A spectral, quasi-cylindrical and dispersion-free Particle-In-Cell algorithm

Rémi Lehe[1,], Manuel Kirchen[1], Igor A. Andriyash[1], Brendan B. Godfrey[1,1], Jean-Luc Vay[1]

[a] *Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*
[b] *Center for Free-Electron Laser Science & Department of Physics, University of Hamburg, 22761 Hamburg, Germany*
[c] *LOA, ENSTA ParisTech, CNRS, École polytechnique, Université Paris-Saclay, 828 bd des Maréchaux, 91762 Palaiseau cédex France*
[d] *University of Maryland, College Park, MD 20742, USA*

Abstract

We propose a spectral Particle-In-Cell (PIC) algorithm that is based on the combination of a Hankel transform and a Fourier transform. For physical problems that have close-to-cylindrical symmetry, this algorithm can be much faster than full 3D PIC algorithms. In addition, unlike standard finite-difference PIC codes, the proposed algorithm is free of spurious numerical dispersion, in vacuum. This algorithm is benchmarked in several situations that are of interest for laser-plasma interactions. These benchmarks show that it avoids a number of numerical artifacts, that would otherwise affect the physics in a standard PIC algorithm – including the zero-order numerical Cherenkov effect.

Keywords: particle-in-cell, pseudo-spectral, Hankel transform, cylindrical geometry

Introduction

Particle-In-Cell (PIC) algorithms [**?** **?**] are extensively used in several areas of physics, including the study of astrophysical plasmas, fusion plasmas, laser-plasma interactions and accelerator physics. Yet, despite their wide use, PIC algorithms can be very computationally demanding, especially in three-dimensions, and are still subject to a range of numerical artifacts. These shortcomings can be particularly significant when simulating accelerated particle beams, or laser-plasma interactions (such as laser-wakefield acceleration) for two reasons:

- these systems often have close-to-cylindrical symmetry (e.g. particle beams and laser pulses are often cylindrically symmetric). This prevents the use of 2D Cartesian or 2D cylindrical PIC algorithms (which are only well-suited for slab-like and azimuthal symmetry), and is instead often dealt with by using 3D Cartesian PIC algorithms, which can be very computationally expensive;
- the physical objects of interest (e.g. the laser, or the accelerated particle beam) often propagate close to the speed of light. This makes them very sensitive to *spurious numerical dispersion*, i.e. the fact that the electromagnetic waves do not propagate exactly at the physical speed of light in a standard PIC code, but travel instead at a spuriously-altered, resolution-dependent velocity. In the above-mentioned cases, spurious numerical dispersion can lead to substantial numerical artifacts which can mask or disrupt the physics at stake in the simulation. This includes, for instance, numerical Cherenkov effects in general [**?**], but also more specific artifacts, such as e.g. the erroneous prediction of the dephasing length in laser-wakefield acceleration [**?**].

Yet several modifications can be made to the PIC algorithm, in order to mitigate these difficulties and increase the speed and accuracy of the simulations in these physical situations:

- one of these modifications is the development of cylindrical PIC algorithms with azimuthal Fourier decomposition [**?** **?** **?**] of the electromagnetic field components (sometimes referred to as *quasi-3D* algorithms, or as *quasi-cylindrical* algorithms as we do here). By taking into account the symmetry of the system, these algorithms can typically reduce the cost of the simulation to a few times that of a 2D Cartesian simulation, instead of that of a full 3D Cartesian simulation. Moreover, unlike

*Corresponding author. Tel:+1 510-486-6785

2D Cartesian algorithms, these algorithms are well adapted to close-to-cylindrical physical systems and can accurately capture physical effects that are intrinsically 3D (such as e.g. the non-linear self-focusing of an intense laser in a plasma [**?**]);

- a second, separate modification was introduced by the development of *spectral* Cartesian PIC algorithms [**?** **?** **?** **?** **?**] i.e. algorithms that solve the Maxwell equations in Fourier space. (These algorithm are also sometimes referred to as *pseudo-spectral algorithms* when they make use of an intermediate interpolation grid in real space, and this term then also applies to the algorithm presented here.) These algorithms contrast with *finite-difference* algorithms, which solve the Maxwell equations by approximating the derivatives as finite differences on a discrete spatial grid. Importantly, while finite-difference algorithms suffer from spurious numerical dispersion, there is a class of spectral algorithms – often referred to as *Pseudo-Spectral Analytical Time Domain* (PSATD) algorithms [**?** **?**] – which exhibits no spurious numerical dispersion in vacuum. As a consequence, these algorithms are free from the associated numerical artifacts. Moreover, it was shown recently [**?** **?** **?** **?**] that spectral PIC algorithms have better stability properties when performing PIC simulations in a Lorentz-boosted frame [**?** **?** **?**]. These stability properties are very promising, since boosted-frame simulations can be faster than their laboratory-frame counterparts by several orders of magnitude [**?**].

Even though these two improvements are both very valuable, they cannot be combined with each other in their present formulation. Fundamentally, this is because the *quasi-cylindrical* algorithm uses a cylindrical system of coordinates, whereas most spectral algorithms (and in particular the PSATD algorithms) were developed in a Cartesian system of coordinates. This incompatibility is however not definitive, and the aim of this article is to overcome these differences by developing a *spectral quasi-cylindrical* formalism. In this document, we derive this formalism, and we show that it can be used to build a PIC algorithm that combines the speed of *quasi-cylindrical* algorithms with the accuracy and lack of spurious numerical dispersion of the PSATD algorithms.

Although the algorithm described here is, to our knowledge, unique in its capabilities, there are several other existing codes which have some similarities with it. One such example is the hybrid, quasi-cylindrical version of OSIRIS [**?**]. This algorithm involves a Fourier transform in the longitudinal direction, but retains a finite-difference formulation in the transverse (radial) direction. As a consequence, this algorithm is not fully spectral, nor fully dispersion-free. Another such example is the code PlaRes [**?**], which is designed to simulate the physics of free-electron lasers (FEL). This code also uses a spectral quasi-cylindrical formalism, but it models the fields in narrow spectral intervals around the resonant FEL frequencies, relies on the scalar and vector potentials ϕ and $\boldsymbol{A}$ and uses no spatial grid. As a result, PlaRes differs significantly from the standard PIC formulation, and from the algorithm described here.

In the present article, we start by deriving the equations of our spectral quasi-cylindrical formalism (**??**). We then explain how these equations were discretized and implemented in a fully working PIC code (**??**), and we report on the results of a number of benchmarks that were performed with this code (**??**). Finally, we describe two typical physical situations in which our spectral algorithm performs better than standard finite-difference algorithms (**??**).

1. Representation of the fields and continuous equations

1.1. A reminder on spectral Cartesian codes

It is well-known that the Maxwell equations in Cartesian coordinates

$$\frac{1}{c^2}\partial_t E_x = \partial_y B_z - \partial_z B_y - \mu_0 j_x \qquad \partial_t B_x = -\partial_y E_z + \partial_z E_y \tag{1a}$$

$$\frac{1}{c^2}\partial_t E_y = \partial_z B_x - \partial_x B_z - \mu_0 j_y \qquad \partial_t B_y = -\partial_z E_x + \partial_x E_z \tag{1b}$$

$$\frac{1}{c^2}\partial_t E_z = \partial_x B_y - \partial_y B_x - \mu_0 j_z \qquad \partial_t B_z = -\partial_x E_y + \partial_y E_x \tag{1c}$$

can be solved by representing the fields as a sum of Fourier modes.

$$F_u(\boldsymbol{r}) = \frac{1}{(2\pi)^3}\int_{-\infty}^{\infty}\mathrm{d}k_x \int_{-\infty}^{\infty}\mathrm{d}k_y \int_{-\infty}^{\infty}\mathrm{d}k_z \; \mathcal{F}_u(\boldsymbol{k})\, e^{i(k_x x + k_y y + k_z z)} \tag{2}$$

with

$$\mathcal{F}_u(\boldsymbol{k}) = \int_{-\infty}^{\infty} \mathrm{d}x \int_{-\infty}^{\infty} \mathrm{d}y \int_{-\infty}^{\infty} \mathrm{d}z \; F_u(\boldsymbol{r})\, e^{-i(k_x x + k_y y + k_z z)} \tag{3}$$

where F is any of the fields E, B or j, and where u is either x, y or z. $\mathcal{F}$ represents the Fourier components of F, which will be denoted $\mathcal{E}$, $\mathcal{B}$ or $\mathcal{J}$ depending on whether F represents E, B or j. With this representation, the different Fourier modes decouple and the equations **??????** become

$$\frac{1}{c^2}\partial_t \mathcal{E}_x = ik_y \mathcal{B}_z - ik_z \mathcal{B}_y - \mu_0 \mathcal{J}_x \qquad \partial_t \mathcal{B}_x = -ik_y \mathcal{E}_z + ik_z \mathcal{E}_y \tag{4a}$$

$$\frac{1}{c^2}\partial_t \mathcal{E}_y = ik_z \mathcal{B}_x - ik_x \mathcal{B}_z - \mu_0 \mathcal{J}_y \qquad \partial_t \mathcal{B}_y = -ik_z \mathcal{E}_x + ik_x \mathcal{E}_z \tag{4b}$$

$$\frac{1}{c^2}\partial_t \mathcal{E}_z = ik_x \mathcal{B}_y - ik_y \mathcal{B}_x - \mu_0 \mathcal{J}_z \qquad \partial_t \mathcal{B}_z = -ik_x \mathcal{E}_y + ik_y \mathcal{E}_x \tag{4c}$$

The Fourier coefficients $\mathcal{E}$ and $\mathcal{B}$ can be then integrated in time, and transformed back into real space using **??**. This is the core principle of spectral Cartesian algorithms, including the PSATD algorithms.

1.2. Spectral quasi-cylindrical representation

The Fourier representation **??** is no longer the appropriate representation when the Maxwell equations are written in cylindrical coordinates.

$$\frac{1}{c^2}\partial_t E_r = \frac{1}{r}\partial_\theta B_z - \partial_z B_\theta - \mu_0 j_r \qquad \partial_t B_r = -\frac{1}{r}\partial_\theta E_z + \partial_z E_\theta \tag{5a}$$

$$\frac{1}{c^2}\partial_t E_\theta = \partial_z B_r - \partial_r B_z - \mu_0 j_\theta \qquad \partial_t B_\theta = -\partial_z E_r + \partial_r E_z \tag{5b}$$

$$\frac{1}{c^2}\partial_t E_z = \frac{1}{r}\partial_r r B_\theta - \frac{1}{r}\partial_\theta B_r - \mu_0 j_z \qquad \partial_t B_z = -\frac{1}{r}\partial_r r E_\theta + \frac{1}{r}\partial_\theta E_r \tag{5c}$$

When replacing the representation **??** into the **??????**, the Fourier modes do not decouple. Instead one has to use the Fourier-Hankel representation:

$$F_z(\boldsymbol{r}) = \frac{1}{(2\pi)^2} \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{d}k_z \int_0^{\infty} k_\perp \mathrm{d}k_\perp \; \hat{\mathcal{F}}_{z,m}(k_z, k_\perp)\; J_m(k_\perp r)\, e^{-im\theta + ik_z z} \tag{6a}$$

$$F_r(\boldsymbol{r}) = \frac{1}{(2\pi)^2} \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{d}k_z \int_0^{\infty} k_\perp \mathrm{d}k_\perp \; \left(\hat{\mathcal{F}}_{+,m}(k_z, k_\perp)\; J_{m+1}(k_\perp r) + \hat{\mathcal{F}}_{-,m}(k_z, k_\perp)\; J_{m-1}(k_\perp r) \right) e^{-im\theta + ik_z z} \tag{6b}$$

$$F_\theta(\boldsymbol{r}) = \frac{1}{(2\pi)^2} \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty} \mathrm{d}k_z \int_0^{\infty} k_\perp \mathrm{d}k_\perp \; i \left(\hat{\mathcal{F}}_{+,m}(k_z, k_\perp)\; J_{m+1}(k_\perp r) - \hat{\mathcal{F}}_{-,m}(k_z, k_\perp)\; J_{m-1}(k_\perp r) \right) e^{-im\theta + ik_z z} \tag{6c}$$

where F is either E, B or j, where J_m denotes the Bessel function of order m, and where $\hat{\mathcal{F}}_{z,m}$, $\hat{\mathcal{F}}_{+,m}$ and $\hat{\mathcal{F}}_{-,m}$ represent the spectral components of F. (See **??** for a derivation of the above equations.) Conversely, the spectral components $\hat{\mathcal{F}}_{z,m}$, $\hat{\mathcal{F}}_{+,m}$ and $\hat{\mathcal{F}}_{-,m}$ are related to the real-space fields F_r, F_θ and F_z by:

$$\hat{\mathcal{F}}_{z,m}(k_z, k_\perp) = \int_{-\infty}^{\infty} \mathrm{d}z \int_0^{\infty} r\mathrm{d}r \int_0^{2\pi} \mathrm{d}\theta \; F_z(\boldsymbol{r})\; J_m(k_\perp r) e^{im\theta - ik_z z} \tag{7a}$$

$$\hat{\mathcal{F}}_{+,m}(k_z, k_\perp) = \int_{-\infty}^{\infty} \mathrm{d}z \int_0^{\infty} r\mathrm{d}r \int_0^{2\pi} \mathrm{d}\theta \; \frac{F_r(\boldsymbol{r}) - iF_\theta(\boldsymbol{r})}{2}\; J_{m+1}(k_\perp r) e^{im\theta - ik_z z} \tag{7b}$$

$$\hat{\mathcal{F}}_{-,m}(k_z, k_\perp) = \int_{-\infty}^{\infty} \mathrm{d}z \int_0^{\infty} r\mathrm{d}r \int_0^{2\pi} \mathrm{d}\theta \; \frac{F_r(\boldsymbol{r}) + iF_\theta(\boldsymbol{r})}{2}\; J_{m-1}(k_\perp r) e^{im\theta - ik_z z} \tag{7c}$$

In the above equations, notice here that the Cartesian component F_z and cylindrical components F_r, F_θ do not transform in the same manner, which is due to their different behavior close to the axis. (Again, see **??** for a more detailed explanation.) Note also that scalar fields (like ρ) transform in the same way as the Cartesian component F_z.

