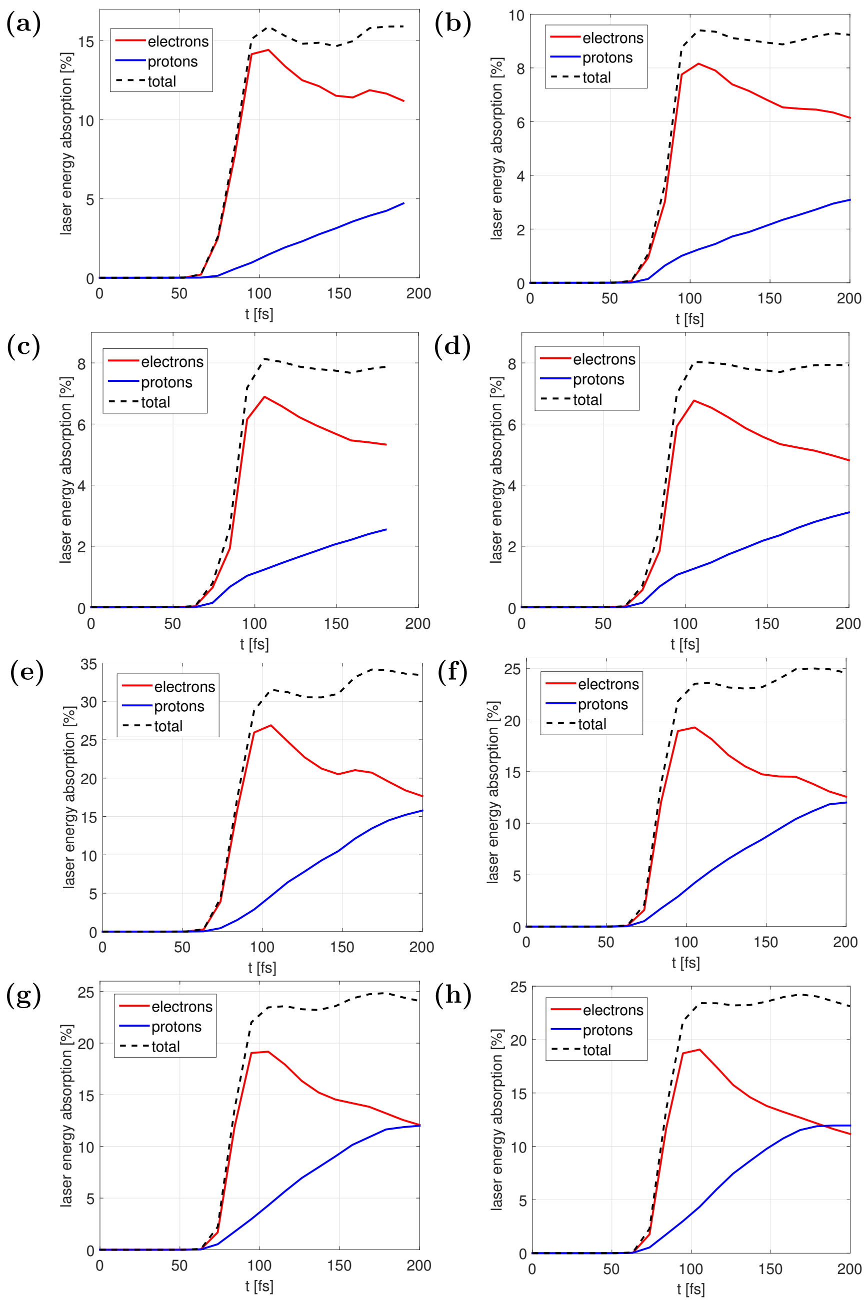

Figure 12: **(a)** w0 = 05 micron, I = 1e20 W/cm2 **(b)** w0 = 1 micron, I = 1e20 W/cm2 **(c)** w0 = 2 micron, I = 1e20 W/cm2 **(d)** w0 = 4 micron, I = 1e20 W/cm2 **(e)** w0 = 05 micron, I = 1e21 W/cm2 **(f)** w0 = 1 micron, I = 1e21 W/cm2 **(g)** w0 = 2 micron, I = 1e21 W/cm2 **(h)** w0 = 4 micron, I = 1e21 W/cm2

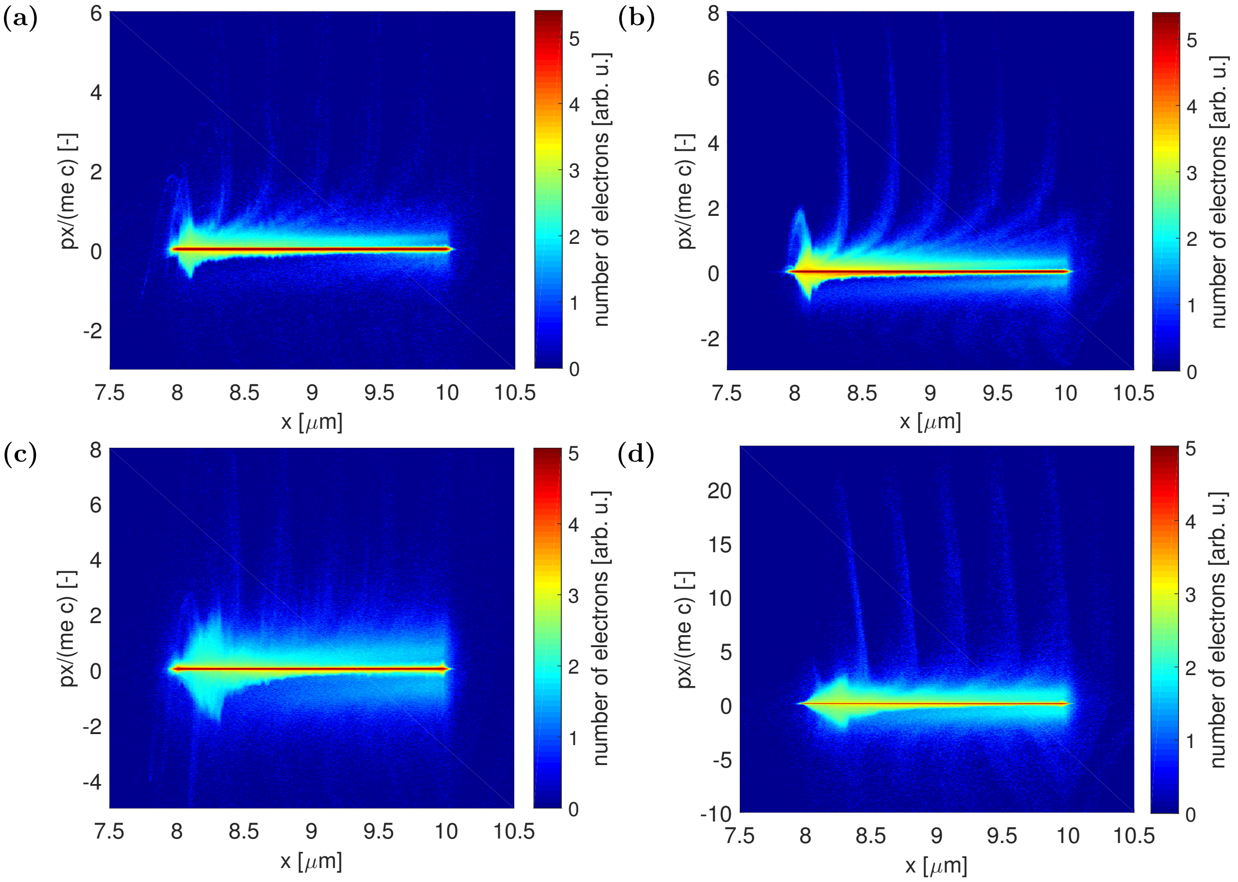

Figure 13: **(a)** w0 = 05 micron, I = 1e20 W/cm2, t = 100 fs **(b)** w0 = 2 micron, I = 1e20 W/cm2, t = 100 fs **(c)** w0 = 05 micron, I = 1e21 W/cm2, t = 100 fs **(d)** w0 = 2 micron, I = 1e21 W/cm2, t = 100 fs

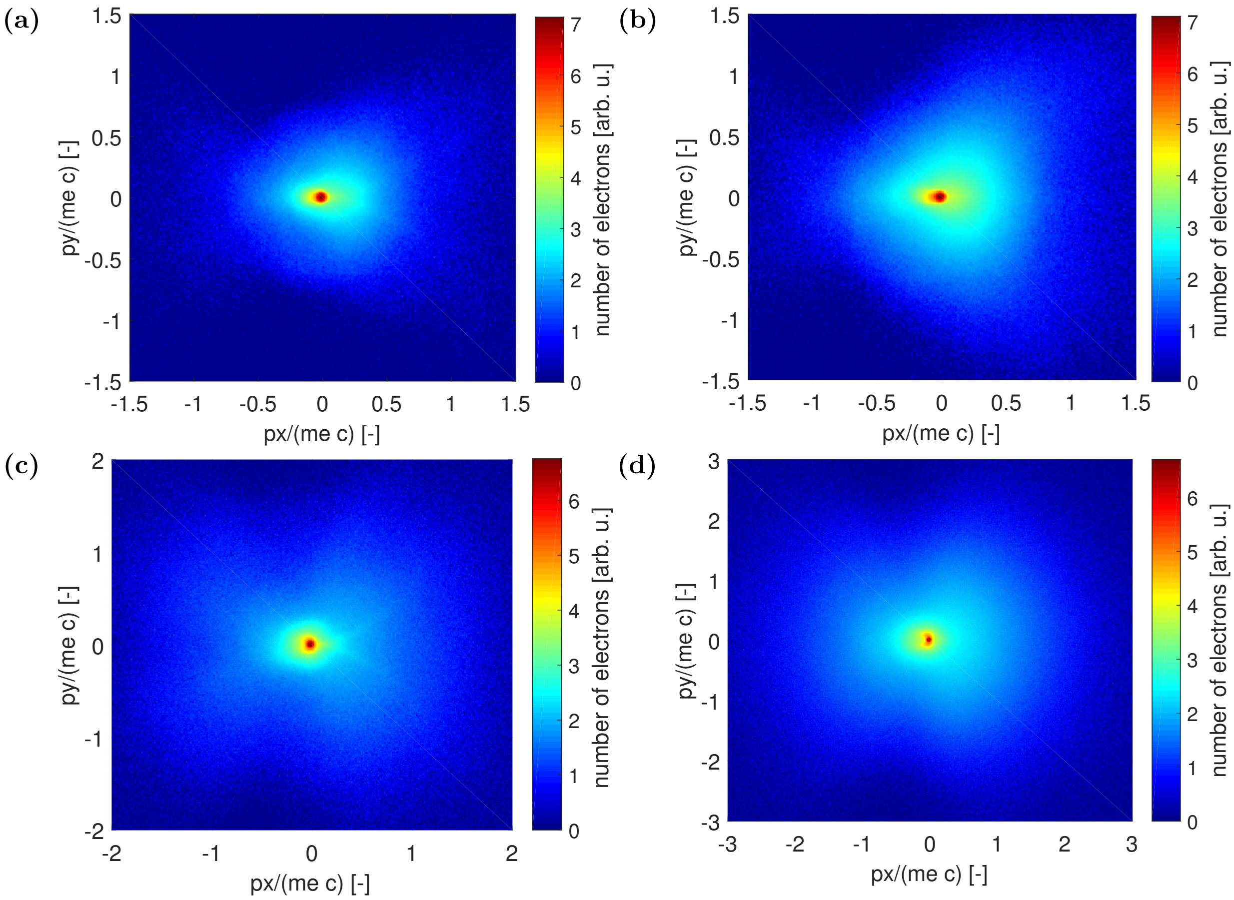

Figure 14: (a) w0 = 05 micron, I = 1e20 W/cm2, t = 100 fs **(b)** w0 = 2 micron, I = 1e20 W/cm2, t = 100 fs **(c)** w0 = 05 micron, I = 1e21 W/cm2, t = 100 fs **(d)** w0 = 2 micron, I = 1e21 W/cm2, t = 100 fs