When replacing **??????** into the Maxwell equations in cylindrical coordinates **??????**, the different modes decouple, and the equations for the spectral coefficients become:

$$\frac{1}{c^2}\partial_t \hat{\mathcal{E}}_{+,m} = -\frac{ik_\perp}{2}\hat{\mathcal{B}}_{z,m} + k_z\hat{\mathcal{B}}_{+,m} - \mu_0\hat{\mathcal{J}}_{+,m} \qquad \partial_t\hat{\mathcal{B}}_{+,m} = \frac{ik_\perp}{2}\hat{\mathcal{E}}_{z,m} - k_z\hat{\mathcal{E}}_{+,m} \tag{8a}$$

$$\frac{1}{c^2}\partial_t \hat{\mathcal{E}}_{-,m} = -\frac{ik_\perp}{2}\hat{\mathcal{B}}_{z,m} - k_z\hat{\mathcal{B}}_{-,m} - \mu_0\hat{\mathcal{J}}_{-,m} \qquad \partial_t\hat{\mathcal{B}}_{-,m} = \frac{ik_\perp}{2}\hat{\mathcal{E}}_{z,m} + k_z\hat{\mathcal{E}}_{-,m} \tag{8b}$$

$$\frac{1}{c^2}\partial_t \hat{\mathcal{E}}_{z,m} = ik_\perp\hat{\mathcal{B}}_{+,m} + ik_\perp\hat{\mathcal{B}}_{-,m} - \mu_0\hat{\mathcal{J}}_{z,m} \qquad \partial_t\hat{\mathcal{B}}_{z,m} = -ik_\perp\hat{\mathcal{E}}_{+,m} - ik_\perp\hat{\mathcal{E}}_{-,m} \tag{8c}$$

(See **??** for a derivation of these equations.) Notice that these equations have a similar structure as the spectral Cartesian equations **??????**, as they also involve the product of the fields with the components of $\boldsymbol{k}$, in their right-hand side. They do differ however in their details, as evidenced by the signs, the factors $1/2$ and the presence or absence of the complex number i in some terms.

Similarly, in this formalism, the conservation equations $\boldsymbol{\nabla}\cdot\boldsymbol{E} = \rho/\epsilon_0$ and $\boldsymbol{\nabla}\cdot\boldsymbol{B} = 0$ become

$$k_\perp(\hat{\mathcal{E}}_{+,m} - \hat{\mathcal{E}}_{-,m}) + ik_z\hat{\mathcal{E}}_{z,m} = \frac{\hat{\rho}_m}{\epsilon_0} \qquad k_\perp(\hat{\mathcal{B}}_{+,m} - \hat{\mathcal{B}}_{-,m}) + ik_z\hat{\mathcal{B}}_{z,m} = 0 \tag{9}$$

As expected, the above conservation equations **??** are preserved by the Maxwell equations **??????**, provided that the current satisfies $\partial_t\rho + \boldsymbol{\nabla}\cdot\boldsymbol{j} = 0$, i.e. in spectral space:

$$\partial_t\hat{\rho}_m + k_\perp(\hat{\mathcal{J}}_{+,m} - \hat{\mathcal{J}}_{-,m}) + ik_z\hat{\mathcal{J}}_{z,m} = 0 \tag{10}$$

(This can be checked, for instance, by differentiating **??** in time and by using **??????** to re-express the time derivatives.)

As in a spectral Cartesian PIC codes, the equations **??????** can be integrated in time, and the fields can then be transformed back into real space, using **??????**. Therefore, the methods used to integrate the fields in time in a spectral Cartesian code (e.g. PSATD) should be transposable to this spectral quasi-cylindrical formalism.

However, the advantage of this formalism over its Cartesian counterpart is that the sum over m in **??????** can generally be truncated to only a few terms, for physical situations that have close-to-cylindrical symmetry. (A similar truncation is done in [**?**].) This is because the different values of m correspond to different azimuthal modes of the form $e^{-im\theta}$, and because these modes are typically zero for large values of $|m|$, in situations with close-to-cylindrical symmetry. As a result, the representation of the fields is reduced to a few 2D arrays $\hat{\mathcal{F}}_m(k_z, k_\perp)$ instead of the Cartesian 3D arrays $\mathcal{F}(k_x, k_y, k_z)$. This reduction makes the manipulation of the fields much more computationally efficient.

2. Numerical implementation

2.1. Overview of the algorithm

The PIC algorithm described here uses the above-mentioned representation of the fields, in order to solve the Maxwell equations and the motion of charged particles in a finite-size simulation box.

Note that, when using spectral algorithms in a finite box, it is often necessary to arbitrarily adopt specific boundary conditions (which may not match the physics at stake), in order to be able to conveniently represent the fields. For instance, in Cartesian spectral codes, periodic boundaries are arbitrarily chosen in order to be able to represent the fields as a discrete sum of Fourier modes. However, this is mostly for mathematical convenience and it does not preclude the application, at each timestep, of another type of boundary condition in real space and *within* the finite box (e.g. Perfectly Matched Layers), before transforming the fields to spectral space (see e.g. [**?** **?**]). In the same spirit, here we arbitrarily impose periodic boundary conditions along z, and Dirichlet boundary conditions along r (more specifically $\boldsymbol{E}(r_{max}) = \boldsymbol{0}$ and $\boldsymbol{B}(r_{max}) = \boldsymbol{0}$), since in this case the fields can be decomposed into a discrete Fourier-Bessel series (see e.g. [**?**]). Yet again, this does not prevent the practical application of other boundary conditions in real space.

Although, as explained in **??**, the fields are represented by a few 2D arrays, the particles are still distributed in 3D and their motion is integrated in 3D Cartesian coordinates. As in a spectral Cartesian code, we do not perform the current deposition and field gathering directly from the macroparticles to the spectral space. (This is inefficient since the deposition and gathering are local operations in real space – i.e. affect only the few cells next to the macroparticle – but global operations in spectral space –

i.e. they affect all the spectral modes simultaneously.) Instead we use an intermediate grid where these operations can be performed locally, and where the fields $\hat{F}_{u,m}$ are defined by

$$F_u(\boldsymbol{r}) = \sum_{m=-\infty}^{\infty} \hat{F}_{u,m}(r,z) e^{-im\theta} \tag{11}$$

$$\hat{F}_{u,m}(r,z) = \frac{1}{2\pi} \int_0^{2\pi} \mathrm{d}\theta \; F_u(\boldsymbol{r}) e^{im\theta} \tag{12}$$

where F is either E, B or j and u is either z, r or θ. Notice that this representation is the same as that of [**?** **?**]. In this representation, the spectral decomposition in the azimuthal direction is preserved, since the factors $e^{im\theta}$ can be efficiently computed from the particle Cartesian positions x, y, z by using the relation $e^{im\theta} = (x+iy)^m / r^m$ [**?**].

After the particles deposit their charge and current onto this intermediate grid, the fields of the grid are transformed into spectral space where, as mentioned in **??**, the Maxwell equations can be easily integrated. From **????** and **?????????????**, the transformation between the intermediate grid $\hat{F}_m(r,z)$ and the spectral grid $\hat{\mathcal{F}}_m(k_\perp, k_z)$ is:

$$\hat{\mathcal{F}}_{z,m}(k_\perp, k_z) = \mathrm{HT}_m[\ \mathrm{FT}[\ \hat{F}_{z,m}(r,z)\ ]\ ] \tag{13a}$$

$$\hat{\mathcal{F}}_{+,m}(k_\perp, k_z) = \mathrm{HT}_{m+1}\left[\ \mathrm{FT}\left[\ \frac{\hat{F}_{r,m} - i\hat{F}_{\theta,m}}{2}\ \right]\ \right] \tag{13b}$$

$$\hat{\mathcal{F}}_{-,m}(k_\perp, k_z) = \mathrm{HT}_{m-1}\left[\ \mathrm{FT}\left[\ \frac{\hat{F}_{r,m} + i\hat{F}_{\theta,m}}{2}\ \right]\ \right] \tag{13c}$$

and

$$\hat{F}_{z,m}(r,z) = \mathrm{IFT}[\ \mathrm{IHT}_m[\hat{\mathcal{F}}_{z,m}(k_\perp, k_z)]\ ] \tag{14a}$$

$$\hat{F}_{r,m}(r,z) = \mathrm{IFT}\left[\ \mathrm{IHT}_{m+1}[\hat{\mathcal{F}}_{+,m}(k_\perp, k_z)] + \mathrm{IHT}_{m-1}[\hat{\mathcal{F}}_{-,m}(k_\perp, k_z)]\ \right] \tag{14b}$$

$$\hat{F}_{\theta,m}(r,z) = i\ \mathrm{IFT}\left[\ \mathrm{IHT}_{m+1}[\hat{\mathcal{F}}_{+,m}(k_\perp, k_z)] - \mathrm{IHT}_{m-1}[\hat{\mathcal{F}}_{-,m}(k_\perp, k_z)]\ \right] \tag{14c}$$

where FT represents a Fourier Transform along the z axis and HT_n represents a Hankel Transform of order n along the transverse r axis, and where IFT and IHT_n represent the corresponding inverse transformations:

$$\mathrm{FT}[f]\,(k_z) \equiv \int_{-\infty}^{\infty} \mathrm{d}z\, e^{-ik_z z}\ f(z) \qquad \mathrm{IFT}[g]\,(z) \equiv \frac{1}{2\pi} \int_{-\infty}^{\infty} \mathrm{d}k_z\, e^{ik_z z}\ g(k_z) \tag{15}$$

$$\mathrm{HT}_n[f]\,(k_\perp) \equiv 2\pi \int_0^{\infty} r\mathrm{d}r\ J_n(k_\perp r)\ f(r) \qquad \mathrm{IHT}_n[g]\,(r) \equiv \frac{1}{2\pi} \int_0^{\infty} k_\perp \mathrm{d}k_\perp\ J_n(k_\perp r)\ g(k_\perp) \tag{16}$$

The above equations show that the transformation from the intermediate grid ($\hat{F}_{u,m}$) to the spectral grid ($\hat{\mathcal{F}}_{u,m}$) is the combination of a Fourier transform (in z) and a Hankel transform (in r). The Fourier transform in z can be discretized through a Fast Fourier Transform (FFT) algorithm, which requires an evenly-spaced grid in z and in k_z. On the other hand, there is more freedom of choice for the Discrete Hankel Transform (DHT), and the implementation that we chose is described in the next section (**??**) and in the **??**.

?? gives an overview of the successive steps involved in one PIC cycle, including the respective role of the intermediate grid ($\hat{F}_{u,m}$) and spectral grid ($\hat{\mathcal{F}}_{u,m}$). Note that, in the PSATD scheme that we chose (and which is described in more details in **??**), all the fields are defined at integer timesteps, except for the currents, which are defined at half timesteps. **????????** describe the successive steps of the PIC cycle in more details.

2.2. Transverse discretization of the intermediate grid, of the spectral grid and of the Hankel Transform

When transforming from the intermediate to the spectral grid, the FFT algorithm is the natural way to discretize the Fourier transform along z, due to its favorable computational scaling ($\propto N_z \log(N_z)$). On the other hand, there are a variety of existing algorithms (that are not mathematically equivalent) to discretize the Hankel transform (e.g. [**?** **?** **?** **?** **?**]), and the use of one or the other is very dependent on the application pursued. Broadly speaking, choosing an algorithm for the Discrete Hankel Transform (DHT) algorithms consists in:

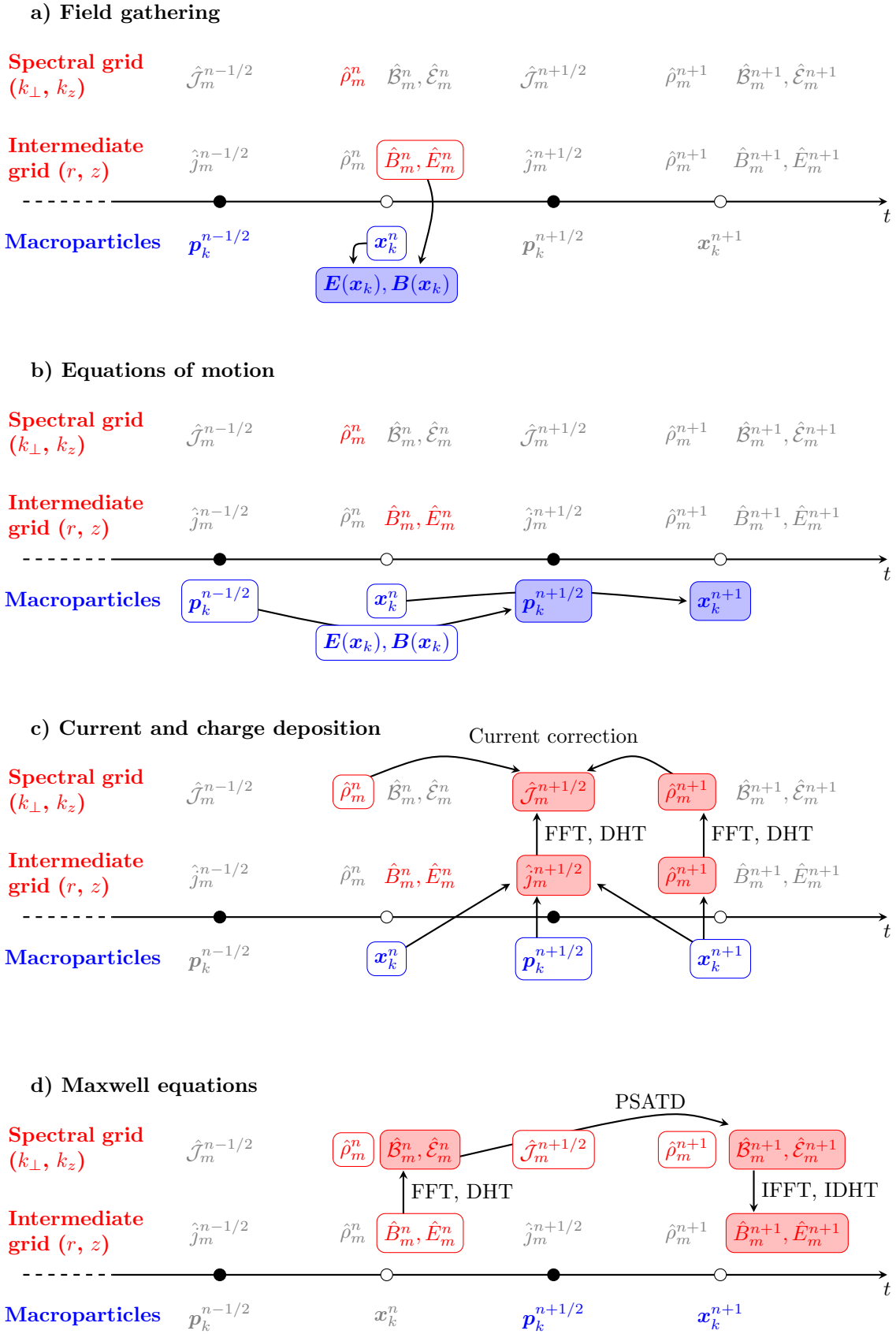

Figure 1: Schematic description of the 4 steps of a PIC cycle. At any given time, the quantities that are known are shown in color (red and blue), while the quantities that are unknown or have been erased from memory are shown in gray. The quantities that are being calculated at a given step are displayed with a colored background, and arrows indicate which quantities are used for this calculation.