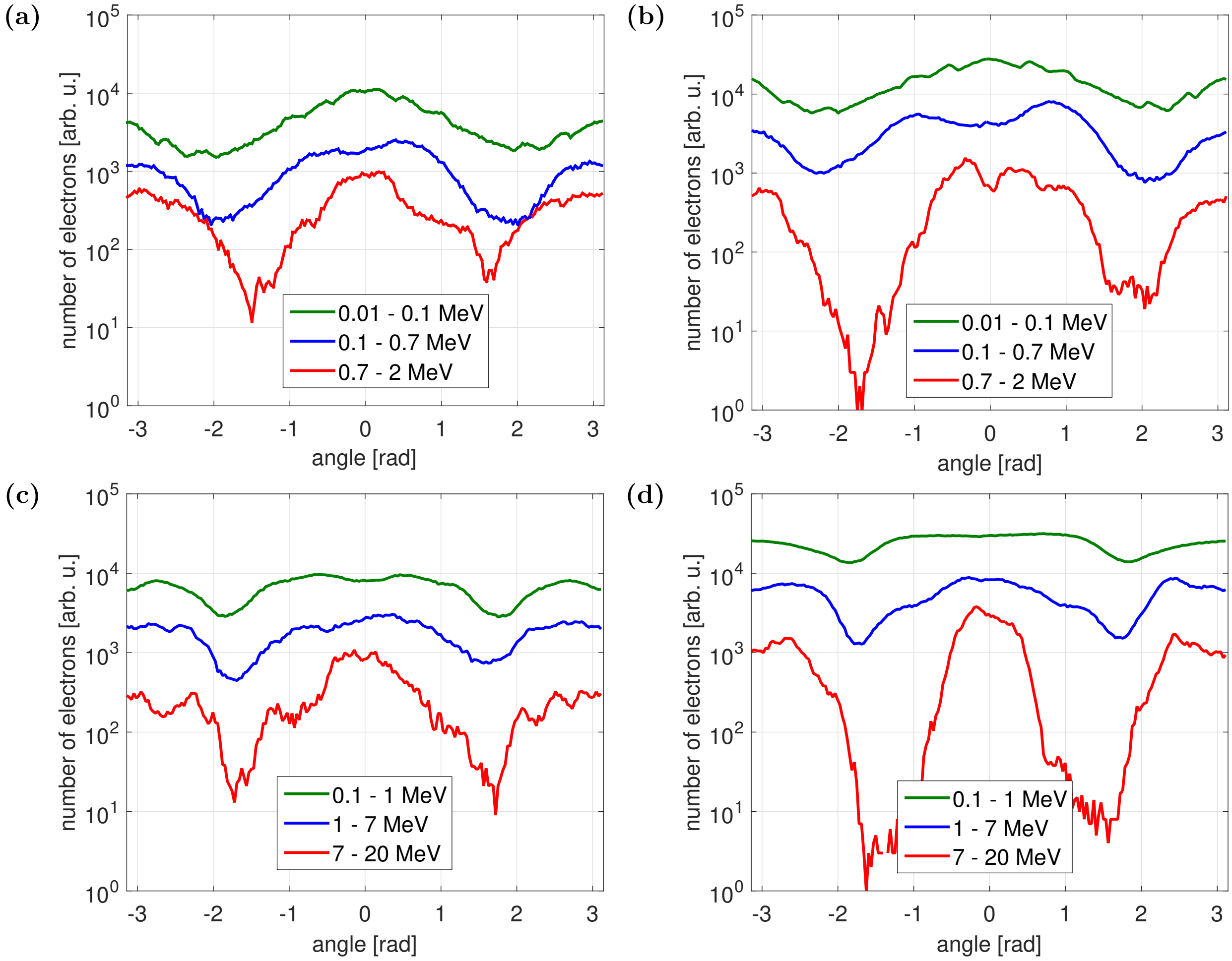
(a)
(b)
(c)
(d)
number of electrons [arb. u.]
angle [rad]
0.01 - 0.1 MeV
0.1 - 0.7 MeV
0.7 - 2 MeV
0.1 - 1 MeV
1 - 7 MeV
7 - 20 MeV

Figure 15

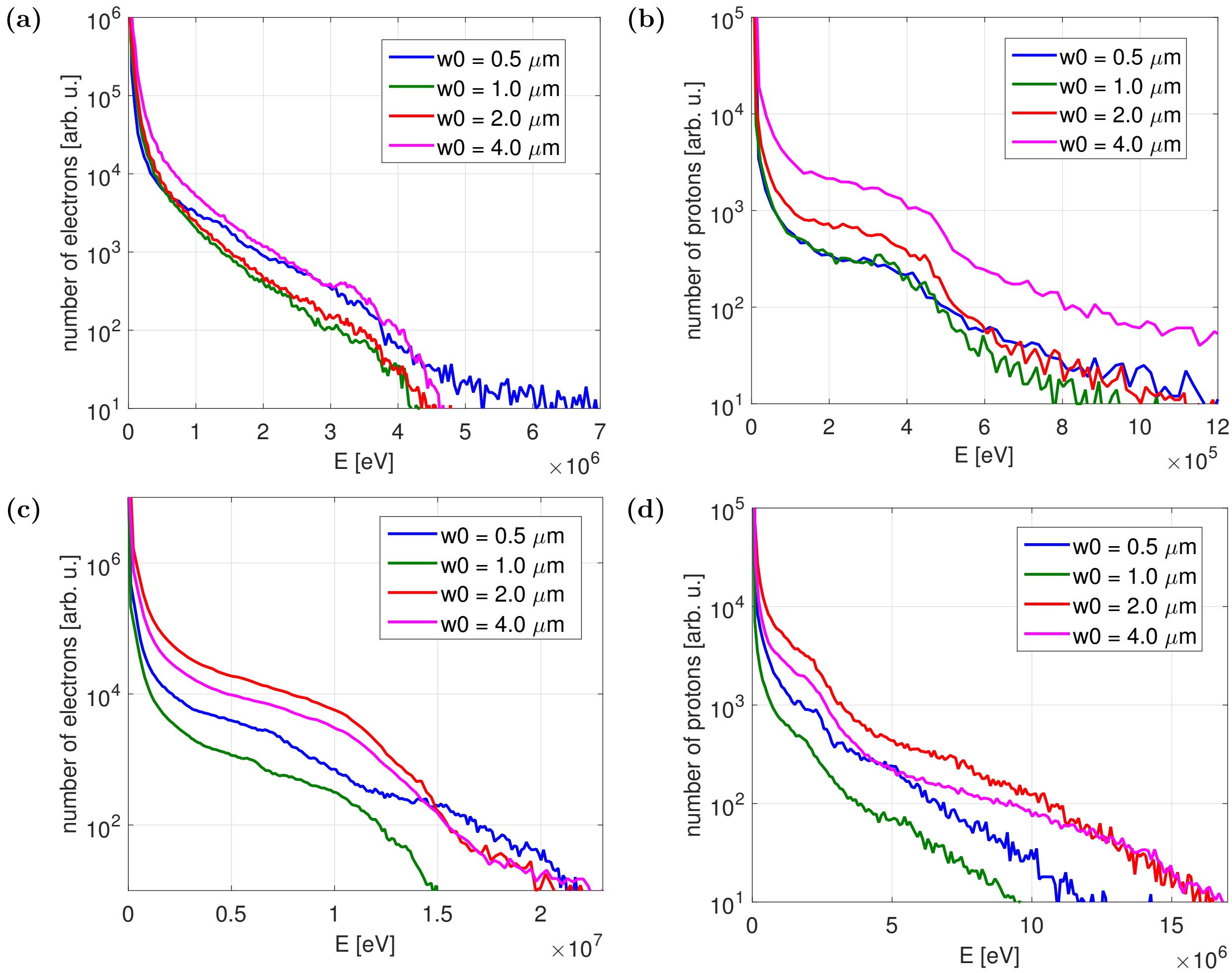
(a)
number of electrons [arb. u.]
E [eV]
$\times 10^6$
w0 = 0.5 μm
w0 = 1.0 μm
w0 = 2.0 μm
w0 = 4.0 μm
(b)
number of protons [arb. u.]
E [eV]
$\times 10^5$
w0 = 0.5 μm
w0 = 1.0 μm
w0 = 2.0 μm
w0 = 4.0 μm
(c)
number of electrons [arb. u.]
E [eV]
$\times 10^7$
w0 = 0.5 μm
w0 = 1.0 μm
w0 = 2.0 μm
w0 = 4.0 μm
(d)
number of protons [arb. u.]
E [eV]
$\times 10^6$
w0 = 0.5 μm
w0 = 1.0 μm
w0 = 2.0 μm
w0 = 4.0 μm

Figure 16

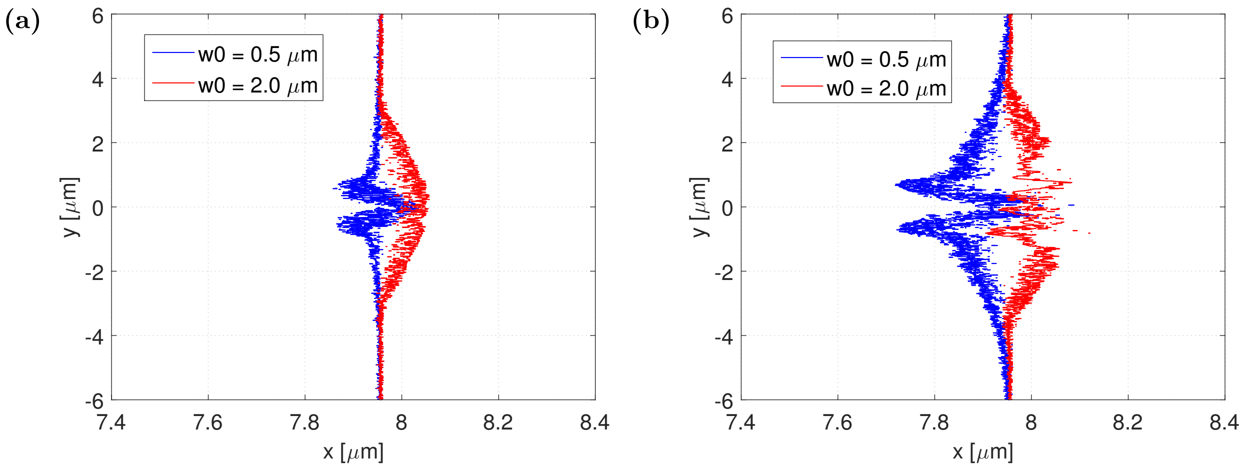

Figure 17: **(a)** nc protons, t = 100 fs, I = 1e20 W/cm2 **(b)** nc protons, t = 100 fs, I = 1e21 W/cm2

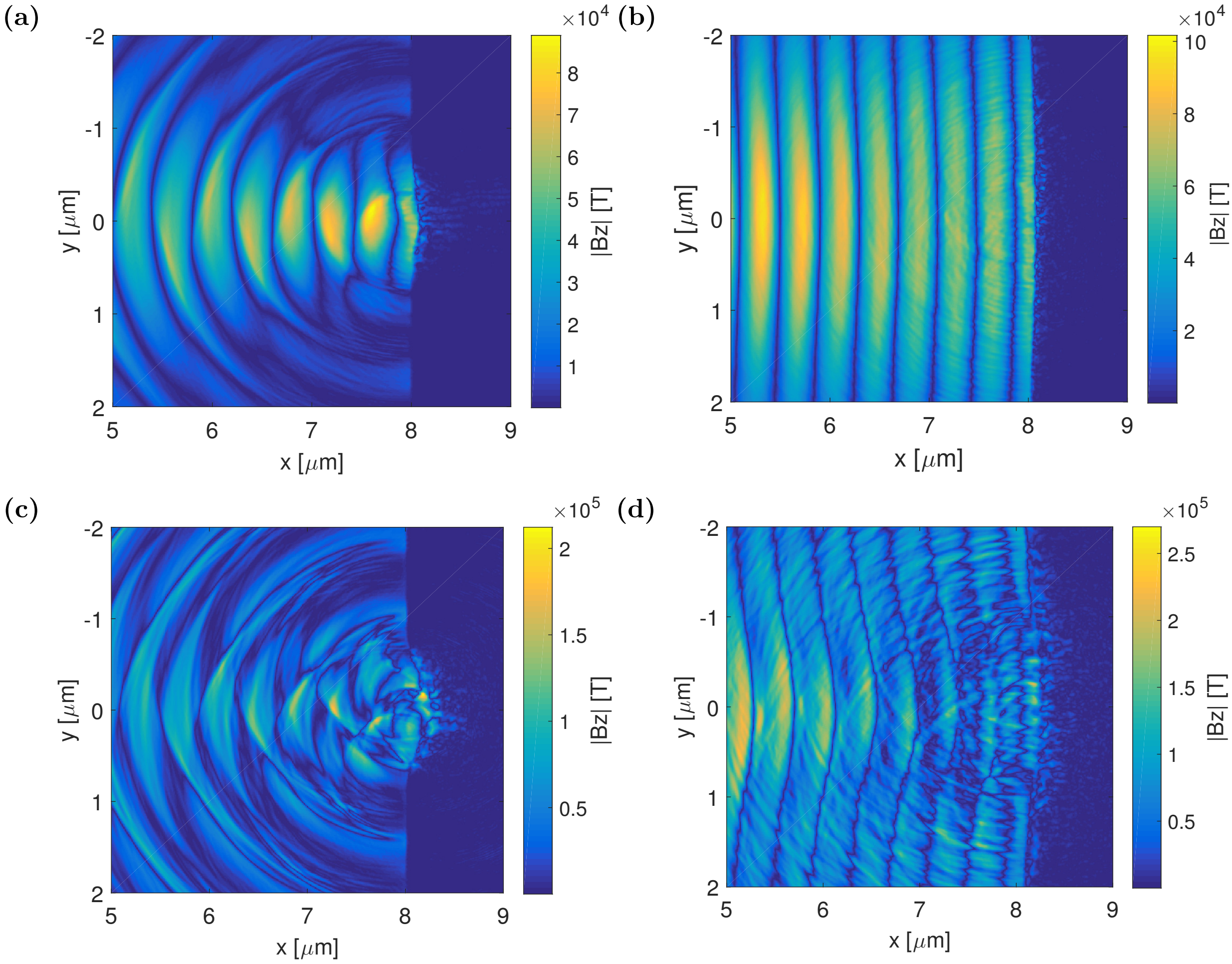

Figure 18

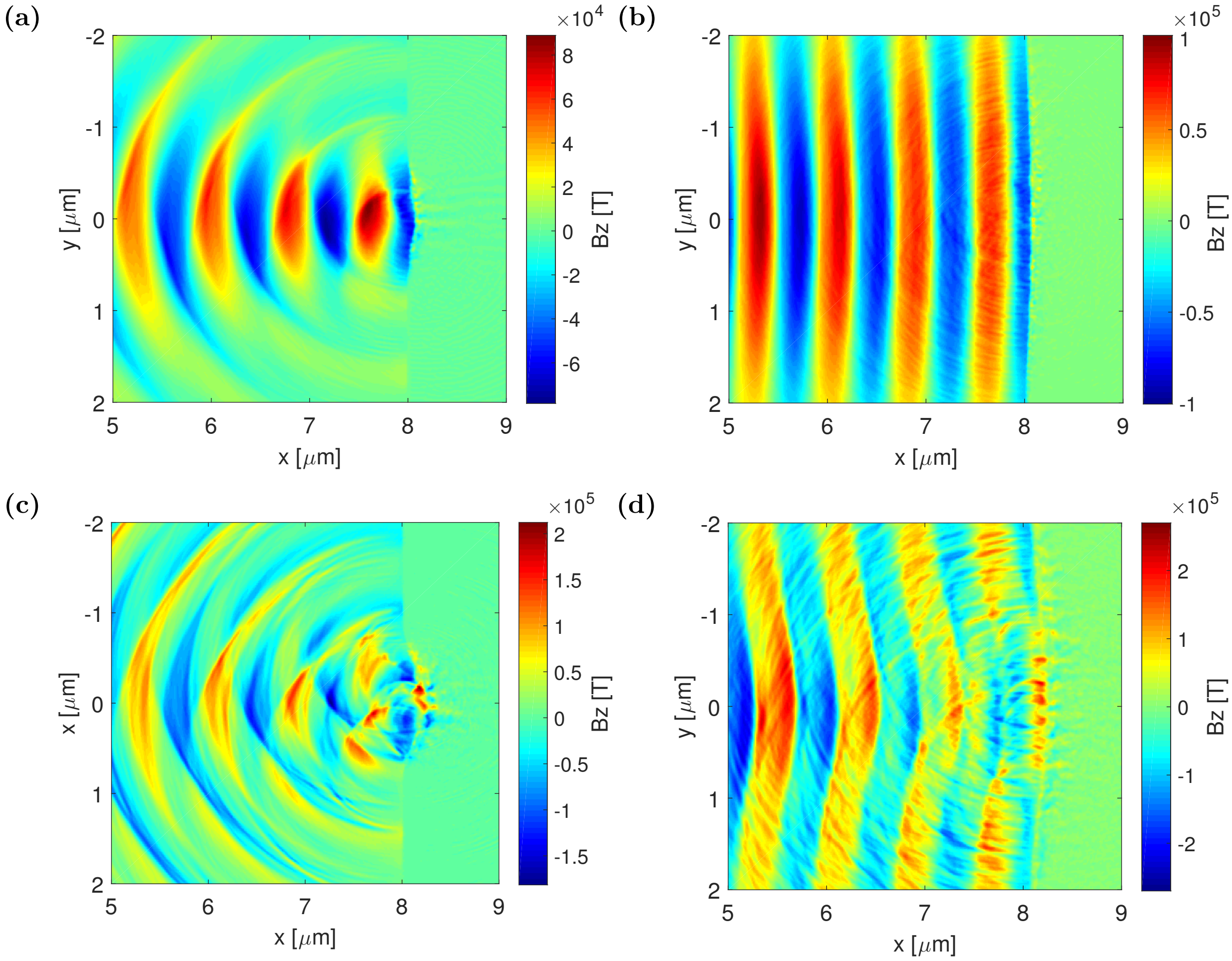

Figure 19

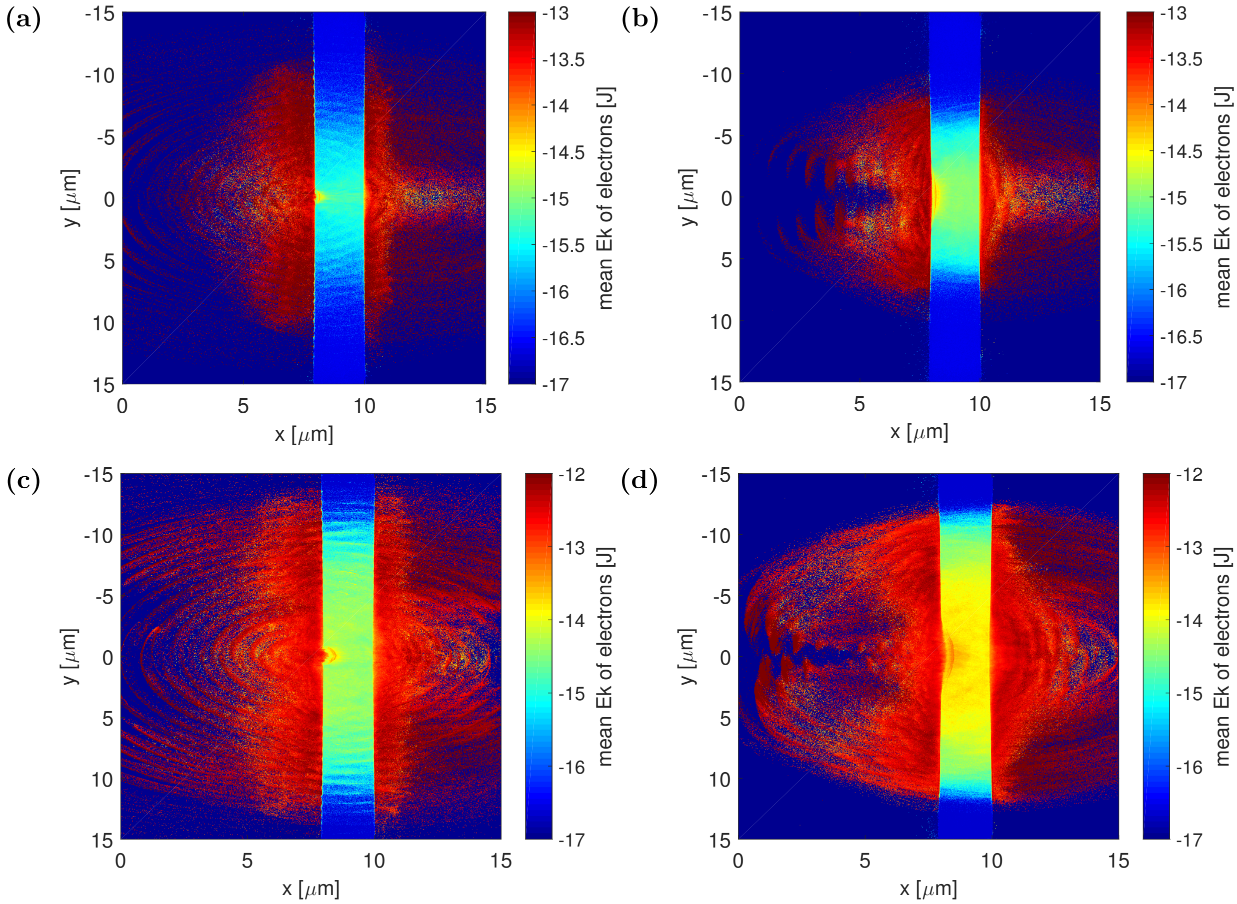

Figure 20

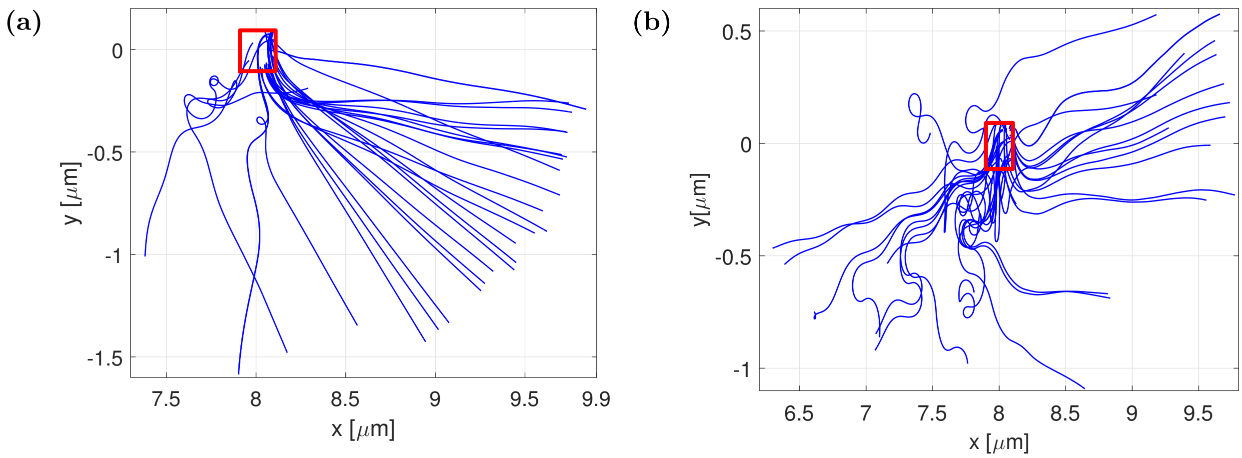

Figure 21: **(a)** electrons, w0 = 0.5 micron, t = 100 fs, I = 1e21 W/cm2 **(b)** electrons, wo = 2 micron, t = 100 fs, I = 1e21 W/cm2

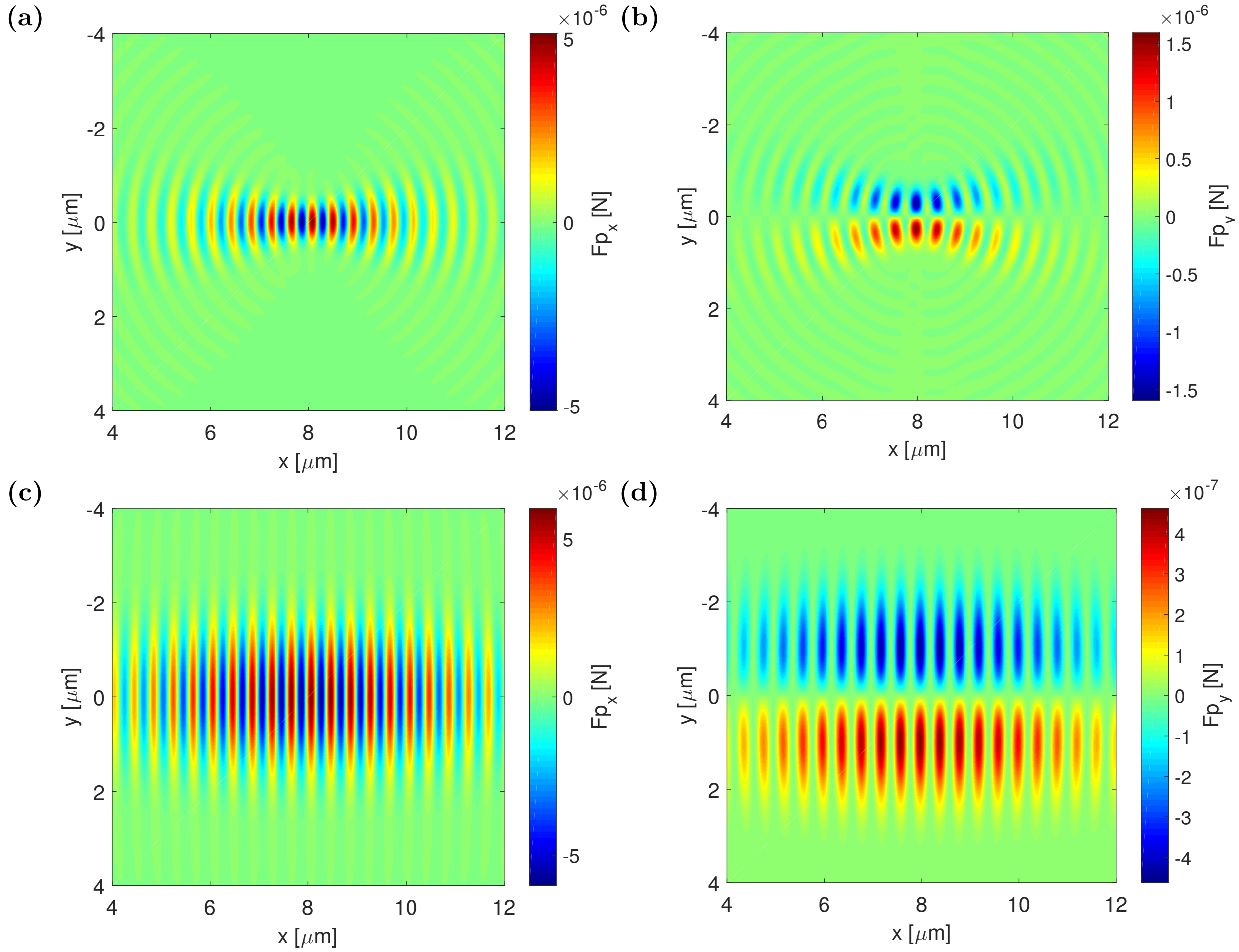

Figure 22: **(a)** 5.1312e-06 **(b)** 1.5913e-06 **(c)** 5.9528e-06 **(d)** 4.6193e-07