- choosing a discrete grid in r and $k_\perp$ space, on which to sample the functions to be transformed. In some algorithms, these grids may not be evenly-spaced, and can be for instance logarithmically spaced [**?**] or can correspond to the zeros of Bessel functions [**? ? ?**],
- once a grid is chosen, the DHT amounts to a linear operation on a finite set of points (the points of the grid) and can thus be represented by a matrix. Thus the second choice is that of a matrix that represents, as closely as possible, the exact Hankel Transform.

Here, we choose to discretize the algorithm on an evenly-spaced grid in r (for the intermediate grid) :

$$r_j = \Delta r \left(j + \frac{1}{2} \right) \qquad j \in \{0, ..., N_r - 1\} \qquad \text{where} \quad \Delta r = \frac{r_{max}}{N_r} \tag{17}$$

but on an irregular grid in $k_\perp$ (for the spectral grid). This is because, for the chosen boundary condition ($\boldsymbol{E}(r_{max}) = \boldsymbol{0}$, $\boldsymbol{B}(r_{max}) = \boldsymbol{0}$), the fields can be expressed as a discrete sum of Bessel modes J_m with

$$k^m_{\perp,j} = \frac{\alpha^m_j}{r_{max}} \qquad j \in \{0, ..., N_r - 1\} \tag{18}$$

where α^m_j is the jth positive zero of the Bessel function of order m J_m (including the trivial value $\alpha^m_0 = 0$ for $m > 0$; see e.g. [**?**]). These values are represented in figure **??**. Notice that it is not an issue that the spectral components $\hat{\mathcal{F}}_m$ for different azimuthal modes m are discretized on different $k_\perp$ grids, since each azimuthal mode m evolves separately in **??????**.

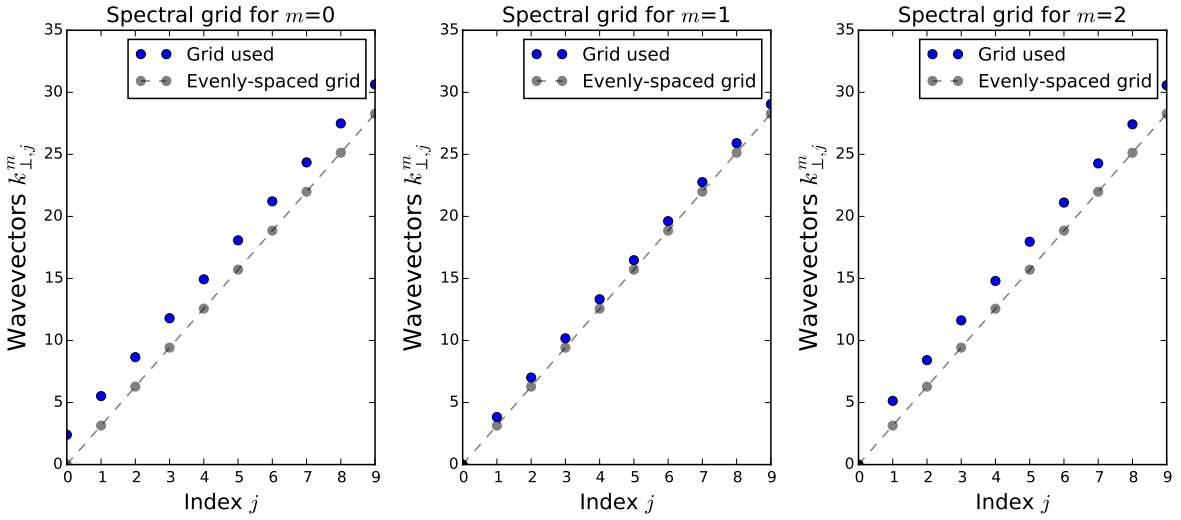

Figure 2: Position of the grid points in $k_\perp$ space (blue dots) for the azimuthal modes $m = 0$, $m = 1$ and $m = 2$ for $N_r = 10$ and $r_{max} = 1$. These values are compared with those of an evenly-spaced grid used in spectral Cartesian codes (grey dots, $k_j = j\pi/r_{max}$).

As mentioned above, once this grid is set up, the Discrete Hankel Transform is simply a linear operation on a finite set of points, and can thus be represented by a matrix operation.

$$\mathrm{DHT}^{\mathrm{m}}_{\mathrm{n}}[f]\,(k^m_{\perp,j}) = \sum_{p=0}^{N_r-1} (M_{n,m})_{j,p}\; f(r_p) \qquad \mathrm{IDHT}^{\mathrm{m}}_{\mathrm{n}}[g]\,(r_j) = \sum_{p=0}^{N_r-1} (M'_{n,m})_{j,p}\; g(k^m_{\perp,p}) \tag{19}$$

Notice that the $N_r \times N_r$ transformation matrices $M_{n,m}$ and $M'_{n,m}$ depend on n (order of the Hankel transform, i.e. order of the Bessel function J_n in the integrand of **??**) and m (index of the azimuthal mode, and thus index of the spectral grid $k^m_{\perp,j}$ on which the Hankel transform is performed). In practice, n and m are either equal or they differ by ± 1 (e.g. in **????**, when the azimuthal mode m is transformed using the Hankel transform of order $m + 1$ or $m - 1$.)

The expressions of $M_{n,m}$ and $M'_{n,m}$, for the DHT algorithm that we chose, are given in **??**. In practice, these matrices need to be computed only once (at the beginning of the simulation) and can then be used at each iteration. Notice also that the computational time for this matrix multiplication is proportional to N_r^2, which is slow compared to a 1D FFT ($\propto N_r \log(N_r)$), but still faster than the 2D FFT ($\propto N_x N_y \log(N_x) + N_x N_y \log(N_y)$) which is typically used in the transverse plane of a spectral 3D Cartesian code.

2.3. Field gathering

In the following, we describe the four successive steps of a PIC cycle with our algorithm in detail, starting with the field gathering. When gathering the fields from the intermediate grid to the macroparticles (see **??**), we use the standard linear shape factors :

$$\begin{aligned} F_u(\boldsymbol{r}_k) &= \sum_{p,q} S_{z,p}(z_k) S_{r,q}(r_k) \left[\sum_{m=-N_m}^{N_m} \hat{F}_{u,m}(z_p, r_q) e^{-im\theta_k} \right] \\ &= \sum_{p,q} S_{z,p}(z_k) S_{r,q}(r_k) \left[\hat{F}_{u,0}(z_p, r_q) + 2\,\Re \left(\sum_{m=1}^{N_m} \hat{F}_{u,m}(z_p, r_q) e^{-im\theta_k} \right) \right] \end{aligned} \tag{20}$$

where F is either E or B, u is either r, θ or z, k is the index of the macroparticle, and p and q are the indices of the two nearest cells in z and r respectively. N_m is the total number of azimuthal modes used, and $\Re$ denotes the real part of a complex quantity. Notice that, in **??**, we used the fact that $\hat{F}_{u,-m}(z,r) = \hat{F}^*_{u,m}(z,r)$, which can be inferred from **??**. Incidentally, **??** shows that only the modes with $m \geq 0$ need to be taken into account in the code, as they are sufficient to retrieve the force on the macroparticles.

Finally, in **??**, $S_{z,p}$ and $S_{z,q}$ are the linear shape factors in z and r :

$$S_{z,p}(z) = \frac{z_{p+1} - z}{\Delta z} \qquad S_{z,p+1}(z) = \frac{z - z_p}{\Delta z} \qquad \text{with} \quad z_p \leq z < z_{p+1} \tag{21a}$$

$$S_{r,q}(r) = \frac{r_{q+1} - r}{\Delta r} \qquad S_{r,q+1}(r) = \frac{r - r_q}{\Delta r} \qquad \text{with} \quad r_q \leq r < r_{q+1} \tag{21b}$$

For particles that are in the lower half of the first radial cell ($r < r_0 = \Delta r/2$), **????** require the value $\hat{F}_{u,m}(z_p, r_{-1})$ although $r_{-1} = -\Delta r/2$ is not actually part of the grid. In order to still apply **????**, we explicitly set $\hat{F}_{u,m}(z_p, -\Delta r/2) = \pm \hat{F}_{u,m}(z_p, \Delta r/2)$, where the $-$ sign is chosen whenever the considered field $\hat{F}_{u,m}$ is by definition zero on the axis, and the $+$ sign is chosen otherwise. (e.g. $\hat{E}_{r,0}$ is by definition zero on the axis ; see [**?**] for more details, and for a similar method.)

Note that we use linear shape factors here only for the sake of simplicity, and that higher-order shape factors (e.g. quadratic or cubic) could also be used in principle, with a similar mirroring of the field values across the axis.

2.4. Equations of motion

Since the macroparticles evolve in 3D, we first compute the Cartesian components E_x, E_y, E_z, B_x, B_y and B_z, from the fields E_r, E_θ, E_z, B_r, B_θ and B_z that were gathered at the positions of each macroparticle. We then advance the equations of motion

$$\frac{d\boldsymbol{p}}{dt} = q\boldsymbol{E} + q\boldsymbol{v} \times \boldsymbol{B} \qquad \frac{d\boldsymbol{x}}{dt} = \frac{\boldsymbol{p}}{\gamma\, m} \tag{22}$$

in standard 3D Cartesian coordinates by using the leap-frog pusher described in [**?**].

2.5. Current deposition

As in [**?**], the charge density is calculated on the intermediate grid in the following way:

$$\hat{\rho}_m(z_p, r_q) = \frac{\sum_k S_{z,p}(z_k) S_{r,q}(r_k) Q_k e^{im\theta_k}}{V_q} \tag{23}$$

where Q_k is the charge of the macroparticle with index k, and where V_q is the volume of a cell, which is for our grid

$$V_q = \pi[\,(q+1)^2 - q^2\,]\Delta r^2 \Delta z \tag{24}$$

Similarly, the current deposition is given by

$$\hat{j}_{u,m}(z_p, r_q) = \frac{\sum_k S_{z,p}(z_k) S_{r,q}(r_k) Q_k v_{u,k} e^{im\theta_k}}{V_q} \tag{25}$$

where $u = z, r, \theta$ and the $v_{u,k}$ are the cylindrical components of the velocity of the macroparticle k. As mentioned previously, once the macroparticles have deposited their charge and current on the intermediate grid, we transform them to the spectral grid, using an FFT and a DHT (see **??**).

Note that, when a macroparticle travels through the grid, the factor V_q in the above formulas decreases as the macroparticle comes closer to the axis. Therefore, close to the axis, the impact of a single macroparticle on an individual grid cell can be substantial and this generally translates into higher levels of noise. (This is a general problem with cylindrical and quasi-cylindrical codes, and thus it is not specific to the present spectral version.) In order to mitigate this problem, and more generally in order to avoid the accumulation of noise at high frequency, smoothing is typically applied on the charge and currents, after they have been deposited (but no smoothing is applied on the fields E and B directly). This smoothing is performed directly in spectral space, by multiplying the charge and currents by a transfer function $\hat{\mathcal{T}}(k_z, k_\perp)$ which damps the high frequencies $\boldsymbol{k}$, and whose mathematical form is identical to the spectral Cartesian representation of a single-pass binomial filter [**?**].

$$\hat{\mathcal{T}}(k_z, k_r) = \cos^2\left(\frac{k_z}{k_{z,max}}\frac{\pi}{2}\right)\cos^2\left(\frac{k_\perp}{k_{\perp,max}}\frac{\pi}{2}\right) \tag{26}$$

where $k_{z,max}$ and $k_{\perp,max}$ are the highest wavevectors that the discrete spectral grid supports, in the longitudinal and transverse direction.

Notice also that, in the above deposition scheme, we do not attempt to reproduce the Esirkepov charge-conserving current deposition [**?**], and instead use a simple direct current deposition. It is well-known that this simple deposition does not necessarily satisfy the relation $\partial_t \rho + \boldsymbol{\nabla}\cdot\boldsymbol{j} = 0$, but that this can be corrected, by slightly modifying the currents without modifying their curl (e.g. [**?**]):

$$\boldsymbol{j}' = \boldsymbol{j} - \boldsymbol{\nabla} G \tag{27}$$

where G satisfies the Poisson-like equation

$$\boldsymbol{\nabla}^2 G = \partial_t \rho + \boldsymbol{\nabla}\cdot\boldsymbol{j} \tag{28}$$

The above equation is typically expensive to solve on a spatial grid, but very easy to solve in spectral space. In spectral space and with the notations of **??**, these equations become

$$\hat{\mathcal{J}}'^{n+1/2}_{+,m} = \hat{\mathcal{J}}^{n+1/2}_{+,m} + \frac{k_\perp}{2}\hat{\mathcal{G}}^{n+1/2}_m \qquad \hat{\mathcal{J}}'^{n+1/2}_{-,m} = \hat{\mathcal{J}}^{n+1/2}_{-,m} - \frac{k_\perp}{2}\hat{\mathcal{G}}^{n+1/2}_m \qquad \hat{\mathcal{J}}'^{n+1/2}_{z,m} = \hat{\mathcal{J}}^{n+1/2}_{z,m} - ik_z\hat{\mathcal{G}}^{n+1/2}_m \tag{29}$$

with

$$\hat{\mathcal{G}}^{n+1/2}_m = -\frac{1}{k_\perp^2 + k_z^2}\left(\frac{\hat{\rho}^{n+1}_m - \hat{\rho}^n_m}{\Delta t} + k_\perp(\hat{\mathcal{J}}^{n+1/2}_{+,m} - \hat{\mathcal{J}}^{n+1/2}_{-,m}) + ik_z\hat{\mathcal{J}}^{n+1/2}_{z,m}\right) \tag{30}$$

With this correction, the new currents $\hat{\mathcal{J}}'^{n+1/2}$ do satisfy the charge conservation equation **??**. Therefore we apply this correction in spectral space, at the end of the current deposition at each timestep.