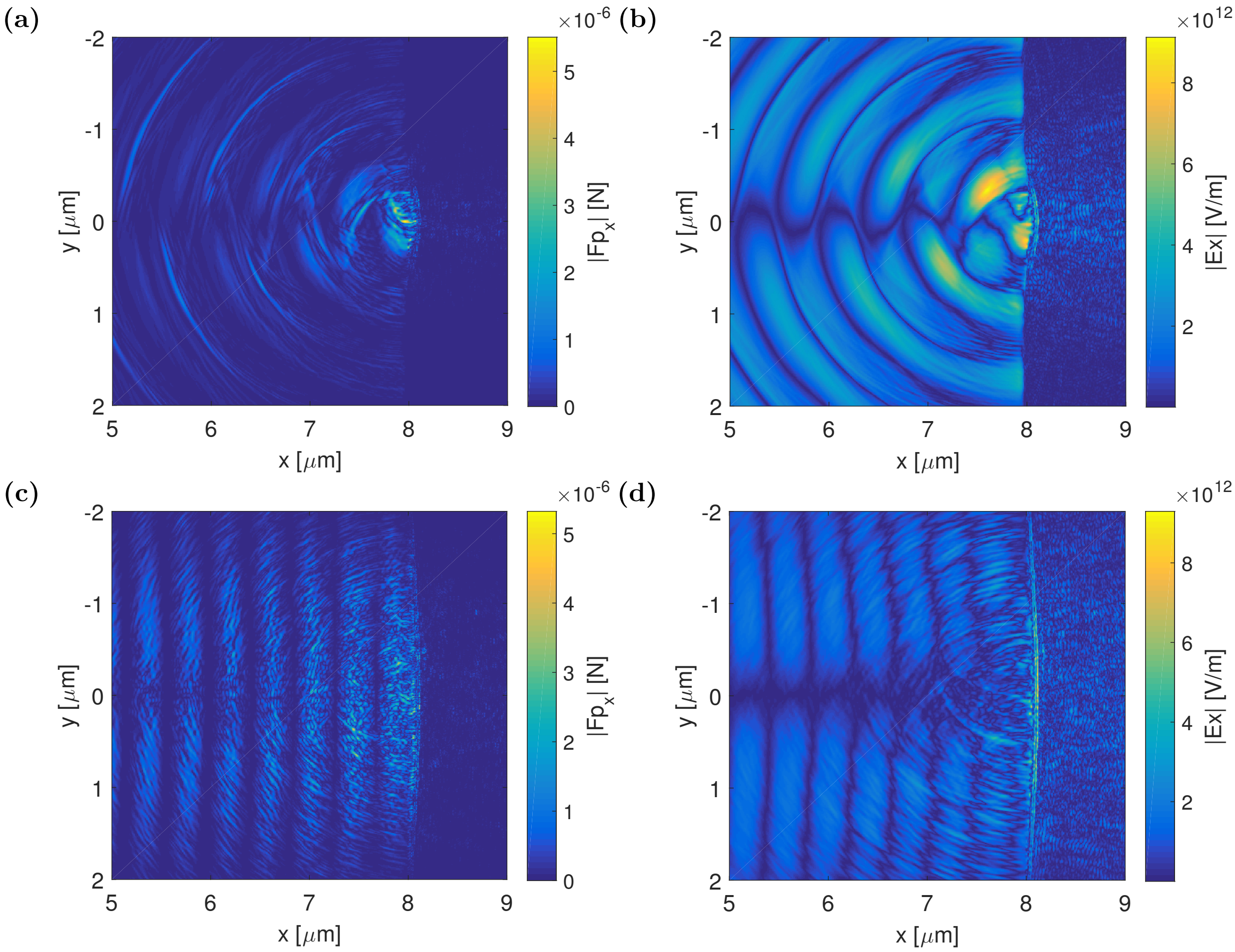

Figure 23

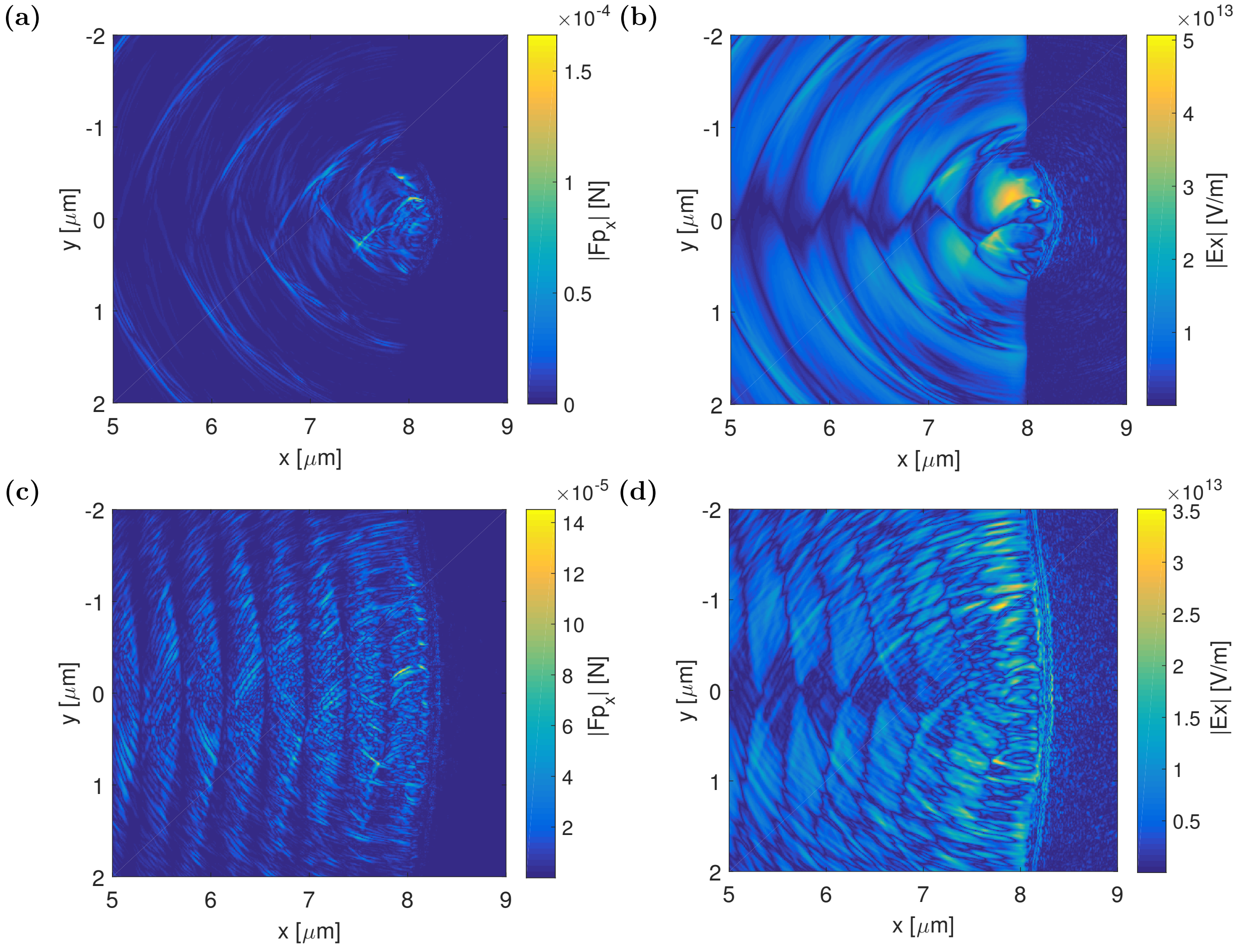

Figure 24

Conclusion

Acknowledgments

I wish express my gratitude to both, my supervisor doc. Ing. Ondřej Klimo, Ph.D. and consultant Dr. Stefan Andreas Weber for constant support and guidance, as well as for providing invaluable advice and direction.

Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum, provided under the programme "Projects of Large Infrastructure for Research, Development, and Innovations" (LM2010005), is greatly appreciated.

Access to the CERIT-SC computing and storage facilities provided under the programme Center CERIT Scientific Cloud, part of the Operational Program Research and Development for Innovations, reg. no. CZ. 1.05/3.2.00/08.0144, is greatly appreciated.

The development of the EPOCH code was funded in part by the UK EPSRC grants EP/G054950/1, EP/G056803/1, EP/G055165/1 and EP/ M022463/1.

Bc. Petr Valenta

Bibliography

[1] T D Arber, K Bennett, C S Brady, A Lawrence-Douglas, M G Ramsay, N J Sircombe, P Gillies, R G Evans, H Schmitz, A R Bell, and C P Ridgers. Contemporary particle-in-cell approach to laser-plasma modelling. *Plasma Physics and Controlled Fusion*, 57(11):113001, 2015.

[2] D. Batani, S. Baton, A. Casner, S. Depierreux, M. Hohenberger, O. Klimo, M. Koenig, C. Labaune, X. Ribeyre, C. Rousseaux, G. Schurtz, W. Theobald, and V.T. Tikhonchuk. Physics issues for shock ignition. *Nuclear Fusion*, 54(5):054009, 2014.

[3] R. Betti et al. Shock ignition: A new approach to high gain inertial confinement fusion on the national ignition facility. *Phys. Rev. Lett.*, 103(4), 2009.

[4] C. K. Birdsall and A. B. Langdon. *Plasma Physics via Computer Simulation.* Series in Plasma Physics. CRC Press, 2004.

[5] T. Zh. Esirkepov. Exact charge conservation scheme for particle-in-cell simulation with an arbitrary form-factor. *Computer Physics Communications*, 135(2), 2001.

[6] H. Fehske, R. Schneider, and A. Weiße. *Computational Many-Particle Physics.* Springer, 2008.

[7] Kai Germaschewski, William Fox, Narges Ahmadi, Liang Wang, Stephen Abbott, Hartmut Ruhl, and Amitava Bhattacharjee. The plasma simulation code: A modern particle-in-cell code with load-balancing and gpu support. *arXiv preprint arXiv:1310.7866*, 2013.

[8] H. Gould, J. Tobochnik, and W. Christian. *An introduction to computer simulation methods: applications to physical systems.* Addison–Wesley series in physics. Addison–Wesley, 3 edition, 2007.

[9] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles.* CRC Press, 2010.

[10] D. A. Jaroszynski, R. Bingham, and R. A. Cairns. *Laser-Plasma Interactions.* Scottish Graduate Series. CRC Press, 2009.

[11] G. Lapenta. Particle in cell methods with application to simulations in space weather. Lecture notes.

[12] T. Pang. *An introduction to computational physics*. Cambridge University Press, 2 edition, 2006.

[13] V T Tikhonchuk, A Colaïtis, A Vallet, E Llor Aisa, G Duchateau, Ph Nicolaï, and X Ribeyre. Physics of laser-plasma interaction for shock ignition of fusion reactions. *Plasma Physics and Controlled Fusion*, 58(1):014018, 2016.

[14] J. Villasenor and O. Buneman. Rigorous charge conservation for local electromagnetic field solvers. *Computer Physics Communications*, 69(2-3), 1992.

[15] K. Yee. Numerical solution of inital boundary value problems involving maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14(3), 1966.

Appendices

Appendix A

Input files

This part contains input files of several most important simulations that have been performed for the purposes within this work.