2.6. Integration of the Maxwell equation using the PSATD scheme

The Maxwell equations **??????** could in principle be integrated by using a finite-difference scheme in time, which would be an adaptation of the Cartesian PSTD scheme [**?**].

$$\hat{\mathcal{B}}^{n+1/2}_{+,m} = \hat{\mathcal{B}}^n_{+,m} - \frac{\Delta t}{2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^n_{z,m} + k_z\hat{\mathcal{E}}^n_{+,m}\right) \tag{31a}$$

$$\hat{\mathcal{B}}^{n+1/2}_{-,m} = \hat{\mathcal{B}}^n_{-,m} - \frac{\Delta t}{2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^n_{z,m} - k_z\hat{\mathcal{E}}^n_{-,m}\right) \tag{31b}$$

$$\hat{\mathcal{B}}^{n+1/2}_{z,m} = \hat{\mathcal{B}}^n_{z,m} - \frac{\Delta t}{2}\left(ik_\perp\hat{\mathcal{E}}^n_{+,m} + ik_\perp\hat{\mathcal{E}}^n_{-,m}\right) \tag{31c}$$

$$\hat{\mathcal{E}}^{n+1}_{+,m} = \hat{\mathcal{E}}^n_{+,m} + c^2\Delta t\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}^{n+1/2}_{z,m} + k_z\hat{\mathcal{B}}^{n+1/2}_{+,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{+,m}\right) \tag{32a}$$

$$\hat{\mathcal{E}}^{n+1}_{-,m} = \hat{\mathcal{E}}^n_{-,m} + c^2\Delta t\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}^{n+1/2}_{z,m} - k_z\hat{\mathcal{B}}^{n+1/2}_{-,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{-,m}\right) \tag{32b}$$

$$\hat{\mathcal{E}}^{n+1}_{z,m} = \hat{\mathcal{E}}^n_{z,m} + c^2\Delta t\left(ik_\perp\hat{\mathcal{B}}^{n+1/2}_{+,m} + ik_\perp\hat{\mathcal{B}}^{n+1/2}_{-,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{z,m}\right) \tag{32c}$$

$$\hat{\mathcal{B}}^{n+1}_{+,m} = \hat{\mathcal{B}}^{n+1/2}_{+,m} - \frac{\Delta t}{2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^{n+1}_{z,m} + k_z\hat{\mathcal{E}}^{n+1}_{+,m}\right) \tag{33a}$$

$$\hat{\mathcal{B}}^{n+1}_{-,m} = \hat{\mathcal{B}}^{n+1/2}_{-,m} - \frac{\Delta t}{2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^{n+1}_{z,m} - k_z\hat{\mathcal{E}}^{n+1}_{-,m}\right) \tag{33b}$$

$$\hat{\mathcal{B}}^{n+1}_{z,m} = \hat{\mathcal{B}}^{n+1/2}_{z,m} - \frac{\Delta t}{2}\left(ik_\perp\hat{\mathcal{E}}^{n+1}_{+,m} + ik_\perp\hat{\mathcal{E}}^{n+1}_{-,m}\right) \tag{33c}$$

However, this type of scheme retains some amount of spurious numerical dispersion, and can thus affect the simulated physics.

Instead, here we use an adaptation of the Cartesian PSATD scheme [**?**] for our spectral quasi-cylindrical representation. As in the case of the standard PSATD, we assume that the currents are constant over one timestep and that the charge density is linear in time over the same timestep. Under these assumptions, the Maxwell equations **??????** can be integrated analytically over that timestep, and they lead to:

$$\hat{\mathcal{E}}^{n+1}_{+,m} = C\hat{\mathcal{E}}^{n}_{+,m} + c^2\frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}^{n}_{z,m} + k_z\hat{\mathcal{B}}^{n}_{+,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{+,m}\right) + \frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\left[\frac{\hat{\rho}^{n+1}_m}{\omega^2}\left(1-\frac{S}{\omega\Delta t}\right) - \frac{\hat{\rho}^{n}_m}{\omega^2}\left(C-\frac{S}{\omega\Delta t}\right)\right] \tag{34a}$$

$$\hat{\mathcal{E}}^{n+1}_{-,m} = C\hat{\mathcal{E}}^{n}_{-,m} + c^2\frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}^{n}_{z,m} - k_z\hat{\mathcal{B}}^{n}_{-,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{-,m}\right) - \frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\left[\frac{\hat{\rho}^{n+1}_m}{\omega^2}\left(1-\frac{S}{\omega\Delta t}\right) - \frac{\hat{\rho}^{n}_m}{\omega^2}\left(C-\frac{S}{\omega\Delta t}\right)\right] \tag{34b}$$

$$\hat{\mathcal{E}}^{n+1}_{z,m} = C\hat{\mathcal{E}}^{n}_{z,m} + c^2\frac{S}{\omega}\left(ik_\perp\hat{\mathcal{B}}^{n}_{+,m} + ik_\perp\hat{\mathcal{B}}^{n}_{-,m} - \mu_0\hat{\mathcal{J}}^{n+1/2}_{z,m}\right) - \frac{c^2}{\epsilon_0}ik_z\left[\frac{\hat{\rho}^{n+1}_m}{\omega^2}\left(1-\frac{S}{\omega\Delta t}\right) - \frac{\hat{\rho}^{n}_m}{\omega^2}\left(C-\frac{S}{\omega\Delta t}\right)\right] \tag{34c}$$

$$\hat{\mathcal{B}}^{n+1}_{+,m} = C\hat{\mathcal{B}}^{n}_{+,m} - \frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^{n}_{z,m} + k_z\hat{\mathcal{E}}^{n}_{+,m}\right) + \mu_0c^2\frac{1-C}{\omega^2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}^{n+1/2}_{z,m} + k_z\hat{\mathcal{J}}^{n+1/2}_{+,m}\right) \tag{35a}$$

$$\hat{\mathcal{B}}^{n+1}_{-,m} = C\hat{\mathcal{B}}^{n}_{-,m} - \frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}^{n}_{z,m} - k_z\hat{\mathcal{E}}^{n}_{-,m}\right) + \mu_0c^2\frac{1-C}{\omega^2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}^{n+1/2}_{z,m} - k_z\hat{\mathcal{J}}^{n+1/2}_{-,m}\right) \tag{35b}$$

$$\hat{\mathcal{B}}^{n+1}_{z,m} = C\hat{\mathcal{B}}^{n}_{z,m} - \frac{S}{\omega}\left(ik_\perp\hat{\mathcal{E}}^{n}_{+,m} + ik_\perp\hat{\mathcal{E}}^{n}_{-,m}\right) + \mu_0c^2\frac{1-C}{\omega^2}\left(ik_\perp\hat{\mathcal{J}}^{n+1/2}_{+,m} + ik_\perp\hat{\mathcal{J}}^{n+1/2}_{-,m}\right) \tag{35c}$$

where $\omega \equiv c\sqrt{k_z^2+k_\perp^2}$, $C \equiv \cos(\omega\Delta t)$ and $S \equiv \sin(\omega\Delta t)$. (See **??** for a derivation of these equations.)

2.7. Practical implementation

The full PIC algorithm described in this section was implemented in the code FBPIC (Fourier-Bessel Particle-In-Cell), which is written in Python. For performance, this implementation makes use of the pre-compiled libraries FFTW [**?**] and BLAS [**?**] (for the matrix multiplication in the DHT), and utilizes the Numba just-in-time compiler [**?**] for the computationally-intensive parts of the code (current deposition and field gathering). In addition, the code was developed for both single-CPU and single-GPU architectures (with the GPU runs being typically more than 40 times faster than the equivalent CPU runs, on modern hardware). Importantly, a precise timing of the different routines showed that, although the time taken by the spectral transforms (FFT and DHT) is not entirely negligible, it does not usually dominate the PIC cycle. For instance, on a K20 GPU and for a grid with $N_z = 4096$, $N_r = 256$ and 16 particles per cell, the FFTs and DHTs take up 6% and 14% of the PIC cycle respectively, while the rest of the time is dominated by the field gathering and current deposition.

Notice that, even though parallelization is generally challenging for spectral algorithms, a multi-CPU/multi-GPU version could still be developped in the future by using the method of [**?**]. For the present algorithm, this would involve a domain decomposition along the z axis, whereby FFTs would be performed locally within each subdomain and whereby a large number of guard cells would be used in order to mitigate the errors at the border between subdomains. This method based on local FFTs has been studied in the Cartesian context [**?**], and is now rather well understood. By contrast, performing domain decomposition in the r direction would be more challenging, due to the absence of previous work on local Hankel transforms. At any rate, the above considerations for parallel implementation are out of the scope of the present article, and will be the subject of future work. The benchmarks described in the following section use a single-CPU/single-GPU version of the algorithm.

3. Benchmarks

We tested the above algorithm in a number of physical situations. These tests included standard problems, such as e.g. a periodic plasma wave, and involved comparing simulated results to analytical solution, as well as verifying that the algorithm conserves the energy to a satisfying level. For the sake of conciseness, in the present article, we restrict the discussion to the tests of the dispersion relation, since the absence of spurious numerical dispersion was one of the original goals of this algorithm.

3.1. Propagation in vacuum

A first test consisted in letting a laser pulse propagate in vacuum and in measuring its group velocity in the simulation. These simulations were run with the PSATD quasi-cylindrical algorithm presented in **??** (see esp. **????**), but also, for comparison, with a PSTD version of this same algorithm (see **??????**), as well as with a finite-difference quasi-cylindrical algorithm equivalent to that of [**?** **?**] and which had been previously implemented in the PIC code WARP [**?**].

The simulations were run with a moving window, in a box with a longitudinal size of 40 μm and a transverse size of 48 μm. (In the case of the spectral algorithm, the fields were damped at the back of the moving window, in a similar way as in [**?**], in order to prevent periodic wrapping of the fields). The laser pulse itself was initialized at focus, with a waist $w_0 = 16$ μm, a length $L = 10$ μm, a wavelength $\lambda = 0.8$ μm and a dimensionless potential vector $a_0 = 10^{-2}$. The resolution was varied while keeping the same cell aspect ratio ($\Delta r = 5\Delta z$). The timestep was set to $c\Delta t = \Delta z$ in the case of the PSATD algorithm, to $c\Delta t = 0.9 \times 2/\pi\sqrt{1/\Delta z^2 + 1/\Delta r^2}$ in the case of the PSTD algorithm, and to $c\Delta t = 1/\sqrt{1/\Delta z^2 + 2/\Delta r^2}$ in the case of the finite-difference algorithm (for the PSTD and finite-difference algorithms, the chosen Δt is close to the Courant limit).

Physically, the on-axis group velocity of the pulse should be slightly lower than c due to the finite waist of the pulse (see e.g. [**?**]). The analytical expression for the corresponding relative difference in on-axis group velocity is

$$\frac{c - v_g}{c} = 2\left(\frac{\lambda}{2\pi w_0}\right)^2 \tag{36}$$

In the case at hand ($w_0 = 16$ μm, $\lambda = 0.8$ μm), this relative difference is extremely small ($(c - v_g)/c = 1.27 \times 10^{-4}$), and it may be difficult for PIC codes to capture this small physical difference.

To assess the capacities of the finite-difference and spectral codes in this regard, **??** displays the relative difference in group velocity, as measured in the simulations. As can be observed, in the finite-difference simulations and PSTD simulations, the group velocity of the laser depends on the resolution, due to spurious numerical dispersion. Moreover, in these cases, the group velocity is considerably different than the analytical prediction, even for a relatively high resolution. (Since **??** does not display the sign of $c - v_g$, it is worth noting here that the PSTD algorithm leads to $v_g > c$ while the finite-difference algorithm has $v_g < c$. It is also important to realize here that, although the finite-difference algorithm performs better than the PSTD algorithm in this particular example, this is not generalizable to other cases, as e.g. the numerical dispersion of both algorithms behave differently when Δt or the ratio $\Delta z/\Delta r$ is changed.)

On the other hand, with the PSATD algorithm, the group velocity is practically independent of the resolution, and displays very good agreement with the analytical prediction. This corroborates the fact that the PSATD quasi-cylindrical algorithm described here has no spurious numerical dispersion in vacuum.

Since the PSTD algorithm is less acurate than the PSATD algorithm, while not providing any substantial advantage in terms of speed or practical implementation, we do not consider it further here and perform the rest of the tests with the PSATD algorithm.

3.2. Linear propagation in a plasma

In order to confirm that the spectral algorithm also performs well in the presence of a plasma, we ran the same type of simulations with a uniform, pre-ionized plasma. The numerical parameters of the simulations, as well as the physical parameters of the laser, were the same as in the previous subsection (**??**). The plasma was represented by 16 macroparticles per cell, and had a density $n_e = 1.75 \times 10^{18}$ cm$^{-3} = 10^{-3}\, n_c$, where n_c is the critical density for $\lambda = 0.8$ μm.

Since the intensity of the laser is very low here ($a_0 = 10^{-2}$), the propagation is linear, and the on-axis group velocity is given by (e.g. [**?**])

$$\frac{c - v_g}{c} = \frac{n_e}{2n_c} + 2\left(\frac{\lambda}{2\pi w_0}\right)^2 \tag{37}$$

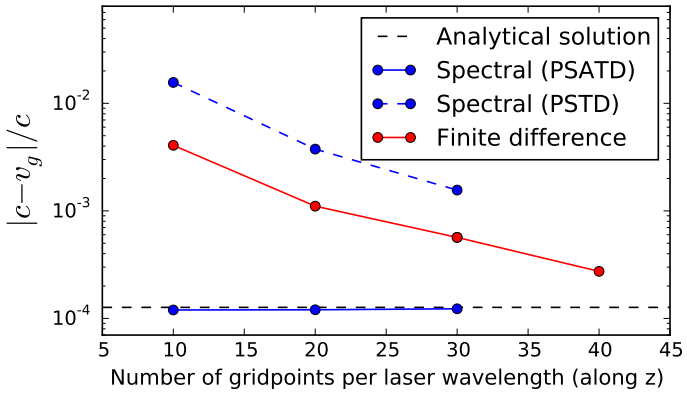

Figure 3: Relative difference between c and the group velocity of a laser pulse in vacuum, for different resolutions. The dashed black line represents the analytical prediction given by **??**, while the blue and red points represent the results of PIC simulation, with the PSATD, PSTD and finite-difference algorithm.

In **??**, we compare this analytical prediction with the group velocity in the finite-difference and spectral simulations. Again, the group velocity is resolution-dependent in the finite-difference algorithm, and it is substantially different than the analytical prediction even at high resolution. In the spectral code, the group velocity velocity exhibits a very weak dependence on resolution (which is likely due to the errors of current deposition and field gathering on the finite grid). However, its value remains always very close to the analytical prediction at all resolutions.

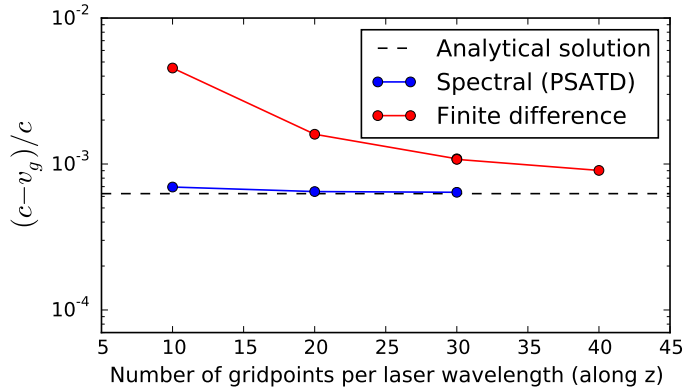

Figure 4: Relative difference between c and the group velocity of a laser pulse in a plasma at $10^{-3}\,n_c$, for different resolutions. The dashed line represents the analytical prediction given by **??**, while the blue and red points represent the results of PIC simulations.

We emphasize that, although the difference between c and v_g is small here and may thus seem unimportant, it is this difference which determines the dephasing length and thus the maximum beam energy, in a laser-wakefield simulation. It is therefore paramount to obtain its correct value in the simulation codes. This problem is well-known in the case of standard finite-difference codes. For Cartesian finite-difference codes, a common solution is to use a scheme which is dispersion-free along the z axis (e.g. [**?** **?** **?**]), but no such scheme has been developed for quasi-cylindrical codes. Alternatively, spurious numerical dispersion is often dealt with in finite-difference codes by either using an even finer grid in z (which is very computationally expensive) or a *coarser* grid in r. (Recall that, in the simulations shown here, $\Delta r = 5\Delta z$. A coarser resolution in r allows to use a slightly larger timestep Δt that approaches $c\Delta z$, the limit at which spurious numerical dispersion vanishes in the z-direction.) However, a coarser resolution in r may not always be adapted to resolve the physics of interest, while a finer resolution in z is expensive. It is therefore remarkable that, with the spectral quasi-cylindrical algorithm, the correct group velocity is obtained independently of the cell aspect ratio, without having to specifically adapt the resolution in r or z.

3.3. Linear laser-wakefield

In order to further ascertain that the spectral algorithm described here gives appropriate results beyond simple dispersion tests, we ran a simulation of a laser-wakefield, and compared the amplitude of the wakefield with the corresponding analytical predictions.

In these simulations, the laser was linearly polarized along the transverse x direction, and had an amplitude $a_0 = 10^{-2}$, a wavelength $\lambda = 0.8$ μm, a length $L = 10$ μm and a waist $w_0 = 20$ μm. The plasma was preionized, with a uniform electron density $n_e = 1.75 \times 10^{18}$ cm$^{-3} = 10^{-3}\, n_c$. As is often the case for simulations of intense laser-plasma interaction (where thermal effects are generally assumed to be negligible), the initial temperature of the plasma is set to 0. The simulation was run in a moving window whose longitudinal and transverse sizes were 80 μm and 60 μm. The spatial and temporal resolutions were $\Delta z = 0.05$ μm, $\Delta r = 0.5$ μm and $\Delta t = \Delta z/c$. Since here $a_0 = 10^{-2}$, the analytical wakefield is given by the linear and quasistatic theory. For a laser of the form $a(z,r,t) = a_\ell(z,t)e^{-r^2/w_0^2}\cos(k_0 z - \omega_0 t)$ – where $a_\ell(z,t)$ is the longitudinal envelope of the laser – the longitudinal and transverse fields E_z and E_y are given by (e.g. [**?**])

$$E_z(z,r,t) = \frac{mc^2}{e}\frac{k_p^2}{4}\int_z^\infty a_\ell^2(z',t)e^{-r^2/w_0^2}\cos[k_p(z-z')]dz' \tag{38a}$$

$$E_y(z,r,t) = -\frac{mc^2}{e}\frac{k_p y}{w_0^2}\int_z^\infty a_\ell^2(z',t)e^{-r^2/w_0^2}\sin[k_p(z-z')]dz' \tag{38b}$$

where k_p is the plasma wavevector.

Here, the longitudinal laser envelope a_ℓ was extracted directly from the simulation, and the above analytical integrals where carried out numerically. The resulting predicted fields E_z and E_y are plotted in dashed lines in the left and right lower panels respectively of **??**. These predicted curves are compared with the fields E_z and E_y extracted directly from the simulation (red lines in the lower panels). These fields from the simulation are also displayed as colormaps in the upper panels of **??**, and these colormaps include the line along which the quantities in the lower panels are plotted (dashed line).

As can be seen on the lower panels of **??**, the analytical and simulated curves overlap very precisely, which confirms the validity of our PIC algorithm. This agreement is all the more remarkable as the E_z field has been extracted from the cells that are closest to the axis, a region where quasi-cylindrical algorithms are typically very noisy.

4. Advantages over finite-difference algorithms

Since finite-difference algorithms and spectral algorithms are considerably different, each of them has its own advantages and shortcomings.

For instance, spectral algorithms – including the one described here – are generally more difficult to parallelize than finite-difference codes. Similarly, the implementation of specific boundary conditions, and of a moving window, is usually more challenging in a spectral algorithm. On the other hand, due to their high accuracy, spectral algorithms can avoid a number of numerical artifacts that are typically present in finite-difference algorithms. This is quite important as, in a finite-difference simulation, these artifacts may remain unnoticed unless additional care is taken in their analysis, and yet they may affect the physics at stake in an important way.

One example of such an artifact is spurious numerical dispersion, which, as mentioned in [**?**] and in **??**, can modify the dephasing length of a laser-wakefield accelerator – a spurious effect which may be difficult to discern, especially in the cases where there is no analytical formula for this length. As shown in **??**, the algorithm described here would not suffer from this effect, as it is free of spurious numerical dispersion. Similarly, in this section we describe two other typical situations in which our spectral quasi-cylindrical algorithm avoids important artifacts, that would otherwise arise in a finite-difference code.

4.1. Suppression of zero-order numerical Cherenkov effect

A first artifact which is avoided is the zero-order numerical Cherenkov effect. To zero order, the numerical Cherenkov effect is a consequence of the spurious numerical dispersion [**?**], and arises because some relativistic particles can travel faster than the numerically-altered velocity of the electromagnetic waves, in the simulations. This typically causes these relativistic particles to emit a characteristic spurious radiation. Recently, it was shown that this spurious radiation can have a very substantial impact in lab-frame simulations of laser-wakefield acceleration, in particular by spuriously increasing the emittance of

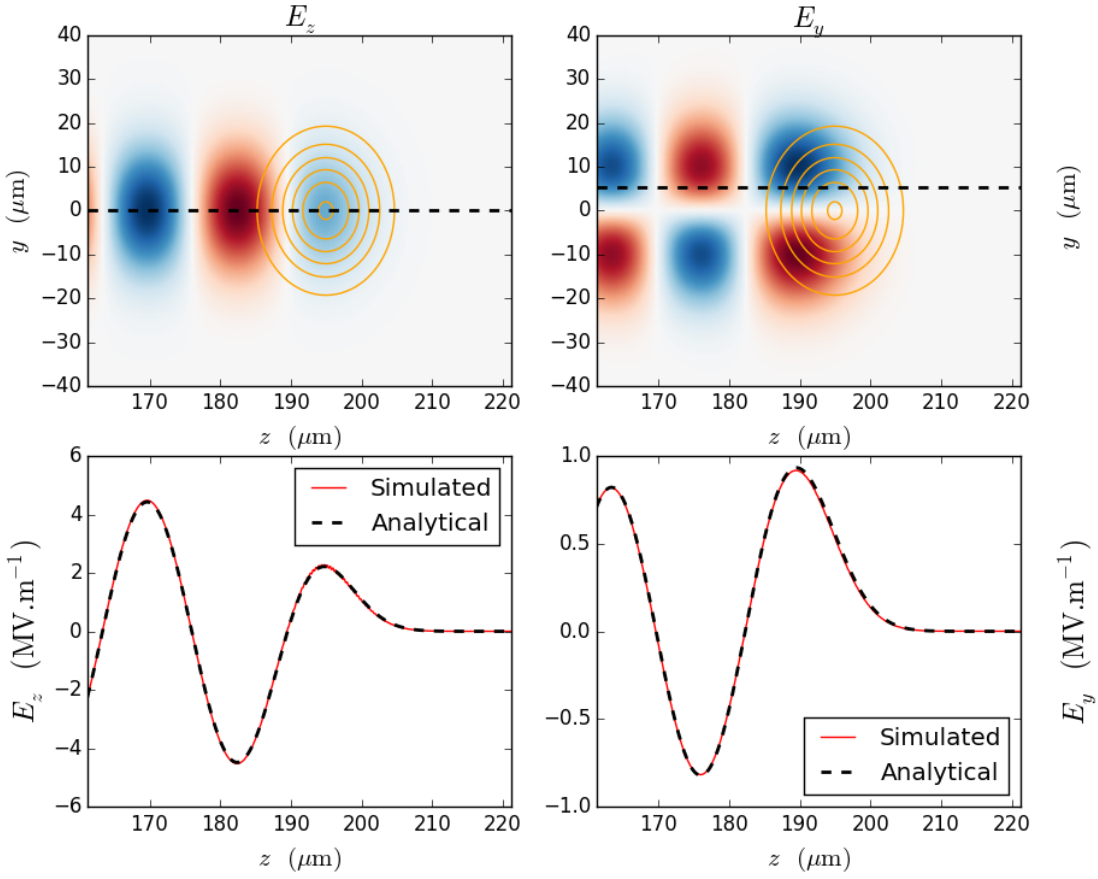

Figure 5: Upper panels: Colormaps of the fields E_z and E_y as extracted from the simulation (blue and red), along with the laser envelope (orange contour lines). (The laser propagates to the right and is polarized along x.) The dashed lines indicate the position where the curves of the lower panels have been extracted. Lower panels: profile of the fields at a given radial position, as given by **????** (dashed line) or as given by the simulation (red line).

the accelerated beam [**?**]. It was also shown that, by modifying the PIC algorithm and its numerical dispersion relation, the zero-order Cherenkov radiation can be suppressed. (Notice that there exists also a set of higher-order, *aliased* numerical Cherenkov effects, which are not as easily suppressed [**? ? ? ? ? ? ? ?**]. These high-order effects can lead to disruptive instabilities in boosted-frame simulations, but on the other hand no such instability was observed in typical lab-frame simulations of laser-plasma acceleration. It is in fact likely that these instabilities may not have enough time to develop in the case of typical lab-frame simulations.)

Since the spectral quasi-cylindrical algorithm described here is dispersion-free, it should not exhibit the zero-order numerical Cherenkov effect. In order to confirm this prediction, we ran a simulation of laser-wakefield acceleration with our spectral quasi-cylindrical algorithm, and compared it with an equivalent simulation that uses the finite-difference quasi-cylindrical algorithm of WARP.

In these simulations, a laser pulse with a waist $w_0 = 16$ μm, a length $L = 10$ μm, an amplitude $a_0 = 4$ and a wavelength $\lambda = 0.8$ μm is sent into a longitudinally-shaped pre-ionized gas jet. The gas jet has a 100 μm-long rising density gradient at its entrance, followed by a 200 μm plateau at a density $n_e = 1 \times 10^{18}$ cm^{-3} and then by a 100 μm-long downramp, so as to finally reach a density $n_e = 0.5 \times 10^{18}$ cm^{-3}. Again, the initial temperature of the plasma is set to 0. The downramp causes the injection of an electron beam, which is then accelerated in the subsequent density plateau at $n_e = 0.5 \times 10^{18}$ cm^{-3}. Both the spectral and the finite-difference simulations were run in a moving window whose longitudinal and transverse dimensions were 160 μm and 48 μm, and both simulations used a resolution $\Delta z = 0.032$ μm and $\Delta r = 0.19$ μm. Again, the finite-difference simulation was run with a timestep $c\Delta t = 1/\sqrt{1/\Delta z^2 + 2/\Delta r^2}$ while the spectral simulation was run with $c\Delta t = \Delta z$. Importantly, the charge and currents are smoothed in both simulations. The spectral algorithms performs smoothing

in spectral space as described in **??**, while the finite-difference algorithm applies a single-pass binomial filter on the spatial grid, in both the z and r directions.

?? shows a snapshot of the two simulations, at a similar physical time. The red colormap represents the quantity $|E_y + cB_x|$, while the superimposed shades of blue represent the electron density. (The quantity $E_y + cB_x$ is chosen because it corresponds to the y component of the Lorentz force $\boldsymbol{F} = q\boldsymbol{E} + q\boldsymbol{v} \times \boldsymbol{B}$ felt by a relativistic electron having $\boldsymbol{v} = c\boldsymbol{e_z}$.) As can be seen in the top panels, the global aspect of the bubble and of the injected bunch is similar in both simulations. In particular, the injected charge was found to be almost the same (766 pC with the finite-difference algorithm and 750 pC with the spectral algorithm). However, a closer look at the bunch in the finite-difference algorithm (see the middle left panel in **??**) reveals that the bunch emits a high-frequency radiation. This radiation seems to be due to the zero-order numerical Cherenkov effect – an interpretation which is confirmed by the observation of the corresponding characteristic double-parabola in $\boldsymbol{k}$ space [**?** **?** **?**] on the lower left panel of **??**. Importantly, the amplitude of this unphysical field, in the middle panel of **??**, happens to be comparable to the focusing fields inside the bubble, and it can thus potentially affect the bunch in a substantial manner. On the other hand, the spectral algorithm does not exhibit this unphysical radiation in real space (see the middle right panel in **??**), nor does it exhibit the corresponding characteristic pattern in $\boldsymbol{k}$ space (see the lower right panel). This was indeed expected from its dispersion-free property, and it confirms the fact that our spectral quasi-cylindrical algorithm is free of the unphysical artifacts associated with the zero-order numerical numerical Cherenkov effect.

For completeness, we remark that a similar suppression of the numerical Cherenkov radiation was obtained in [**?** **?**], by using modified finite-difference algorithms. However, it is important to note that these algorithms were applicable to a Cartesian PIC code, but not to a quasi-cylindrical code. An adaptation to cylindrical geometry is proposed in [**?**], and was shown to efficiently suppress zero-order numerical Cherenkov effect. However, this adaptation is not, strictly speaking, dispersion-free.

4.2. Accurate force of a laser on a copropagating electron

Another case in which our spectral algorithm performs better than finite-difference algorithms is whenever a relativistic electron bunch overlaps with a copropagating laser. This situation occurs for instance in laser-wakefield acceleration, when the accelerated electron bunch progressively catches up with the laser pulse and may eventually overlap with it (e.g. [**?** **?**]). It is also the typical configuration for simulations of free-electron lasers, where the overlap of the electron bunch and the copropagating laser radiation, inside an undulator, leads to a growing instability.

Despite the practical importance of this physical configuration, it was recently shown that standard finite-difference PIC codes tend to largely overestimate the force felt by the electrons inside a copropagating laser (see the appendix of [**?**]). This overestimation was shown to be mainly due to the staggering in time of the $\boldsymbol{E}$ and $\boldsymbol{B}$ fields in a standard finite-difference code. This staggering indeed results in an improper numerical compensation of the two terms of the Lorentz force $\boldsymbol{F} = q\boldsymbol{E} + q\boldsymbol{v} \times \boldsymbol{B}$. However, in our spectral quasi-cylindrical algorithm, the fields $\boldsymbol{E}$ and $\boldsymbol{B}$ are not staggered in time, and thus the force on a copropagating electron bunch should be correct.