Listing A.1: test lbcs

```
begin:control
   nx = 1600
   ny = 4800
   x_min = -8 * micron
   x_max = 8 * micron
   y_min = -24 * micron
   y_max = 24 * micron
   t_end = 150 * femto
   field_order = 2
   stdout_frequency = 1
   smooth_currents = T
   dt_multiplier = 0.95
end:control

begin:boundaries
   cpml_thickness = 16
   cpml_kappa_max = 20
   cpml_a_max = 0.2
   cpml_sigma_max = 0.7
   bc_x_min_field = cpml_laser
   bc_x_max_field = cpml_outflow
   bc_x_min_particle = thermal
   bc_x_max_particle = thermal
   bc_y_min_field = cpml_outflow
   bc_y_max_field = cpml_outflow
   bc_y_min_particle = thermal
   bc_y_max_particle = thermal
end:boundaries

begin:laser
   boundary = x_min
   id = 1
   fwhm_time = 20 * femto
   t_0 = 50 * femto
   w_0 = 0.7 * micron # 5.0 * micron
   pos = 0.0 * micron
   focus = 0.0 * micron
```

```
   amp = 1.0e15
   lambda = 1.0 * micron
   pol_angle = 0.0 * pi
   t_start = 0.0 * femto
   t_end = 300 * femto
end:laser

begin:output
   name = fields
   file_prefix = field
   dump_at_times = 50.00 * femto, 76.685127616 * femto, 103.370255232 * femto
   grid = always + single
   ex = always + single
   ey = always + single
   bz = always + single
   dump_first = F
   dump_last = F
   restartable = F
end:output
```

Listing A.2: test paraxial

```
begin:control
   nx = 1600
   ny = 4800
   x_min = -8 * micron
   x_max = 8 * micron
   y_min = -24 * micron
   y_max = 24 * micron
   t_end = 150 * femto
   field_order = 2
   stdout_frequency = 1
   smooth_currents = T
   dt_multiplier = 0.95
end:control

begin:boundaries
   cpml_thickness = 16
   cpml_kappa_max = 20
   cpml_a_max = 0.2
   cpml_sigma_max = 0.7
   bc_x_min_field = cpml_laser
   bc_x_max_field = cpml_outflow
   bc_x_min_particle = thermal
   bc_x_max_particle = thermal
   bc_y_min_field = cpml_outflow
   bc_y_max_field = cpml_outflow
   bc_y_min_particle = thermal
   bc_y_max_particle = thermal
end:boundaries

begin:constant
   cell_xsize = (x_max - x_min) / nx
   cell_ysize = (y_max - y_min) / ny
   las_lambda0 = 1.0 * micron
   las_omega = 2.0 * pi * c / las_lambda0
   las_time = 2.0 * pi / las_omega
   theta = 0
   w0 = 0.7 * micron # 5.0 * micron
   fwhm_time = 20 * femto
   w_time = fwhm_time / (2.0 * (sqrt(loge(2.0))))
```

```
    xfocus = 8 * micron
    yc = -xfocus * tan(theta)
    zR = pi * w0^2 / las_lambda0
    lfocus = xfocus / cos(theta)
    wl = w0 * sqrt(1.0+(lfocus^2)/(zR^2))
    w_y = wl / cos(theta)
    r_curv = lfocus * (1.0+(zR^2)/(lfocus^2))
    intensity_fac2d = 1.0 / sqrt(1.0+lfocus^2/zR^2) * cos(theta)
    n_crit = critical(las_omega)
end:constant

begin:laser
    boundary = x_min
    amp = 1.8895615869e14
    lambda = 1.0 * micron
    pol_angle = 0.0 * pi
    phase = -2.0*pi*(y-yc)*sin(theta)/las_lambda0 + pi/(las_lambda0)*((y-yc)*cos(theta))^2*1.0/
      r_curv - atan(((yc-y)*sin(theta)+lfocus)/zR)
    profile = gauss(y, yc, sqrt(2)*w_y)
    t_profile = gauss(time, 50*femto, w_time)
    t_start = 0.0 * femto
    t_end = 300 * femto
end:laser

begin:output
    name = fields
    file_prefix = field
    dump_at_times = 50.00 * femto, 76.685127616 * femto, 103.370255232 * femto
    grid = always + single
    ex = always + single
    ey = always + single
    bz = always + single
    dump_first = F
    dump_last = F
    restartable = F
end:output
```

Listing A.3: 2kppc

```
begin:control
    nx = 1875
    ny = 5000
    x_min = 0 * micron
    x_max = 15 * micron
    y_min = -20 * micron
    y_max = 20 * micron
    t_end = 200 * femto
    field_order = 2
    stdout_frequency = 1
    smooth_currents = T
    dt_multiplier = 0.95
end:control

begin:boundaries
    cpml_thickness = 16
    cpml_kappa_max = 20
    cpml_a_max = 0.2
    cpml_sigma_max = 0.7
    bc_x_min_field = cpml_laser
    bc_x_max_field = cpml_outflow
    bc_x_min_particle = thermal
```

```
  bc_x_max_particle = thermal
  bc_y_min_field = cpml_outflow
  bc_y_max_field = cpml_outflow
  bc_y_min_particle = thermal
  bc_y_max_particle = thermal
end:boundaries

begin:species
  name = electron
  charge = -1.0
  mass = 1.0
  npart_per_cell = 2000
  density = if((x gt 8 * micron) and (x lt 10 * micron) and (abs(y) lt 15 * micron), 100.0 *
   n_crit, 0.0)
  temp_ev = 100
end:species

begin:species
  name = proton
  charge = 1.0
  mass = 1836.2
  npart_per_cell = 100
  density = density(electron)
  temp_ev = 100
end:species

begin:laser
  boundary = x_min
  id = 1
  fwhm_time = 30 * femto
  t_0 = 60 * femto
  w_0 = 0.5 * micron
  pos = 0.0 * micron
  focus = 8.0 * micron
  intensity_w_cm2 = 1.0e20
  lambda = 0.8 * micron
  pol_angle = 0.0 * pi
  t_start = 0.0 * femto
  t_end = 300 * femto
end:laser

begin:subset
  name = tenth
  random_fraction = 0.1
  include_species:electron
  include_species:proton
end:subset

begin:output
  name = particles
  file_prefix = part
  dt_snapshot = 10 * femto
  particle_grid = tenth + single
  px = tenth + single
  py = tenth + single
  pz = tenth + single
  particle_weight = tenth + single
  dump_first = F
  dump_last = F
  restartable = F
end:output

```

```
begin:output
   name = fields
   file_prefix = field
   dt_snapshot = 10 * femto
   grid = always + single
   ex = always + single
   ey = always + single
   bz = always + single
   dump_first = F
   dump_last = F
   restartable = F
end:output

begin:output
   name = density
   file_prefix = dens
   dt_snapshot = 10 * femto
   mass_density = always + single + species
   charge_density = always + single + species
   number_density = always + single + species
   dump_first = F
   dump_last = F
   restartable = F
end:output

begin:output
   name = diag
   file_prefix = diag
   dt_snapshot = 10 * femto
   ekbar = always + single + species
   ekflux = always + single + species
   poynt_flux = always + single
   temperature = always + single + species
   total_energy_sum = always
   dump_first = F
   dump_last = F
   restartable = F
end:output

begin:output
   name = restart
   file_prefix = r
   dt_snapshot = 50.0 * femto
   dump_first = F
   dump_last = F
   restartable = T
end:output
```

Appendix B

Code listings

Below, one can find the most important functions and methods that has been created within this work to provide some new functionalities and features into the code EPOCH. These serve mainly for the computations of laser fields at boundaries that are consistent with the Maxwell's equations using discrete Fourier transforms. Also, one can find the methods for the data manipulation and routines that interface corresponding C++ functions into the FORTRAN simulation code. Finally, the C++ and FORTRAN adaptors for ParaView Catalyst that enable in-situ visualization and diagnostics with a sample visualization Python script pipeline are all attached.

The following part is provided as it is, only with a short captions. It is mainly intended for those who are interested in the way of implementation and do not want to browse in the full source code, which can be found on the attached CD. Closer details are discussed in the third and fourth chapter of this work.

Listing B.1: Function performing forward fast Fourier transform using Intel® MKL library

```
std::vector<std::complex<double>> fft::mkl_fft_forward(std::vector<std::complex<double>> in) {
DFTI_DESCRIPTOR_HANDLE desc;
MKL_LONG status;
DftiCreateDescriptor(&desc, DFTI_DOUBLE, DFTI_COMPLEX, 1, static_cast<MKL_LONG>(in.size()));
DftiCommitDescriptor(desc);
status = DftiComputeForward(desc, in.data());
if(status != 0) {
std::cerr << DftiErrorMessage(status) << std::endl;
abort();
}
DftiFreeDescriptor(&desc);
return in;
}
```

Listing B.2: Function performing backward fast Fourier transform using Intel® MKL library

```
std::vector<std::complex<double>> fft::mkl_fft_backward(std::vector<std::complex<double>> in) {
DFTI_DESCRIPTOR_HANDLE desc;
MKL_LONG status;
DftiCreateDescriptor(&desc, DFTI_DOUBLE, DFTI_COMPLEX, 1, static_cast<MKL_LONG>(in.size()));
DftiCommitDescriptor(desc);
```

```
status = DftiComputeBackward(desc, in.data());
if(status != 0) {
std::cerr << DftiErrorMessage(status) << std::endl;
abort();
}
DftiFreeDescriptor(&desc);
return in;
}
```

Listing B.3: Function performing forward fast Fourier transform using FFTW library

```
std::vector<std::complex<double>> fft::fftw_fft_forward(std::vector<std::complex<double>> in) {
fftw_plan p = fftw_plan_dft_1d(in.size(), reinterpret_cast<fftw_complex*>(in.data()),
    reinterpret_cast<fftw_complex*>(in.data()), FFTW_FORWARD, FFTW_ESTIMATE);
fftw_execute(p);
fftw_destroy_plan(p);
return in;
}
```

Listing B.4: Function performing backward fast Fourier transform using FFTW library

```
std::vector<std::complex<double>> fft::fftw_fft_backward(std::vector<std::complex<double>> in) {
fftw_plan p = fftw_plan_dft_1d(in.size(), reinterpret_cast<fftw_complex*>(in.data()),
    reinterpret_cast<fftw_complex*>(in.data()), FFTW_BACKWARD, FFTW_ESTIMATE);
fftw_execute(p);
fftw_destroy_plan(p);
return in;
}
```

Listing B.5: Function performing forward discrete Fourier transform without using any library

```
std::vector<std::complex<double>> fft::fft_forward(std::vector<std::complex<double>> in) {
std::vector<std::complex<double>> out(in.size());
for(auto j = 0; j < out.size(); j++) {
for(auto l = 0; l < out.size(); l++) {
out.at(j) += in.at(l) * exp(-2.0 * constants::pi * I * l * j / in.size());
}
}
return out;
}
```

Listing B.6: Function performing backward discrete Fourier transform without using any library

```
std::vector<std::complex<double>> fft::fft_backward(std::vector<std::complex<double>> in) {
std::vector<std::complex<double>> out(in.size());
for(auto j = 0; j < out.size(); j++) {
for(auto l = 0; l < out.size(); l++) {
out.at(j) += in.at(l) * exp(+2.0 * constants::pi * I * l * j / in.size());
}
}
return out;
}
```

Listing B.7: Method for performing discrete Fourier transform in time

```
void laser_bcs::dft_time(field_2d<std::complex<double>>& field) const {
#ifdef OPENMP
#pragma omp parallel for schedule(static)
```

```
#endif
for(auto j = 0; j < this->domain->Nx; j++) {
#ifdef USE_MKL
field.add_col(fft::mkl_fft_backward(field.get_col(j)), j);
#elif USE_FFTW
field.add_col(fft::fftw_fft_backward(field.get_col(j)), j);
#else
field.add_col(fft::fft_backward(field.get_col(j)), j);
#endif
}
field.multiply(this->domain->dt / (2.0 * constants::pi));
return;
}
```

Listing B.8: Method for performing inverse discrete Fourier transform in time

```
void laser_bcs::idft_time(field_2d<std::complex<double>>& field) const {
#ifdef OPENMP
#pragma omp parallel for schedule(static)
#endif
for(auto j = 0; j < this->domain->Nx; j++) {
#ifdef USE_MKL
field.add_col(fft::mkl_fft_forward(field.get_col(j)), j);
#elif USE_FFTW
field.add_col(fft::fftw_fft_forward(field.get_col(j)), j);
#else
field.add_col(fft::fft_forward(field.get_col(j)), j);
#endif
}
field.multiply(2.0 * (2.0 * constants::pi) / (this->domain->Nt * this->domain->dt));
return;
}
```

Listing B.9: Method for performing discrete Fourier transform in space

```
void laser_bcs::dft_space(field_2d<std::complex<double>>& field) const {
std::vector<std::complex<double>> row_global(this->domain->Nx_global);
std::vector<std::complex<double>> row_local;
for(auto j = 0; j < this->domain->Nt; j++) {
row_local = field.get_row(j);
MPI_Gatherv(row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX, row_global.data(), this->
    domain->counts.data(), this->domain->displs.data(), MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD
    );
if(this->domain->rank == 0) {
#ifdef USE_MKL
row_global = fft::mkl_fft_forward(row_global);
#elif USE_FFTW
row_global = fft::fftw_fft_forward(row_global);
#else
row_global = fft::fft_forward(row_global);
#endif
}
MPI_Scatterv(row_global.data(), this->domain->counts.data(), this->domain->displs.data(),
    MPI_CXX_DOUBLE_COMPLEX, row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX, 0,
    MPI_COMM_WORLD);
field.add_row(row_local, j);
}
field.multiply(this->domain->dx / (2.0 * constants::pi));
return;
}
```

Listing B.10: Method for performing inverse discrete Fourier transform in space

```
void laser_bcs::idft_space(field_2d<std::complex<double>>& field) const {
std::vector<std::complex<double>> row_global(this->domain->Nx_global);
std::vector<std::complex<double>> row_local;
for(auto j = 0; j < this->domain->Nt; j++) {
row_local = field.get_row(j);
MPI_Gatherv(row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX, row_global.data(), this->
    domain->counts.data(), this->domain->displs.data(), MPI_CXX_DOUBLE_COMPLEX, 0, MPI_COMM_WORLD
    );
if(this->domain->rank == 0) {
#ifdef USE_MKL
row_global = fft::mkl_fft_backward(row_global);
#elif USE_FFTW
row_global = fft::fftw_fft_backward(row_global);
#else
row_global = fft::fft_backward(row_global);
#endif
}
MPI_Scatterv(row_global.data(), this->domain->counts.data(), this->domain->displs.data(),
    MPI_CXX_DOUBLE_COMPLEX, row_local.data(), this->domain->Nx, MPI_CXX_DOUBLE_COMPLEX, 0,
    MPI_COMM_WORLD);
field.add_row(row_local, j);
}
field.multiply((2.0 * constants::pi) / (this->domain->Nx_global * this->domain->dx));
return;
}
```

Listing B.11: Method for dumping data into shared file

```
template <typename T>
void field_2d<T>::dump_to_shared_file(std::string name, int row_first, int row_last, int
    row_size_local, int row_size_global, int col_start) const {
MPI_File file;
MPI_Offset offset = 0;
MPI_Status status;
MPI_Datatype local_array;
int col_size = row_last - row_first;
const int ndims = 2;
std::array<int, ndims> size_global = {col_size, row_size_global};
std::array<int, ndims> size_local = {col_size, row_size_local};
std::array<int, ndims> start_coords = {0, col_start};
MPI_Type_create_subarray(2, size_global.data(), size_local.data(), start_coords.data(),
    MPI_ORDER_C, MPI_DOUBLE, &local_array);
MPI_Type_commit(&local_array);
std::vector<double> real_part(col_size * row_size_local);
for(auto i = std::make_pair(row_first, 0); i.first < row_last; i.first++, i.second++) {
for(auto j = 0; j < row_size_local; j++) {
real_part[i.second * row_size_local + j] = std::real(this->data[i.first * row_size_local + j]);
}
}
MPI_File_open(MPI_COMM_WORLD, name.data(), MPI_MODE_CREATE|MPI_MODE_WRONLY, MPI_INFO_NULL, &file)
    ;
MPI_File_set_view(file, offset, MPI_DOUBLE, local_array, "native", MPI_INFO_NULL);
MPI_File_write_all(file, real_part.data(), col_size * row_size_local, MPI_DOUBLE, &status);
MPI_File_close(&file);
MPI_Type_free(&local_array);
return;
}
```

Listing B.12: Extern C++ function to fill Fortran arrays with laser fields dumped in binary file

```
void populate_laser_at_boundary(double* field, int* id, const char* data_dir, int* timestep, int* size_global, int* first, int* last) {
double num = 0.0;
std::string laser_id = std::to_string(*id);
std::string path(data_dir);
std::ifstream in;
in.open(path + "/laser_" + laser_id + ".dat", std::ios::binary);
if(in.is_open()) {
in.seekg(((*timestep) * (*size_global) + (*first) - 1) * sizeof(num));
for(auto i = 0; i < *last - *first + 1; i++) {
in.read(reinterpret_cast<char*>(&num), sizeof(num));
field[i] = num;
}
in.close();
} else {
std::cout << "error: cannot read file " << path + "/" + filename + laser_id + ".dat" << std::endl;
}
return;
}
```

Listing B.13: Fortran interfaces for C++ library functions

```
INTERFACE

SUBROUTINE compute_laser_at_boundary(rank, nproc, laser_start, laser_end, &
fwhm_time, t_0, omega, pos, amp, w_0, id, L_min, L_max, L_focus, T_min, T_max, &
T_ncells, cpml_thickness, t_end, T_cell_size, L_cell_size, dt, output_path) bind(c)
USE, INTRINSIC :: iso_c_binding
IMPLICIT NONE
INTEGER(c_int), INTENT(IN) :: rank, nproc, id, T_ncells, cpml_thickness
CHARACTER(kind=c_char), DIMENSION(*), INTENT(IN) :: output_path
REAL(c_double), INTENT(IN) :: laser_start, laser_end, fwhm_time, t_0, omega, pos, &
amp, w_0, L_min, L_max, L_focus, T_min, T_max, t_end, T_cell_size, L_cell_size, dt
END SUBROUTINE compute_laser_at_boundary

SUBROUTINE populate_laser_at_boundary(field, laser_id, output_path, timestep, size_global, first, last) bind(c)
USE, INTRINSIC :: iso_c_binding
IMPLICIT NONE
INTEGER(c_int), INTENT(IN) :: laser_id, timestep, size_global, first, last
CHARACTER(kind=c_char), DIMENSION(*), INTENT(IN) :: output_path
REAL(c_double), DIMENSION(*), INTENT(OUT) :: field
END SUBROUTINE populate_laser_at_boundary

END INTERFACE
```

Listing B.14: Fortran subroutines for Maxwell consistent computation of laser fields at boundaries

```
SUBROUTINE Maxwell_consistent_computation_of_EM_fields

TYPE(laser_block), POINTER :: current

current => laser_x_min
DO WHILE(ASSOCIATED(current))
CALL compute_laser_at_boundary(rank, nproc, current%t_start, current%t_end, &
current%fwhm_time, current%t_0, current%omega, current%pos, current%amp, current%w_0, &
current%id, x_min, x_max, current%focus, y_min, y_max, ny_global, cpml_thickness, t_end, &
dy, dx, dt, TRIM(data_dir)//C_NULL_CHAR)
```

```
current => current%next
ENDDO

current => laser_x_max
DO WHILE(ASSOCIATED(current))
CALL compute_laser_at_boundary(rank, nproc, current%t_start, current%t_end, &
current%fwhm_time, current%t_0, current%omega, current%pos, current%amp, current%w_0, &
current%id, x_min, x_max, current%focus, y_min, y_max, ny_global, cpml_thickness, t_end, &
dy, dx, dt, TRIM(data_dir)//C_NULL_CHAR)
current => current%next
ENDDO

current => laser_y_min
DO WHILE(ASSOCIATED(current))
CALL compute_laser_at_boundary(rank, nproc, current%t_start, current%t_end, &
current%fwhm_time, current%t_0, current%omega, current%pos, current%amp, current%w_0, &
current%id, y_min, y_max, current%focus, x_min, x_max, nx_global, cpml_thickness, t_end, &
dx, dy, dt, TRIM(data_dir)//C_NULL_CHAR)
current => current%next
ENDDO

current => laser_y_max
DO WHILE(ASSOCIATED(current))
CALL compute_laser_at_boundary(rank, nproc, current%t_start, current%t_end, &
current%fwhm_time, current%t_0, current%omega, current%pos, current%amp, current%w_0, &
current%id, y_min, y_max, current%focus, x_min, x_max, nx_global, cpml_thickness, t_end, &
dx, dy, dt, TRIM(data_dir)//C_NULL_CHAR)
current => current%next
ENDDO

END SUBROUTINE Maxwell_consistent_computation_of_EM_fields
```

Listing B.15: Fortran subroutines for populating laser sources at boundaries

```
SUBROUTINE get_source_x_boundary(buffer, laser_id)
REAL(num), DIMENSION(:), INTENT(INOUT) :: buffer
INTEGER, INTENT(IN) :: laser_id
CALL populate_laser_at_boundary(buffer, laser_id, TRIM(data_dir)//C_NULL_CHAR, &
step, ny_global, ny_global_min, ny_global_max)
END SUBROUTINE get_source_x_boundary

SUBROUTINE get_source_y_boundary(buffer, laser_id)
REAL(num), DIMENSION(:), INTENT(INOUT) :: buffer
INTEGER, INTENT(IN) :: laser_id
CALL populate_laser_at_boundary(buffer, laser_id, TRIM(data_dir)//C_NULL_CHAR, &
step, nx_global, nx_global_min, nx_global_max)
END SUBROUTINE get_source_y_boundary
```

Listing B.16: Fortran adaptor for ParaView Catalyst

```
MODULE coprocessor

USE, INTRINSIC :: iso_c_binding
USE fields

IMPLICIT NONE

CONTAINS

SUBROUTINE init_coproc(step, time)
```

```fortran
INTEGER, INTENT(in) :: step
REAL(num), INTENT(in) :: time
INTEGER :: ilen, i
CHARACTER(len=200) :: arg
CALL coprocessorinitialize()
DO i = 1, iargc()
CALL getarg(i, arg)
ilen = len_trim(arg)
arg(ilen+1:) = C_NULL_CHAR
CALL coprocessoraddpythonscript(arg, ilen)
ENDDO
CALL createinputdatadescription(step, time, "essential")
END SUBROUTINE init_coproc

SUBROUTINE run_coproc(step, time)
INTEGER, INTENT(in) :: step
REAL(num), INTENT(in) :: time
INTEGER :: flag, mytid, ntids, n, i, j
INTEGER, DIMENSION(6) :: local_extent, global_extent
INTEGER, DIMENSION(4) :: lim
REAL(num), DIMENSION(3*nx*ny) :: e_field, b_field
#ifdef OPENMP
    INTEGER :: omp_get_thread_num, omp_get_num_threads, omp_get_max_threads
    EXTERNAL :: omp_get_thread_num, omp_get_num_threads, omp_get_max_threads
#endif

local_extent = (/ nx_global_min, nx_global_max, ny_global_min, ny_global_max, 0, 0 /)
global_extent = (/ 1, nx_global, 1, ny_global, 0, 0 /)
lim = (/ 1 + ng, nx + ng, 1 + ng, ny + ng /)

CALL requestdatadescription(step, time, flag)
IF (flag /= 0) THEN
CALL buildgrid(rank, nproc, step, time, local_extent, global_extent, &
x(lim(1):lim(2)), y(lim(3):lim(4)), "essential"//C_NULL_CHAR)

!$omp parallel default(none) private(mytid,i,j,n) &
shared(ntids,lim,e_field,b_field,ex,ey,ez,bx,by,bz)
#ifdef OPENMP
     mytid = OMP_GET_THREAD_NUM()
     ntids = OMP_GET_NUM_THREADS()
#endif
DO j = 0, lim(4) - lim(3), ntids
n = (j+mytid)*(1 + lim(2) - lim(1))*3 + 1
IF(j + mytid <= lim(4)) THEN
DO i = 0, lim(2) - lim(1)
e_field(n + 0) = ex(lim(1) + i, lim(3) + j + mytid)
e_field(n + 1) = ey(lim(1) + i, lim(3) + j + mytid)
e_field(n + 2) = ez(lim(1) + i, lim(3) + j + mytid)
b_field(n + 0) = bx(lim(1) + i, lim(3) + j + mytid)
b_field(n + 1) = by(lim(1) + i, lim(3) + j + mytid)
b_field(n + 2) = bz(lim(1) + i, lim(3) + j + mytid)
n = n + 3
ENDDO
ENDIF
ENDDO
!$omp end parallel

CALL addfield(rank, e_field, "E (V/m)"//C_NULL_CHAR, 3, "essential"//C_NULL_CHAR)
CALL addfield(rank, b_field, "B (T)"//C_NULL_CHAR, 3, "essential"//C_NULL_CHAR)
CALL coprocess()
ENDIF
END SUBROUTINE run_coproc
```

```

SUBROUTINE finalise_coproc()
CALL coprocessorfinalize()
END SUBROUTINE finalise_coproc

END MODULE coprocessor
```