To confirm this prediction, we ran a test simulation in which a single relativistic macroparticle copropagates with a laser pulse. The initial configuration of the simulation is represented in **??**. The laser pulse is polarized along x and is characterized by a waist $w_0 = 25$ μm, a length $L = 7$ μm, an amplitude $a_0 = 0.2$ and a wavelength $\lambda = 0.8$ μm, while the macroparticle represents a relativistic electron having an initial Lorentz factor $\gamma_e = 25$. For comparison, the simulation was run, again, with both our spectral quasi-cylindrical algorithm and the finite-difference quasi-cylindrical algorithm of Warp. In order to study numerical convergence, the simulations were run with various longitudinal resolution Δz, but with a fixed cell aspect ratio $\Delta r = 10\Delta z$. The timestep was again $c\Delta t = 1/\sqrt{1/\Delta z^2 + 2/\Delta r^2}$ for the finite-difference simulation and $c\Delta t = \Delta z$ for the spectral simulation. Finally, the simulations were run in a moving window with a longitudinal and transverse size of 40 μm and 60 μm respectively.

In order to assess the correctness of the force felt by the electron, we analyze the evolution of its transverse momentum in time. Analytically, it can be shown from the Lagrangian $\mathcal{L} = -\sqrt{1-\boldsymbol{\beta}^2}\, mc^2 - e\boldsymbol{v} \cdot \boldsymbol{A}$ that the equation of motion along the x axis (i.e. the axis of laser polarization) is

$$\frac{1}{c}\frac{d}{dt}\left(\frac{p_x}{m_e c} - a_x\right) = -\frac{p_x}{\gamma m_e c}\frac{\partial a_x}{\partial x} \qquad \sim \frac{a_0^2}{\gamma w_0} \tag{39}$$

where a_x is the normalized vector potential of the laser pulse. The right-hand side corresponds to the ponderomotive force, and, for the parameters and timescale considered here, it is in fact negligible. The

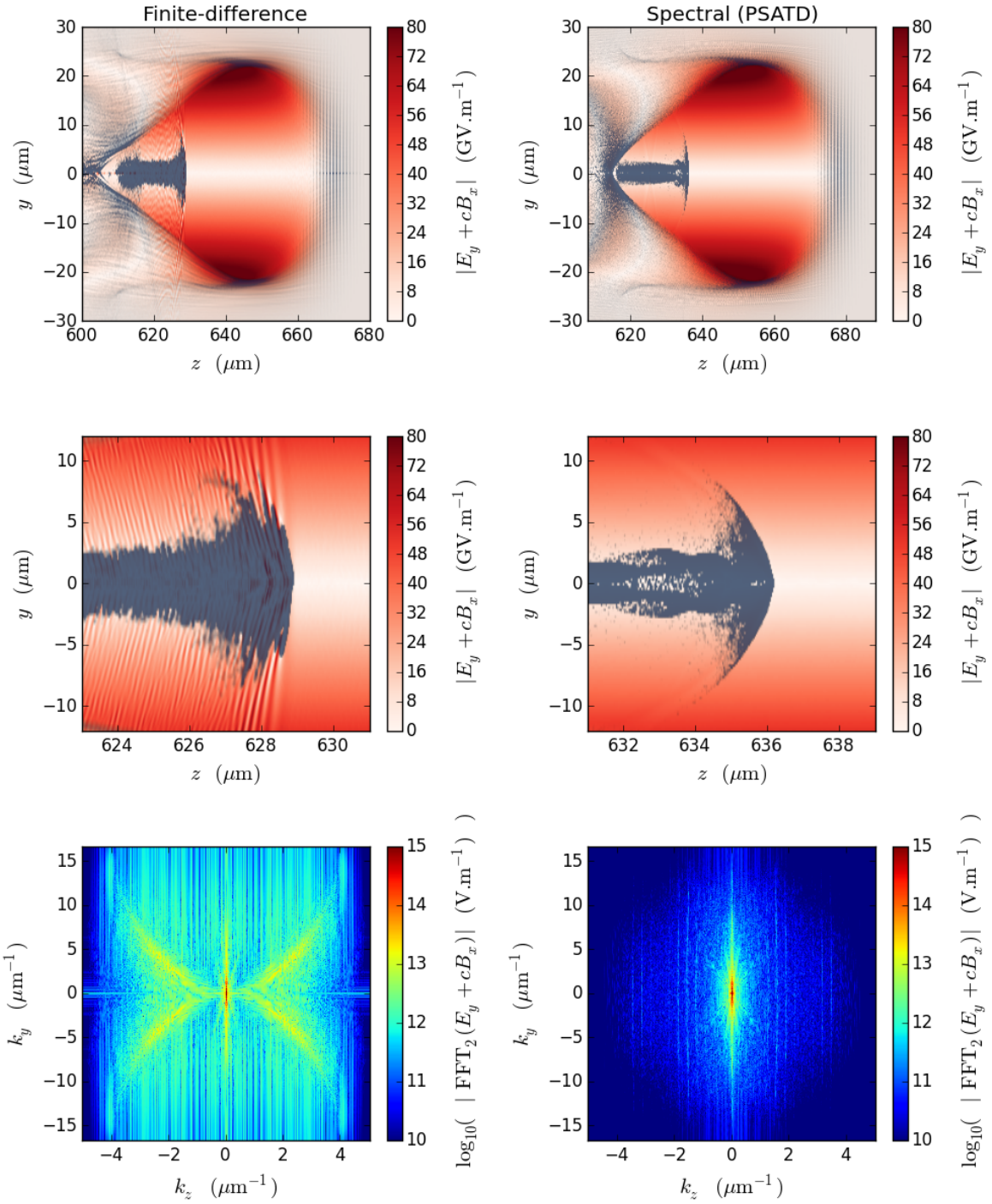

Figure 6: Snapshot of a finite-difference simulation (left) and of a spectral simulation (right), at a similar physical time. Upper panels: Representation of the bubble in real space; the superimposed blue shades represent the electron density. Middle panels: More detailed view of the region surrounding the bunch. Lower panels: two-dimensional Fourier transform of the quantity $E_y + cB_x$; the central spot around $k_z = 0$ corresponds to the physical fields, while the double-parabola in the left panel is a signature of the zero-order Cherenkov effect.

canonical momentum of the electron is thus approximately conserved.

$$\frac{p_x}{m_e c} - a_x = const. \tag{40}$$

Since, inside the laser pulse, a_x oscillates between the values $-a_0$ and a_0, the quantity $p_x/m_e c$ is predicted to also also oscillate between $-a_0$ and a_0, as the electron progressively dephases with the laser.

Keeping in mind that here initially $a_0 = 0.2$, one can observe on **??** that the oscillation amplitude of

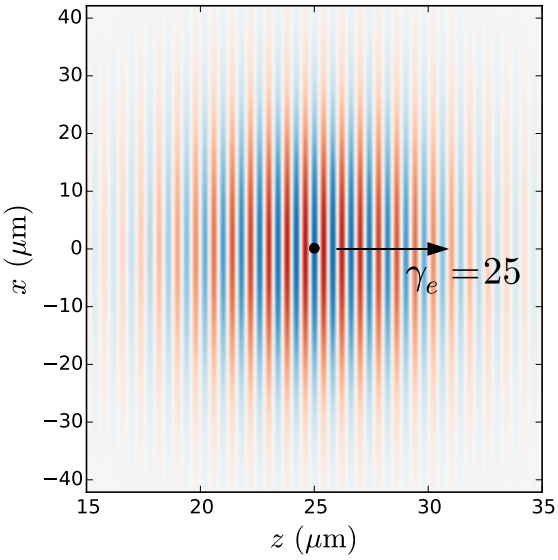

Figure 7: Representation of the situation which is simulated in **??**. The red and blue colormap corresponds to the electric field of the laser pulse, while the black dot corresponds to the electron considered. The laser propagates to the right, and is polarized along x.

$p_x/m_e c$ is much higher than analytically predicted in the finite-difference simulation. In addition, these oscillations strongly depend on the resolution of the simulation, which confirms their spurious nature. These observations are consistent with those of [**?**], and they are due to the above-mentioned erroneous calculation of the Lorentz force. On the other hand, in the spectral simulations, the oscillations of $p_x/m_e c$ exhibits virtually no dependence on the resolution. In addition, the oscillations of $p_x/m_e c$ have a realistic amplitude. (The fact that these oscillations do not reach exactly the value 0.2 may be due to the fact that a_0 decreases as the laser propagates, as a consequence of diffraction.) These observations confirm that the spectral quasi-cylindrical algorithm properly calculates the Lorentz force on the electron. This is also consistent with our expectations, which were based on the fact that $\boldsymbol{E}$ and $\boldsymbol{B}$ fields are not staggered in time, in this algorithm.

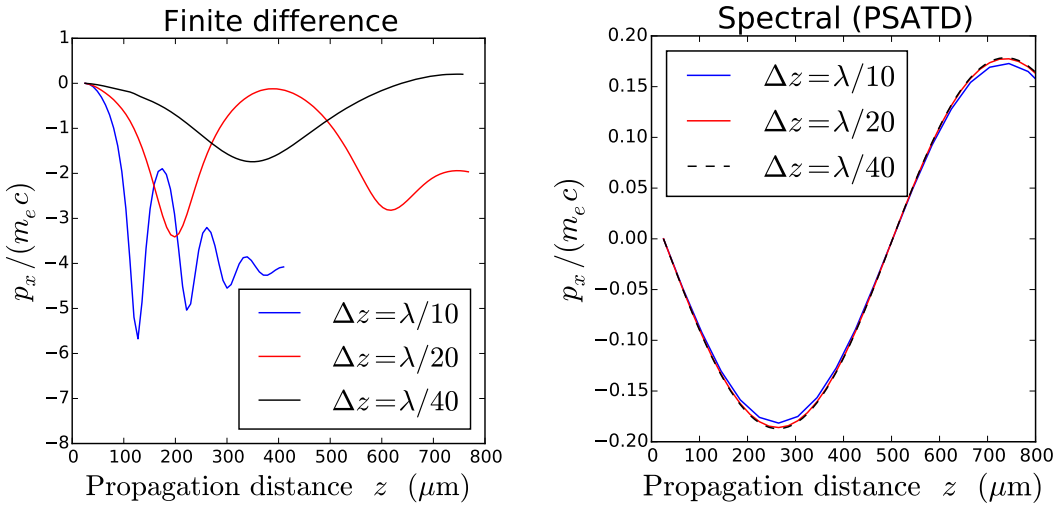

Figure 8: Evolution of the transverse normalized momentum of a macroparticle, as it copropagates with a laser pulse having $a_0 = 0.2$ (see **??** for a schematic representation of the situation simulated). The left and right plots correspond to the finite-difference and spectral algorithms respectively (note the different vertical scales on the two plots). The colored curves correspond to different resolutions for the simulations.

Conclusion

In this article, we derived the equations of a spectral quasi-cylindrical PIC algorithm, and discussed its numerical implementation. As explained in the text, because this algorithm is quasi-cylindrical, it requires much less computational time and memory than a full 3D algorithm. In addition, the fact that it is spectral (and uses the PSATD algorithm) prevents a certain number of artifacts, that are usually present in a finite-difference algorithm.

This second aspect was tested in detail in this article, through several benchmarks. By comparing simulations with analytical results, we showed that our spectral quasi-cylindrical code is free of spurious numerical dispersion, that it does not exhibit zero-order numerical Cherenkov effect in typical lab-frame simulations, and that it can accurately calculate the force of a laser on a copropagating electron. By running the same benchmarks with a finite-difference quasi-cylindrical algorithm, we showed that these numerical artifacts can, on the contrary, be present in finite-difference PIC algorithms.

However, it must not be forgotten that finite-difference algorithms have advantages of their own, including easier parallelization and application of boundary conditions. In the case of our spectral quasi-cylindrical algorithm, parallelization to several nodes could nonetheless be achieved by following for instance the method of [**?**]. This will be the subject of future work.

Acknowledgements. We thank Axel Huebl of the PIConGPU team [**?**] for interesting discussions on the GPU implementation, and Kevin Peters (U. Hamburg) for contributing to the validation of the code. We acknowledge computational resources from the Physnet cluster, University of Hamburg.

This work was supported by the Director, Office of Science, Office of High Energy Physics, U.S. Dept. of Energy under Contract No. DE-AC02-05CH11231, including from the Laboratory Directed Research and Development (LDRD) funding from Berkeley Lab.

Appendix A. Derivation of the spectral quasi-cylindrical representation

In order to derive the representation **??????**, we have to distinguish the Cartesian components (e.g. E_x, E_y, E_z, B_x, B_y, B_z), which are well-defined everywhere in space and thus have a regular Fourier representation, from the cylindrical components (e.g. E_r, E_θ, B_r, B_θ), which are ill-defined at $r = 0$. (For instance, for a field of the form $\boldsymbol{E} = E_0 \boldsymbol{e}_x$, the expression of E_r is $E_r = E_0 \cos(\theta)$, which ill-defined at $r = 0$, as θ itself is ill-defined at this position.)