Listing B.17: C++ adaptor for ParaView Catalyst

```
#include "vtkCPDataDescription.h"
#include "vtkCPInputDataDescription.h"
#include "vtkCPProcessor.h"
#include "vtkCPPythonScriptPipeline.h"
#include "vtkCPPythonAdaptorAPI.h"

#ifdef INSITU_DOUBLE_PREC
#include "vtkDoubleArray.h"
#else
#include "vtkFloatArray.h"
#endif

#include "vtkSmartPointer.h"
#include "vtkRectilinearGrid.h"
#include "vtkPointData.h"
#include "vtkImageData.h"
#include "vtkMultiBlockDataSet.h"
#include "vtkMultiPieceDataSet.h"

#include <iostream>
#include <fstream>
#include <vector>
#include <array>

#ifdef __cplusplus
extern "C" {
#endif
void createinputdatadescription_(int* step, double* time, const char* grid_name);
void buildgrid_(int* rank, int* size, int* step, double* time, int* local_extent, int*
    global_extent, double* x_coords, double* y_coords, const char* grid_name);
void addfield_(int* rank, double* input_field, char* name, int* components, const char* grid_name
    );
#ifdef __cplusplus
}
#endif

void createinputdatadescription_(int* step, double* time, const char* grid_name) {
if (!vtkCPPythonAdaptorAPI::GetCoProcessorData()) {
vtkGenericWarningMacro("unable to access coprocessor data");
return;
}
vtkCPPythonAdaptorAPI::GetCoProcessorData()->AddInput(grid_name);
vtkCPPythonAdaptorAPI::GetCoProcessorData()->SetTimeData(*time, static_cast<vtkIdType>(*step));
return;
}

void buildgrid_(int* rank, int* size, int* step, double* time, int* local_extent, int*
    global_extent, double* x_coords, double* y_coords, const char* grid_name) {
if (!vtkCPPythonAdaptorAPI::GetCoProcessorData()) {
vtkGenericWarningMacro("unable to access coprocessor data");
return;
}
```

```
vtkCPPythonAdaptorAPI::GetCoProcessorData()->SetTimeData(*time, static_cast<vtkIdType>(*step));
if(!vtkCPPythonAdaptorAPI::GetCoProcessorData()->GetInputDescriptionByName(grid_name)->GetGrid()) {
vtkCPInputDataDescription* idd = vtkCPPythonAdaptorAPI::GetCoProcessorData()->GetInputDescriptionByName(grid_name);
if (!idd) {
vtkGenericWarningMacro("cannot access data description to attach grid to");
return;
}

vtkSmartPointer<vtkRectilinearGrid> rectilinear_grid =
vtkSmartPointer<vtkRectilinearGrid>::New();
rectilinear_grid->SetExtent(local_extent);

int* ext = rectilinear_grid->GetExtent();
int dim[3] = {ext[1] - ext[0] + 1, ext[3] - ext[2] + 1, ext[5] - ext[4] + 1};

#ifdef INSITU_DOUBLE_PREC
vtkSmartPointer<vtkDoubleArray> x_array = vtkSmartPointer<vtkDoubleArray>::New();
vtkSmartPointer<vtkDoubleArray> y_array = vtkSmartPointer<vtkDoubleArray>::New();
double* x_c = x_coords;
double* y_c = y_coords;
#else
vtkSmartPointer<vtkFloatArray> x_array = vtkSmartPointer<vtkFloatArray>::New();
vtkSmartPointer<vtkFloatArray> y_array = vtkSmartPointer<vtkFloatArray>::New();
float* x_c = new float[dim[0]];
float* y_c = new float[dim[1]];
for(std::size_t i = 0; i < dim[0]; i++) {
x_c[i] = static_cast<float>(x_coords[i]);
}
for(std::size_t i = 0; i < dim[1]; i++) {
y_c[i] = static_cast<float>(y_coords[i]);
}
#endif
x_array->SetNumberOfComponents(1);
y_array->SetNumberOfComponents(1);
x_array->SetArray(x_c, static_cast<vtkIdType>(dim[0]), 1);
y_array->SetArray(y_c, static_cast<vtkIdType>(dim[1]), 1);
rectilinear_grid->SetXCoordinates(x_array);
rectilinear_grid->SetYCoordinates(y_array);

vtkSmartPointer<vtkMultiPieceDataSet> multi_piece = vtkSmartPointer<vtkMultiPieceDataSet>::New();
multi_piece->SetNumberOfPieces(*size);
multi_piece->SetPiece(*rank, rectilinear_grid);

vtkSmartPointer<vtkMultiBlockDataSet> grid = vtkSmartPointer<vtkMultiBlockDataSet>::New();
grid->SetNumberOfBlocks(1);
grid->SetBlock(0, multi_piece);

idd->SetWholeExtent(global_extent);
idd->SetGrid(grid);
}
return;
}

void addfield_(int* rank, double* input_field, char* name, int* components, const char* grid_name) {
if (!vtkCPPythonAdaptorAPI::GetCoProcessorData()) {
vtkGenericWarningMacro("unable to access coprocessor data");
return;
}
vtkCPInputDataDescription* idd = vtkCPPythonAdaptorAPI::GetCoProcessorData()->
```

```
    GetInputDescriptionByName(grid_name);
vtkMultiBlockDataSet* multi_block = vtkMultiBlockDataSet::SafeDownCast(idd->GetGrid());
vtkMultiPieceDataSet* multi_piece = vtkMultiPieceDataSet::SafeDownCast(multi_block->GetBlock(0));
vtkRectilinearGrid* type = vtkRectilinearGrid::SafeDownCast(multi_piece->GetPiece(*rank));
if (!type) {
vtkGenericWarningMacro("no adaptor grid to attach field data to");
return;
}
if (idd->IsFieldNeeded(name)) {
int size = (*components)*type->GetNumberOfPoints();
#ifdef INSITU_DOUBLE_PREC
double* array = input_field;
vtkSmartPointer<vtkDoubleArray> field = vtkSmartPointer<vtkDoubleArray>::New();
#else
float* array = new float[size];
for(std::size_t i = 0; i < size; i++) {
array[i] = static_cast<float>(input_field[i]);
}
vtkSmartPointer<vtkFloatArray> field = vtkSmartPointer<vtkFloatArray>::New();
#endif
field->SetName(name);
field->SetNumberOfComponents(*components);
field->SetArray(array, static_cast<vtkIdType>(size), 1);
type->GetPointData()->AddArray(field);
}
return;
}
```

Listing B.18: Sample visualization pipeline using Python script

```
try: paraview.simple
except: from paraview.simple import *

from paraview import coprocessing

inputs = ['essential']
update_freq = 1
output_freq = 10000

def CreateCoProcessor():
def _CreatePipeline(coprocessor, datadescription):
class Pipeline:

essential = coprocessor.CreateProducer(datadescription, "essential")

multi_block_binary_writer = servermanager.writers.XMLMultiBlockDataWriter(Input=essential,
    DataMode='Appended', EncodeAppendedData=0, HeaderType='UInt32', CompressorType='ZLib')
coprocessor.RegisterWriter(multi_block_binary_writer, filename='full_grid_%t.vtm', freq=
    output_freq)

class CoProcessor(coprocessing.CoProcessor):
def CreatePipeline(self, datadescription):
self.Pipeline = _CreatePipeline(self, datadescription)

coprocessor = CoProcessor()
freqs = {}
for name in inputs:
freqs[name] = [update_freq]

coprocessor.SetUpdateFrequencies(freqs)
return coprocessor
```

```

coprocessor = CreateCoProcessor()
coprocessor.EnableLiveVisualization(True)

def RequestDataDescription(datadescription):
"Callback to populate the request for current timestep"
global coprocessor

if datadescription.GetForceOutput() == True:
for i in range(datadescription.GetNumberOfInputDescriptions()):
datadescription.GetInputDescription(i).AllFieldsOn()
datadescription.GetInputDescription(i).GenerateMeshOn()
return

coprocessor.LoadRequestedData(datadescription)

def DoCoProcessing(datadescription):
"Callback to do co-processing for current timestep"
global coprocessor

coprocessor.UpdateProducers(datadescription)
coprocessor.WriteData(datadescription);
coprocessor.WriteImages(datadescription, rescale_lookuptable=False)
coprocessor.DoLiveVisualization(datadescription, "visualization_node", 11111)
```