Appendix A.1. Cartesian components

Let F_u be the Cartesian component of a field F (typically F is E, B or J and u is x, y or z). Its Fourier representation is thus given by **????**:

$$F_u(\boldsymbol{r}) = \frac{1}{(2\pi)^3} \int_{-\infty}^{\infty} \mathrm{d}k_x \int_{-\infty}^{\infty} \mathrm{d}k_y \int_{-\infty}^{\infty} \mathrm{d}k_z \; \mathcal{F}_u(\boldsymbol{k}) e^{i(k_x x + k_y y + k_z z)} \tag{A.1}$$

$$\mathcal{F}_u(\boldsymbol{k}) = \int_{-\infty}^{\infty} \mathrm{d}x \int_{-\infty}^{\infty} \mathrm{d}y \int_{-\infty}^{\infty} \mathrm{d}z \; F_u(\boldsymbol{r}) e^{-i(k_x x + k_y y + k_z z)} \tag{A.2}$$

Using the change of variable $k_x = k_\perp \cos(\phi)$, $k_y = k_\perp \sin(\phi)$, $x = r\cos(\theta)$, $y = r\sin(\theta)$, this becomes

$$F_u(\boldsymbol{r}) = \frac{1}{(2\pi)^3} \int_{-\infty}^{\infty} \mathrm{d}k_z \int_{0}^{\infty} k_\perp \mathrm{d}k_\perp \int_{0}^{2\pi} \mathrm{d}\phi \; \mathcal{F}_u(\boldsymbol{k}) e^{i(k_\perp r \cos(\theta-\phi) + k_z z)} \tag{A.3}$$

$$\mathcal{F}_u(\boldsymbol{k}) = \int_{-\infty}^{\infty} \mathrm{d}z \int_{0}^{\infty} r\mathrm{d}r \int_{-\infty}^{\infty} \mathrm{d}\theta \; F_u(\boldsymbol{r}) e^{-i(k_\perp r \cos(\theta-\phi) + k_z z)} \tag{A.4}$$

Let us now use the relation $e^{ik_\perp r\cos(\theta-\phi)} = \sum_{m=-\infty}^{\infty} i^m J_m(k_\perp r)e^{im(\phi-\theta)}$ (which is simply another way of writing the well-known relation $e^{i\alpha\sin\psi} = \sum_{m=-\infty}^{\infty} J_m(\alpha)e^{im\psi}$). The above equations become:

$$F_u(\boldsymbol{r}) = \sum_{m=-\infty}^{\infty} \frac{1}{(2\pi)^3}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp \mathrm{d}k_\perp \int_0^{2\pi}\mathrm{d}\phi\; i^m \mathcal{F}_u(\boldsymbol{k})\; J_m(k_\perp r)e^{-im(\theta-\phi)+ik_z z} \tag{A.5}$$

$$\mathcal{F}_u(\boldsymbol{k}) = \sum_{m=-\infty}^{\infty} \int_{-\infty}^{\infty}\mathrm{d}z \int_0^{\infty} r\mathrm{d}r \int_0^{2\pi}\mathrm{d}\theta\;\; (-i)^m F_u(\boldsymbol{r})\; J_m(k_\perp r)e^{-im(\phi-\theta)-ik_z z} \tag{A.6}$$

We now define $\hat{\mathcal{F}}_{u,m}(k_z,k_\perp) = \frac{1}{2\pi}\int_0^{2\pi}\mathrm{d}\phi\; i^m\mathcal{F}_u(\boldsymbol{k})e^{im\phi}$. This results in the following equations :

$$F_u(\boldsymbol{r}) = \frac{1}{(2\pi)^2}\sum_{m=-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp\mathrm{d}k_\perp\; \hat{\mathcal{F}}_{u,m}(k_z,k_\perp)\; J_m(k_\perp r)e^{-im\theta+ik_z z} \tag{A.7}$$

$$\hat{\mathcal{F}}_{u,m}(k_z,k_\perp) = \int_{-\infty}^{\infty}\mathrm{d}z \int_0^{\infty} r\mathrm{d}r \int_0^{2\pi}\mathrm{d}\theta\; F_u(\boldsymbol{r})\; J_m(k_\perp r)e^{-im\theta-ik_z z} \tag{A.8}$$

These equations correspond to **????**.

Appendix A.2. Cylindrical components

Let us now consider fields of the type E_r, B_r or J_r, which we denote generally by F_r. We have :

$$F_r = \cos(\theta)F_x + \sin(\theta)F_y = \frac{F_x - iF_y}{2}e^{i\theta} + \frac{F_x+iF_y}{2}e^{-i\theta} \tag{A.9}$$

Using **??** leads to

$$F_r = \frac{1}{(2\pi)^2}\sum_{m=-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp\mathrm{d}k_\perp\; \left(J_m(k_\perp r)\frac{\hat{\mathcal{F}}_{x,m} - i\hat{\mathcal{F}}_{y,m}}{2}e^{-i(m-1)\theta+ik_z z} + J_m(k_\perp r)\frac{\hat{\mathcal{F}}_{x,m}+i\hat{\mathcal{F}}_{y,m}}{2}e^{-i(m+1)\theta+ik_z z}\right) \tag{A.10}$$

$$F_r = \frac{1}{(2\pi)^2}\sum_{m=-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp\mathrm{d}k_\perp\; \left(J_{m+1}(k_\perp r)\frac{\hat{\mathcal{F}}_{x,m+1} - i\hat{\mathcal{F}}_{y,m+1}}{2}e^{-im\theta+ik_z z} + J_{m-1}(k_\perp r)\frac{\hat{\mathcal{F}}_{x,m-1}+i\hat{\mathcal{F}}_{y,m-1}}{2}e^{-im\theta+ik_z} \right. \tag{A.11}$$

where we relabeled the dummy variable m in the above sums. Let us thus define $\hat{\mathcal{F}}_{-,m} = (\hat{\mathcal{F}}_{x,m-1} + i\hat{\mathcal{F}}_{y,m-1})/2$ and $\hat{\mathcal{F}}_{+,m} = (\hat{\mathcal{F}}_{x,m+1} - i\hat{\mathcal{F}}_{y,m+1})/2$. This results in:

$$F_r(\boldsymbol{r}) = \frac{1}{(2\pi)^2}\sum_{m=-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp\mathrm{d}k_\perp\; \left(\hat{\mathcal{F}}_{+,m}\; J_{m+1}(k_\perp r) + \hat{\mathcal{F}}_{-,m}\; J_{m-1}(k_\perp r)\right)e^{-im\theta+ik_z z} \tag{A.12}$$

With the same definitions and the same method, it is also easy to show that:

$$F_\theta(\boldsymbol{r}) = \sum_{m=-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{d}k_z \int_0^{\infty} k_\perp\mathrm{d}k_\perp\; i\left(\hat{\mathcal{F}}_{+,m}\; J_{m+1}(k_\perp r) - \hat{\mathcal{F}}_{-,m}\; J_{m-1}(k_\perp r)\right)e^{-im\theta+ik_z z} \tag{A.13}$$

Appendix B. Maxwell equations for the spectral coefficients

In this section, let us derive the Maxwell equations for the spectral coefficients **??????** from the Maxwell equations written in cylindrical coordinates **??????**.

When replacing the Fourier-Hankel decomposition (**??????**) in the Maxwell equations **??????**, we first notice that the modes proportional to $e^{-im\theta+ik_z z}$ for different values of m and k_z are not coupled. These different modes can thus be treated separately. The same cannot be said of the modes corresponding to different values of $k_\perp$, since they may be coupled through the Bessel functions $J_m(k_\perp r)$ and their

derivatives. In the following, we write only the equations corresponding to $\partial_t \boldsymbol{B} = -\boldsymbol{\nabla} \times \boldsymbol{E}$, since the equation $c^{-2}\partial_t \boldsymbol{E} = \boldsymbol{\nabla} \times \boldsymbol{B} - \mu_0 \boldsymbol{j}$ can be treated very similarly. These equations become

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \left[\partial_t \hat{\mathcal{B}}_{+,m} J_{m+1}(k_\perp r) + \partial_t \hat{\mathcal{B}}_{-,m} J_{m-1}(k_\perp r) \right] = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[\hat{\mathcal{E}}_{z,m} \frac{im}{r} J_m(k_\perp r) - k_z \hat{\mathcal{E}}_{+,m} J_{m+1}(k_\perp r) + k_z \hat{\mathcal{E}}_{-,m} J_{m-1}(k_\perp r) \right] \tag{B.1a}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \left[\partial_t \hat{\mathcal{B}}_{+,m} J_{m+1}(k_\perp r) - \partial_t \hat{\mathcal{B}}_{-,m} J_{m-1}(k_\perp r) \right] = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[-k_z \hat{\mathcal{E}}_{+,m} J_{m+1}(k_\perp r) - k_z \hat{\mathcal{E}}_{-,m} J_{m-1}(k_\perp r) - ik_\perp \hat{\mathcal{E}}_{z,m} J'_m(k_\perp r) \right] \tag{B.1b}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, \partial_t \hat{\mathcal{B}}_{z,m} J_m(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[-ik_\perp \hat{\mathcal{E}}_{+,m} \left(\frac{J_{m+1}(k_\perp r)}{k_\perp r} + J'_{m+1}(k_\perp r) \right) + ik_\perp \hat{\mathcal{E}}_{-,m} \left(\frac{J_{m-1}(k_\perp r)}{k_\perp r} + J'_{m-1}(k_\perp r) \right) - \frac{im}{r} \left(E_{+,m} J_{m+1}(k_\perp r) + E_{-,m} J_{m-1}(k_\perp r) \right) \right] \tag{B.1c}$$

By taking the sum and difference of the first two equations, and by rearranging the third equation, we obtain:

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, 2\, \partial_t \hat{\mathcal{B}}_{+,m} J_{m+1}(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[ik_\perp \hat{\mathcal{E}}_{z,m} \left(\frac{m}{k_\perp r} J_m(k_\perp r) - J'_m(k_\perp r) \right) - 2k_z \hat{\mathcal{E}}_{+,m} J_{m+1}(k_\perp r) \right] \tag{B.2a}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, 2\, \partial_t \hat{\mathcal{B}}_{-,m} J_{m-1}(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[ik_\perp \hat{\mathcal{E}}_{z,m} \left(\frac{m}{k_\perp r} J_m(k_\perp r) + J'_m(k_\perp r) \right) + 2k_z \hat{\mathcal{E}}_{-,m} J_{m-1}(k_\perp r) \right] \tag{B.2b}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, \partial_t \hat{\mathcal{B}}_{z,m} J_m(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[-ik_\perp \hat{\mathcal{E}}_{+,m} \left(\frac{m+1}{k_\perp r} J_{m+1}(k_\perp r) + J'_{m+1}(k_\perp r) \right) -ik_\perp \hat{\mathcal{E}}_{-,m} \left(\frac{m-1}{k_\perp r} J_{m-1}(k_\perp r) - J'_{m-1}(k_\perp r) \right) \right] \tag{B.2c}$$

We can now use the relations $\frac{m}{k_\perp r} J_m(k_\perp r) + J'_m(k_\perp r) = J_{m-1}(k_\perp r)$ and $\frac{m}{k_\perp r} J_m(k_\perp r) - J'_m(k_\perp r) = J_{m+1}(k_\perp r)$ (see relation 9.1.27 in [**?**]), and obtain :

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, 2\, \partial_t \hat{\mathcal{B}}_{+,m} J_{m+1}(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[ik_\perp \hat{\mathcal{E}}_{z,m} \, J_{m+1}(k_\perp r) - 2k_z \hat{\mathcal{E}}_{+,m} J_{m+1}(k_\perp r) \right] \tag{B.3a}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, 2\, \partial_t \hat{\mathcal{B}}_{-,m} J_{m-1}(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[ik_\perp \hat{\mathcal{E}}_{z,m} \, J_{m-1}(k_\perp r) + 2k_z \hat{\mathcal{E}}_{-,m} J_{m-1}(k_\perp r) \right] \tag{B.3b}$$

$$\int_0^\infty k_\perp \mathrm{d}k_\perp \, \partial_t \hat{\mathcal{B}}_{z,m} J_m(k_\perp r) = \int_0^\infty k_\perp \mathrm{d}k_\perp \left[-ik_\perp \hat{\mathcal{E}}_{+,m} J_m(k_\perp r) - ik_\perp \hat{\mathcal{E}}_{-,m} \, J_m(k_\perp r) \right] \tag{B.3c}$$

Each equation of the above system contains Bessel functions of only one given order ($m+1$, $m-1$ or m). This allows the different $k_\perp$ components to be separated, since the functions $J_n(k_\perp r)$, for a fixed n and different values of $k_\perp$, form a basis of the set of real functions:

$$2\, \partial_t \hat{\mathcal{B}}_{+,m} = ik_\perp \hat{\mathcal{E}}_{z,m} - 2k_z \hat{\mathcal{E}}_{+,m} \tag{B.4a}$$

$$2\, \partial_t \hat{\mathcal{B}}_{-,m} = ik_\perp \hat{\mathcal{E}}_{z,m} + 2k_z \hat{\mathcal{E}}_{-,m} \tag{B.4b}$$

$$\partial_t \hat{\mathcal{B}}_{z,m} = -ik_\perp \hat{\mathcal{E}}_{+,m} - ik_\perp \hat{\mathcal{E}}_{-,m} \tag{B.4c}$$

Appendix C. PSATD scheme, in the Fourier-Hankel representation

We use a scheme very similar to that of [**?**]. In this scheme the currents are considered constant over one timestep, and the charge density is considered linear in time.

Appendix C.1. Expressions for $\hat{\mathcal{B}}_m$

By combining **??????** and **??**, one can find the propagation equations for B.

$$\partial_t^2 \hat{\mathcal{B}}_{+,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{B}}_{+,m} = \mu_0 c^2 \left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}_{z,m} + k_z \hat{\mathcal{J}}_{+,m} \right) \tag{C.1a}$$

$$\partial_t^2 \hat{\mathcal{B}}_{-,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{B}}_{-,m} = \mu_0 c^2 \left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}_{z,m} - k_z \hat{\mathcal{J}}_{-,m} \right) \tag{C.1b}$$

$$\partial_t^2 \hat{\mathcal{B}}_{z,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{B}}_{z,m} = \mu_0 c^2 (ik_\perp \hat{\mathcal{J}}_{+,m} + ik_\perp \hat{\mathcal{J}}_{-,m}) \tag{C.1c}$$

Let us integrate these equations for $t \in [n\Delta t, (n+1)\Delta t]$. In this interval, $\hat{\boldsymbol{\mathcal{J}}}_m(t)$ is constant and equal to $\hat{\boldsymbol{\mathcal{J}}}_m^{n+1/2}$, and thus the right-hand side of the above equations is constant. Using Green functions, the general solution of a differential equation of the form $\partial_t^2 f + \omega^2 f = g_0$, where g_0 is a constant, is

$$f(t) = f(t_0)\cos[\,\omega(t-t_0)\,] + \partial_t f(t_0)\frac{\sin[\,\omega(t-t_0)\,]}{\omega} + \frac{g_0}{\omega^2}(1 - \cos[\,\omega(t-t_0)\,]) \tag{C.2}$$

We thus use the above expression, with $\omega^2 = c^2(k_\perp^2 + k_z^2)$, to integrate the fields from $t_0 = n\Delta t$ to $t = (n+1)\Delta t$. In particular, we use again the Maxwell equations **??????** to obtain the expression of $\partial_t \hat{\mathcal{B}}_m(t_0)$. This yields:

$$\hat{\mathcal{B}}_{+,m}^{n+1} = C\hat{\mathcal{B}}_{+,m}^n - \frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}_{z,m}^n + k_z \hat{\mathcal{E}}_{+,m}^n \right) + \mu_0 c^2 \frac{1-C}{\omega^2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}_{z,m}^{n+1/2} + k_z \hat{\mathcal{J}}_{+,m}^{n+1/2} \right) \tag{C.3a}$$

$$\hat{\mathcal{B}}_{-,m}^{n+1} = C\hat{\mathcal{B}}_{-,m}^n - \frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{E}}_{z,m}^n - k_z \hat{\mathcal{E}}_{-,m}^n \right) + \mu_0 c^2 \frac{1-C}{\omega^2}\left(-\frac{ik_\perp}{2}\hat{\mathcal{J}}_{z,m}^{n+1/2} - k_z \hat{\mathcal{J}}_{-,m}^{n+1/2} \right) \tag{C.3b}$$

$$\hat{\mathcal{B}}_{z,m}^{n+1} = C\hat{\mathcal{B}}_{z,m}^n - \frac{S}{\omega}\left(ik_\perp \hat{\mathcal{E}}_{+,m}^n + ik_\perp \hat{\mathcal{E}}_{-,m}^n \right) + \mu_0 c^2 \frac{1-C}{\omega^2}\left(ik_\perp \hat{\mathcal{J}}_{+,m}^{n+1/2} + ik_\perp \hat{\mathcal{J}}_{-,m}^{n+1/2} \right) \tag{C.3c}$$

where $C = \cos(\omega\Delta t)$ and $S = \sin(\omega\Delta t)$.