Listing B.19: EPOCH CMakeLists file to generate platform-specific build scripts

```
cmake_minimum_required(VERSION 3.1)
project(EPOCH_2D)
enable_language(CXX Fortran)

set(CMAKE_MODULE_PATH ${CMAKE_SOURCE_DIR}/cmake)
set(CMAKE_Fortran_MODULE_DIRECTORY ${CMAKE_SOURCE_DIR}/obj)
set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY ${CMAKE_SOURCE_DIR}/lib)
set(EXECUTABLE_OUTPUT_PATH ${CMAKE_SOURCE_DIR}/bin)

find_package(MPI REQUIRED)
find_package(SDF REQUIRED)

include_directories(${MPI_Fortran_INCLUDE_PATH})
include_directories(${SDF_Fortran_INCLUDE_PATH})
include_directories(src/include)

execute_process(COMMAND ./src/gen_commit_string.sh)
execute_process(COMMAND grep -oP "(?<=COMMIT=)[^ ]+" ./src/COMMIT OUTPUT_VARIABLE COMMIT)
execute_process(COMMAND date +%s OUTPUT_VARIABLE DATE)
execute_process(COMMAND uname -n OUTPUT_VARIABLE MACHINE)

add_definitions('-D_COMMIT="${COMMIT}"')
add_definitions('-D_DATE=${DATE}')
add_definitions('-D_MACHINE="${MACHINE}"')

if(NOT CMAKE_BUILD_TYPE AND NOT CMAKE_CONFIGURATION_TYPES)
message(STATUS "Setting build type to 'Release', Debug mode was not specified.")
set(CMAKE_BUILD_TYPE Release CACHE STRING "Choose the type of build." FORCE)
# Set the possible values of build type for cmake-gui
set_property(CACHE CMAKE_BUILD_TYPE PROPERTY STRINGS "Debug" "Release")
endif()

if(${CMAKE_Fortran_COMPILER_ID} MATCHES "Intel")
set(CMAKE_Fortran_FLAGS_RELEASE "-O3 -xHost -no-prec-div -fno-math-errno -unroll=3 -qopt-
```

```
subscript-in-range -align all")
set(CMAKE_Fortran_FLAGS_DEBUG "-O0 -nothreads -traceback -fltconsistency -C -g -heap-arrays 64 -warn -fp-stack-check -check bounds -fpe0")
elseif(${CMAKE_Fortran_COMPILER_ID} MATCHES "GNU")
set(CMAKE_Fortran_FLAGS_RELEASE "-O2 -fimplicit-none -ffixed-line-length-132")
set(CMAKE_Fortran_FLAGS_DEBUG "-O0 -g -Wall -Wextra -pedantic -fbounds-check -ffpe-trap=invalid,zero,overflow -Wno-unused-parameter")
elseif(${CMAKE_Fortran_COMPILER_ID} MATCHES "PGI")
  set(CMAKE_Fortran_FLAGS_RELEASE "-r8 -fast -fastsse -O3 -Mipa=fast,inline -Minfo")
  set(CMAKE_Fortran_FLAGS_DEBUG "-Mbounds -g")
elseif(${CMAKE_Fortran_COMPILER_ID} MATCHES "G95")
  set(CMAKE_Fortran_FLAGS_RELEASE "-O3")
  set(CMAKE_Fortran_FLAGS_DEBUG "-O0 -g")
elseif(${CMAKE_Fortran_COMPILER_ID} MATCHES "XL")
  set(CMAKE_Fortran_FLAGS_RELEASE "-O5 -qhot -qipa")
  set(CMAKE_Fortran_FLAGS_DEBUG "-O0 -C -g -qfullpath -qinfo -qnosmp -qxflag=dvz -Q! -qnounwind -qnounroll")
else()
message(STATUS "No optimized Fortran compiler flags are known")
message(STATUS "Fortran compiler full path: " ${CMAKE_Fortran_COMPILER})
set(CMAKE_Fortran_FLAGS_RELEASE "-O2")
set(CMAKE_Fortran_FLAGS_DEBUG   "-O0 -g")
endif()

if(${CMAKE_CXX_COMPILER_ID} MATCHES "Intel")
set(CMAKE_CXX_FLAGS_RELEASE "-O3 -std=c++11 -no-prec-div -ansi-alias -qopt-prefetch=4 -unroll-aggressive -m64")
set(CMAKE_CXX_FLAGS_DEBUG "-O0 -std=c++11 -g -traceback -mp1 -fp-trap=common -fp-model strict")
elseif(${CMAKE_CXX_COMPILER_ID} MATCHES "GNU")
set(CMAKE_CXX_FLAGS_RELEASE "-O2 -std=c++11 -msse4 -mtune=native -march=native -funroll-loops -fno-math-errno -ffast-math")
set(CMAKE_CXX_FLAGS_DEBUG "-O0 -std=c++11 -g -pedantic -Wall -Wextra -Wno-unused")
elseif(${CMAKE_CXX_COMPILER_ID} MATCHES "PGI")
  set(CMAKE_CXX_FLAGS_RELEASE "-std=c++0x")
  set(CMAKE_CXX_FLAGS_DEBUG "-std=c++0x")
elseif(${CMAKE_CXX_COMPILER_ID} MATCHES "G95")
  set(CMAKE_CXX_FLAGS_RELEASE "-std=c++0x")
  set(CMAKE_CXX_FLAGS_DEBUG "-std=c++0x")
elseif(${CMAKE_CXX_COMPILER_ID} MATCHES "XL")
  set(CMAKE_CXX_FLAGS_RELEASE "-qlanglvl=extended0x")
  set(CMAKE_CXX_FLAGS_DEBUG "-qlanglvl=extended0x")
else()
message(STATUS "No optimized C++ compiler flags are known")
message(STATUS "C++ compiler full path: " ${CMAKE_CXX_COMPILER})
set(CMAKE_CXX_FLAGS_RELEASE "-O2 -std=c++11")
set(CMAKE_CXX_FLAGS_DEBUG "-O0 -std=c++11 -g")
endif()

set(SOURCES
${CMAKE_SOURCE_DIR}/src/epoch2d.F90
${CMAKE_SOURCE_DIR}/src/boundary.f90
${CMAKE_SOURCE_DIR}/src/fields.f90
${CMAKE_SOURCE_DIR}/src/laser.F90
${CMAKE_SOURCE_DIR}/src/particles.F90
${CMAKE_SOURCE_DIR}/src/shared_data.F90
)

set(FOLDERS deck housekeeping io parser physics_packages user_interaction)
foreach(FOLDER ${FOLDERS})
file(GLOB TMP ${CMAKE_SOURCE_DIR}/src/${FOLDER}/*)
list(APPEND SOURCES ${TMP})
endforeach()
```

```
option(OPENMP "Enable multithreading using OpenMP directives." OFF)
option(PER_SPECIES_WEIGHT "Set every pseudoparticle in a species to represent the same number of real particles." OFF)
option(NO_TRACER_PARTICLES "Don't enable support for tracer particles." OFF)
option(NO_PARTICLE_PROBES "Don't enable support for particle probes." OFF)
option(PARTICLE_SHAPE_TOPHAT "Use second order particle weighting." OFF)
option(PARTICLE_SHAPE_BSPLINE3 "Use fifth order particle weighting." OFF)
option(PARTICLE_ID "Include a unique global particle ID using an 8-byte integer." OFF)
option(PARTICLE_ID4 "Include a unique global particle ID using an 4-byte integer." OFF)
option(PARTICLE_COUNT_UPDATE "Keep global particle counts up to date." OFF)
option(PHOTONS "Include QED routines" OFF)
option(TRIDENT_PHOTONS "Use the Trident process for pair production." OFF)
option(PREFETCH "Use Intel-specific 'mm_prefetch' calls to load next particle in the list into cache ahead of time." OFF)
option(PARSER_DEBUG "Turn on debugging." OFF)
option(PARTICLE_DEBUG "Turn on debugging." OFF)
option(MPI_DEBUG "Turn on debugging." OFF)
option(SIMPLIFY_DEBUG "Turn on debugging." OFF)
option(NO_IO "Don't generate any output at all. Useful for benchmarking." OFF)
option(COLLISIONS_TEST "Bypass the main simulation and only perform collision tests." OFF)
option(PER_PARTICLE_CHARGE_MASS "specify charge and mass per particle rather than per species." OFF)
option(PARSER_CHECKING "Perform checks on evaluated deck expressions." OFF)
option(USE_INSITU "Link epoch with ParaView Catalyst." OFF)
option(INSITU_DOUBLE_PREC "Double precision for data exported insitu." OFF)
option(TIGHT_FOCUSING "Maxwell consistent computation of EM fields at boundary for tight-focusing." OFF)

if(OPENMP)
message(STATUS "Option 'OPENMP' enabled")
add_definitions("-DOPENMP")
if(${CMAKE_Fortran_COMPILER_ID} MATCHES "Intel")
set(CMAKE_Fortran_FLAGS_RELEASE "${CMAKE_Fortran_FLAGS_RELEASE} -qopenmp")
set(CMAKE_Fortran_FLAGS_DEBUG "${CMAKE_Fortran_FLAGS_DEBUG} -qopenmp")
elseif(${CMAKE_Fortran_COMPILER_ID} MATCHES "GNU")
set(CMAKE_Fortran_FLAGS_RELEASE "${CMAKE_Fortran_FLAGS_RELEASE} -fopenmp")
set(CMAKE_Fortran_FLAGS_DEBUG "${CMAKE_Fortran_FLAGS_DEBUG} -fopenmp")
else()
message(STATUS "Fortran OpenMP compiler flag not known.")
endif()
if(${CMAKE_CXX_COMPILER_ID} MATCHES "Intel")
set(CMAKE_CXX_FLAGS_RELEASE "${CMAKE_CXX_FLAGS_RELEASE} -qopenmp")
set(CMAKE_CXX_FLAGS_DEBUG "${CMAKE_CXX_FLAGS_DEBUG} -qopenmp")
elseif(${CMAKE_CXX_COMPILER_ID} MATCHES "GNU")
set(CMAKE_CXX_FLAGS_RELEASE "${CMAKE_CXX_FLAGS_RELEASE} -fopenmp")
set(CMAKE_CXX_FLAGS_DEBUG "${CMAKE_CXX_FLAGS_DEBUG} -fopenmp")
else()
message(STATUS "C++ OpenMP compiler flag not known.")
endif()
endif()

if(TIGHT_FOCUSING AND NOT FFT_LIBRARY)
message(STATUS "No FFT library specified. Setting FFT library to 'none'.")
set(FFT_LIBRARY none CACHE STRING "Choose the FFT library." FORCE)
set_property(CACHE FFT_LIBRARY PROPERTY STRINGS "FFTW" "MKL" "None")
endif()

if(PER_SPECIES_WEIGHT)
message(STATUS "Option 'PER_SPECIES_WEIGHT' enabled")
add_definitions("-DPER_SPECIES_WEIGHT")
endif()
```

```

if(NO_TRACER_PARTICLES)
message(STATUS "Option 'NO_TRACER_PARTICLES' enabled")
add_definitions("-DNO_TRACER_PARTICLES")
endif()

if(NO_PARTICLE_PROBES)
message(STATUS "Option 'NO_PARTICLE_PROBES' enabled")
add_definitions("-DNO_PARTICLE_PROBES")
endif()

if(PARTICLE_SHAPE_TOPHAT)
message(STATUS "Option 'PARTICLE_SHAPE_TOPHAT' enabled")
add_definitions("-DPARTICLE_SHAPE_TOPHAT")
endif()

if(PARTICLE_SHAPE_BSPLINE3)
message(STATUS "Option 'PARTICLE_SHAPE_BSPLINE3' enabled")
add_definitions("-DPARTICLE_SHAPE_BSPLINE3")
endif()

if(PARTICLE_ID)
message(STATUS "Option 'PARTICLE_ID' enabled")
add_definitions("-DPARTICLE_ID")
endif()

if(PARTICLE_ID4)
message(STATUS "Option 'PARTICLE_ID4' enabled")
add_definitions("-DPARTICLE_ID4")
endif()

if(PARTICLE_COUNT_UPDATE)
message(STATUS "Option 'PARTICLE_COUNT_UPDATE' enabled")
add_definitions("-DPARTICLE_COUNT_UPDATE")
endif()

if(PHOTONS)
message(STATUS "Option 'PHOTONS' enabled")
add_definitions("-DPHOTONS")
endif()

if(TRIDENT_PHOTONS)
message(STATUS "Option 'TRIDENT_PHOTONS' enabled")
add_definitions("-DTRIDENT_PHOTONS")
endif()

if(PREFETCH)
message(STATUS "Option 'PREFETCH' enabled")
add_definitions("-DPREFETCH")
endif()

if(PARSER_DEBUG)
message(STATUS "Option 'PARSER_DEBUG' enabled")
add_definitions("-DPARSER_DEBUG")
endif()

if(PARTICLE_DEBUG)
message(STATUS "Option 'PARTICLE_DEBUG' enabled")
add_definitions("-DPARTICLE_DEBUG")
endif()

if(MPI_DEBUG)
```

```
message(STATUS "Option 'MPI_DEBUG' enabled")
add_definitions("-DMPI_DEBUG")
endif()

if(SIMPLIFY_DEBUG)
message(STATUS "Option 'SIMPLIFY_DEBUG' enabled")
add_definitions("-DSIMPLIFY_DEBUG")
endif()

if(NO_IO)
message(STATUS "Option 'NO_IO' enabled")
add_definitions("-DNO_IO")
endif()

if(COLLISIONS_TEST)
message(STATUS "Option 'COLLISIONS_TEST' enabled")
add_definitions("-DCOLLISIONS_TEST")
endif()

if(PER_PARTICLE_CHARGE_MASS)
message(STATUS "Option 'PER_PARTICLE_CHARGE_MASS' enabled")
add_definitions("-DPER_PARTICLE_CHARGE_MASS")
endif()

if(PARSER_CHECKING)
message(STATUS "Option 'PARSER_CHECKING' enabled")
add_definitions("-DPARSER_CHECKING")
endif()

if(USE_INSITU)
message(STATUS "Option 'USE_INSITU' enabled")
find_package(ParaView 5.2 REQUIRED COMPONENTS vtkPVPythonCatalyst)
include(${PARAVIEW_USE_FILE})
file(GLOB ADAPTOR ${CMAKE_SOURCE_DIR}/src/adaptor/*)
list(APPEND SOURCES ${ADAPTOR})
add_definitions("-DINSITU")
if(NOT PARAVIEW_USE_MPI)
message(SEND_ERROR "ParaView must be built with MPI enabled")
endif()
if(INSITU_DOUBLE_PREC)
message(STATUS "Option 'INSITU_DOUBLE_PREC' enabled")
add_definitions("-DINSITU_DOUBLE_PREC")
endif()
endif()

if(TIGHT_FOCUSING)
message(STATUS "Option 'TIGHT_FOCUSING' enabled")
file(GLOB FOCUSING ${CMAKE_SOURCE_DIR}/src/focusing/interface.f90)
list(APPEND SOURCES ${FOCUSING})
include_directories(${MPI_CXX_INCLUDE_PATH})
add_definitions("-DTIGHT_FOCUSING")
set(FOCUS_SRC
${CMAKE_SOURCE_DIR}/src/focusing/main.cpp
${CMAKE_SOURCE_DIR}/src/focusing/domain_param.cpp
${CMAKE_SOURCE_DIR}/src/focusing/laser_param.cpp
${CMAKE_SOURCE_DIR}/src/focusing/global.cpp
${CMAKE_SOURCE_DIR}/src/focusing/laser_bcs.cpp
)
include_directories(${CMAKE_SOURCE_DIR}/src/focusing/inc)
if(${FFT_LIBRARY} MATCHES "FFTW")
if(OPENMP)
message(SEND_ERROR "Multi-threaded FFTW routines are not supported. Disable OpenMP.")
```

```
endif()
message(STATUS "FFT library: FFTW")
message(STATUS "Option 'USE_FFTW' enabled")
add_definitions("-DUSE_FFTW")
find_package(FFTW REQUIRED)
include_directories(${FFTW_INCLUDES})
endif()
if(${FFT_LIBRARY} MATCHES "MKL")
if(NOT ${CMAKE_CXX_COMPILER_ID} MATCHES "Intel")
message(SEND_ERROR "MKL library can be used only with Intel compilers")
endif()
message(STATUS "FFT library: MKL")
message(STATUS "Option 'USE_MKL' enabled")
add_definitions("-DUSE_MKL")
set(CMAKE_CXX_FLAGS_RELEASE "${CMAKE_CXX_FLAGS_RELEASE} -mkl")
set(CMAKE_CXX_FLAGS_DEBUG "${CMAKE_CXX_FLAGS_DEBUG} -mkl")
set(CMAKE_Fortran_FLAGS_RELEASE "${CMAKE_Fortran_FLAGS_RELEASE} -mkl")
set(CMAKE_Fortran_FLAGS_DEBUG "${CMAKE_Fortran_FLAGS_DEBUG} -mkl")
endif()
if(${FFT_LIBRARY} MATCHES "None")
message(STATUS "FFT library: None")
endif()
add_library(focus ${FOCUS_SRC})
if(${FFT_LIBRARY} MATCHES "FFTW")
#if(OPENMP)
#  target_link_libraries(focus LINK_PUBLIC ${FFTW_LIBRARIES} ${FFTW_LIBRARIES_OMP})
#else()
target_link_libraries(focus LINK_PUBLIC ${FFTW_LIBRARIES})
#endif()
endif()
set_target_properties(focus PROPERTIES LINKER_LANGUAGE CXX)
endif()

add_executable(epoch2d ${SOURCES})
target_link_libraries(epoch2d LINK_PRIVATE ${MPI_Fortran_LIBRARIES} ${SDF_Fortran_LIBRARIES})
if(USE_INSITU)
target_link_libraries(epoch2d LINK_PRIVATE vtkPVPythonCatalyst vtkParallelMPI)
endif()
if(TIGHT_FOCUSING)
target_link_libraries(epoch2d LINK_PRIVATE ${MPI_CXX_LIBRARIES} focus)
endif()
set_target_properties(epoch2d PROPERTIES LINKER_LANGUAGE Fortran)
```

Appendix C

CD content

All the stuff created during the work. Nothing more, nothing less. Enjoy.

directory/file	specification
`master_thesis/`	directory containing this document and all other related sources
`epoch/`	directory containing EPOCH source code (cloned on 15-04-2017)
`scripts/`	directory containing scripts intended for post-processing of simulation data