Appendix C.2. Expressions for $\hat{\mathcal{E}}_m$

Similarly, when combining **??????** and **??**, the propagation equations for E are:

$$\partial_t^2 \hat{\mathcal{E}}_{+,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{E}}_{+,m} = \frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\hat{\rho}_m - \mu_0 c^2 \partial_t \hat{\mathcal{J}}_{+,m} \tag{C.4a}$$

$$\partial_t^2 \hat{\mathcal{E}}_{-,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{E}}_{-,m} = -\frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\hat{\rho}_m - \mu_0 c^2 \partial_t \hat{\mathcal{J}}_{-,m} \tag{C.4b}$$

$$\partial_t^2 \hat{\mathcal{E}}_{z,m} + c^2(k_\perp^2 + k_z^2)\hat{\mathcal{E}}_{z,m} = -\frac{c^2}{\epsilon_0} ik_z \hat{\rho}_m - \mu_0 c^2 \partial_t \hat{\mathcal{J}}_{z,m} \tag{C.4c}$$

Let us again integrate these equations for $t \in [n\Delta t, (n+1)\Delta t]$. In this interval, $\hat{\boldsymbol{\mathcal{J}}}_m(t)$ is constant (thus its time derivatives drop), and $\hat{\rho}_m$ is linear in time. As a consequence the right hand side is proportional to $\hat{\rho}_m^n + (\hat{\rho}_m^{n+1} - \hat{\rho}_m^n)(t-t_0)/\Delta t$. Using Green functions, the solution of $\partial_t^2 f + \omega^2 f = \hat{\rho}_m^n + (\hat{\rho}_m^{n+1} - \hat{\rho}_m^n)(t-t_0)/\Delta t$ is

$$f(t) = f(t_0)\cos[\,\omega(t-t_0)\,] + \partial_t f(t_0)\frac{\sin[\,\omega(t-t_0)\,]}{\omega} + \hat{\rho}_m^n \frac{1-\cos[\,\omega(t-t_0)\,]}{\omega^2} + \frac{\hat{\rho}_m^{n+1} - \hat{\rho}_m^n}{\omega^2}\left(\frac{t-t_0}{\Delta t} - \frac{\sin[\,\omega(t-t_0)\,]}{\omega\Delta t} \right) \tag{C.5}$$

which, for $t = t_0 + \Delta t$, reduces to

$$f(t_0 + \Delta t) = f(t_0)C + \partial_t f(t_0)\frac{S}{\omega} + \frac{\hat{\rho}_m^{n+1}}{\omega^2}\left(1 - \frac{S}{\omega\Delta t} \right) - \frac{\hat{\rho}_m^n}{\omega^2}\left(C - \frac{S}{\omega\Delta t} \right) \tag{C.6}$$

Using the above expression, we obtain

$$\hat{\mathcal{E}}_{+,m}^{n+1} = C\hat{\mathcal{E}}_{+,m}^n + c^2\frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}_{z,m}^n + k_z \hat{\mathcal{B}}_{+,m}^n - \mu_0 \hat{\mathcal{J}}_{+,m}^{n+1/2} \right) + \frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\left[\frac{\hat{\rho}_m^{n+1}}{\omega^2}\left(1 - \frac{S}{\omega\Delta t} \right) - \frac{\hat{\rho}_m^n}{\omega^2}\left(C - \frac{S}{\omega\Delta t} \right) \right] \tag{C.7a}$$

$$\hat{\mathcal{E}}_{-,m}^{n+1} = C\hat{\mathcal{E}}_{-,m}^n + c^2\frac{S}{\omega}\left(-\frac{ik_\perp}{2}\hat{\mathcal{B}}_{z,m}^n - k_z \hat{\mathcal{B}}_{-,m}^n - \mu_0 \hat{\mathcal{J}}_{-,m}^{n+1/2} \right) - \frac{c^2}{\epsilon_0}\frac{k_\perp}{2}\left[\frac{\hat{\rho}_m^{n+1}}{\omega^2}\left(1 - \frac{S}{\omega\Delta t} \right) - \frac{\hat{\rho}_m^n}{\omega^2}\left(C - \frac{S}{\omega\Delta t} \right) \right] \tag{C.7b}$$

$$\hat{\mathcal{E}}_{z,m}^{n+1} = C\hat{\mathcal{E}}_{z,m}^n + c^2\frac{S}{\omega}\left(ik_\perp \hat{\mathcal{B}}_{+,m}^n + ik_\perp \hat{\mathcal{B}}_{-,m}^n - \mu_0 \hat{\mathcal{J}}_{z,m}^{n+1/2} \right) - \frac{c^2}{\epsilon_0} ik_z \left[\frac{\hat{\rho}_m^{n+1}}{\omega^2}\left(1 - \frac{S}{\omega\Delta t} \right) - \frac{\hat{\rho}_m^n}{\omega^2}\left(C - \frac{S}{\omega\Delta t} \right) \right] \tag{C.7c}$$

Appendix D. Discrete Hankel Transform

Appendix D.1. Calculation of the transformation matrices $M_{n,m}$ and $M'_{n,m}$

Here, for the Discrete Hankel Transform, we use a transformation similar to that of [**?** **?** **?**], but we extend it to the case of an evenly-spaced grid in real space (as opposed to one that is distributed according to the zeros of the Bessel function, which would have been inconvenient for current deposition and field gathering). Moreover, we impose, as much as possible, that the succession of a Discrete Hankel Transform (DHT) and an Inverse Discrete Hankel Transform (IDHT) retrieves the initial function. As explained in the text of the article, we use a matrix formalism for the DHT:

$$\mathrm{DHT}^{\mathrm{m}}_{\mathrm{n}}[f]\,(k^m_{\perp,j}) = \sum_{p=0}^{N_r-1} (M_{n,m})_{j,p}\, f(r_p) \qquad \mathrm{IDHT}^{\mathrm{m}}_{\mathrm{n}}[g]\,(r_j) = \sum_{p=0}^{N_r-1} (M'_{n,m})_{j,p}\, g(k^m_{\perp,p}) \tag{D.1}$$

where n is the order of the Hankel transform, and where m is the index of the spectral grid $k^m_{\perp,j}$ on which the Hankel transform is evaluated. In practice, as mentioned in the text, these transforms are only used in the cases $n = m-1$, $n = m$ or $n = m+1$.

The $N_r \times N_r$ matrices $M_{n,m}$ and $M'_{n,m}$ can be entirely determined by a set of N_r^2 constraints. These constraints can be found, for instance, by imposing the value of the DHT for a set of N_r different functions, whose exact analytical Hankel transforms are known. In our case, we impose that the DHT be equal to the exact analytical Hankel transform for the eigenmodes of a cavity with perfectly conducting boundary at r_{max} ($\boldsymbol{E}(r_{max}, z) = 0$), since these physical eigenmodes should also be eigenmodes of our PIC cycle. These eigenmodes have the following form:

$$E_z \;\propto J_m(k^m_{\perp,\ell}\, r)\, e^{ik_z z - im\theta}\Theta(r_{max}-r) \tag{D.2a}$$

$$E_r - iE_\theta \;\propto J_{m+1}(k^m_{\perp,\ell}\, r)\, e^{ik_z z - im\theta}\Theta(r_{max}-r) \tag{D.2b}$$

$$E_r + iE_\theta \;\propto J_{m-1}(k^m_{\perp,\ell}\, r)\, e^{ik_z z - im\theta}\Theta(r_{max}-r) \tag{D.2c}$$

where Θ is the Heaviside function and where $k^m_{\perp,\ell} = \alpha^m_\ell / r_{max}$, with α^m_ℓ the ℓth positive zero of the Bessel function of order m. The exact Hankel transform of these modes, evaluated on the discrete set $\{k^m_{\perp,j}\}$ reads (see **??** for a derivation)

$$\mathrm{HT}_n[\, J_n(k^m_{\perp,\ell}\, r)\Theta(r_{max}-r)\,]\,(k^m_{\perp,j}) \quad \equiv 2\pi \int_0^{r_{max}} \mathrm{d}r\; J_n(k^m_{\perp,j} r) J_n(k^m_{\perp,\ell} r) \quad = \pi\, r^2_{max}\, [J_{n+\delta_{n,m}}(\alpha^m_\ell)]^2\, \delta_{j,\ell} \tag{D.3}$$

where n is either $m-1$, m or $m+1$. Note that the above relation is valid for any value of m, j, ℓ and n (provided that $n \in \{m-1, m, m+1\}$), except when $n \neq 0$, $m \neq 0$ and $\ell = 0$ simultaneously (again, see **??** for an explanation). Here we impose that the DHT is consistent with **??**, where applicable

$$\mathrm{DHT}^m_n[\, J_n(k^m_{\perp,\ell}\, r)\Theta(r_{max}-r)\,]\,(k^m_{\perp,j}) \quad \equiv \sum_{p=0}^{N_r-1} (M_{n,m})_{j,p} J_n(k^m_{\perp,\ell}\, r_p) \quad = \pi\, r^2_{max}\, [J_{n+\delta_{n,m}}(\alpha^m_\ell)]^2\, \delta_{j,\ell} \tag{D.4}$$

and we use the constraints given by **??** to obtain the matrix $M_{n,m}$.

Case where $n = 0$ or $m = 0$. In this case, **??** is applicable for any j and ℓ in $\{0, ..., N_r - 1\}$, and thus this provides N_r^2 constraints on the matrix $M_{n,m}$, which allow one to completely determine it. In fact, a closer look at **??** shows that the inverse matrix $M^{-1}_{n,m}$ can be directly extracted from the above relations, since this inverse matrices is defined by the relation $\sum_p (M_{n,m})_{j,p}(M^{-1}_{n,m})_{p,\ell} = \delta_{j,\ell}$. Thus, from the above relations, one can directly infer:

$$(M^{-1}_{n,m})_{p,\ell} = \frac{J_n(k^m_{\perp,\ell}\, r_p)}{\pi\, r^2_{max}\, [J_{n+\delta_{n,m}}(\alpha^m_\ell)]^2} \tag{D.5}$$

The matrix $M_{n,m}$ can then be extracted, by numerically inverting the matrices $M^{-1}_{n,m}$ given by the above expression. In addition, we impose that $M'_{n,m}$ be exactly equal to $M^{-1}_{n,m}$, so that the succession of a DHT and IDHT retrieves exactly the initial function.

Case where $n \neq 0$ and $m \neq 0$. In this case, the equation **??** is not valid for $\ell = 0$, and thus it provides only $N_r(N_r - 1)$ constraints on the matrix $M_{n,m}$, which is not enough to completely determine it. In this case, we use an empirical method in which we impose the additional constraints

$$(M_{n,m})_{0,p} = 0 \qquad \text{for} \quad p \in \{0, ..., N_r - 1\} \tag{D.6}$$

This constraint is imposed because we choose the amplitude of the Hankel mode proportional to $J_n(k_0^m r)$ to be 0 in the simulation. (For $n \neq 0$ and $m \neq 0$, $J_n(k_0^m r) = 0$ for any r, so that the amplitude of the corresponding mode has no physical meaning whatsoever.) This allows the matrix $M_{n,m}$ to be entirely determined. In addition, to obtain $M'_{n,m}$, we impose $(M'_{n,m})_{p,\ell} = \frac{J_n(k_{\perp,\ell}^m\, r_p)}{\pi\, r_{max}^2\, [J_{n+\delta_{n,m}}(\alpha_\ell^m)]^2}$ for any $\ell \in \{1, ..., N_r - 1\}$ (as in the case $n = 0$ or $m = 0$), but also $(M'_{n,m})_{p,\ell} = 0$ for $\ell = 0$. This is done again for consistency with the fact that $J_n(k_0^m r) = 0$. We note that the above method gives satisfying results for $m = 1$ but not for $m = 2$. In the future, further work will be done towards a better Hankel transform representation.

Appendix D.2. Derivation of **??**

Let us first remark that, through a change of variable where $r = r_{max} t$, **??** is equivalent to

$$\int_0^1 t dt\; J_n(\alpha_\ell^m t) J_n(\alpha_j^m t) \quad = \frac{1}{2}\, [J_{n+\delta_{n,m}}(\alpha_\ell^m)]^2\; \delta_{j,\ell} \tag{D.7}$$

and let us prove this equation for different cases, excluding the case where $n \neq 0$, $m \neq 0$ and $\ell = 0$ and in which it is not valid.

Case where $n = m$ and $\ell \neq 0$. In this case, we use the relation 11.4.5 of [**?**]. Since $n = m$, and since by definition α_j^m is the zero of the Bessel function of order m, we have $J_n(\alpha_j^m) = J_n(\alpha_\ell^m) = 0$, and the relation 11.4.5 is thus used in the case $b = 0$ and $a = 1$ (see relation 11.4.5 in [**?**] for the definition of the a and b coefficients). This yields

$$\int_0^1 t dt\; J_n(\alpha_\ell^m t) J_n(\alpha_j^m t) = \frac{1}{2}[J'_n(\alpha_\ell^m)]^2 \delta_{j,\ell} = \frac{1}{2}[J_{n+1}(\alpha_\ell^m)]^2 \delta_{j,\ell} \tag{D.8}$$

where we further used the relation 9.1.27 in [**?**], which reads $J'_n(\alpha_\ell^m) = -J_{n+1}(\alpha_\ell^m) + \frac{n}{\alpha_\ell^m} J_n(\alpha_\ell^m)$, and took into account the fact that $J_n(\alpha_\ell^m) = 0$ in our case.

Case where $n \neq m$ and $\ell \neq 0$. In this case, n is either $m + 1$ or $m - 1$, since we restricted ourselves to $n \in \{m - 1, m, m + 1\}$. Let us prove the equation **??** in the case $n = m + 1$ (the proof for $n = m - 1$ being very similar). By definition, we have $J_m(\alpha_\ell^m) = 0$, and thus from the relation 9.1.27 in [**?**] $J'_{m+1}(\alpha_\ell^m) + \frac{m+1}{\alpha_\ell^m} J_{m+1}(\alpha_\ell^m) = 0$. We can thus apply the relation 11.4.5 in [**?**] with $a = m + 1$ and $b = 1$, and find

$$\int_0^1 t dt\; J_n(\alpha_\ell^m t) J_n(\alpha_j^m t) = \frac{1}{2(\alpha_\ell^m)^2} \left((m+1)^2 + (\alpha_\ell^m)^2 - n^2\right) [J_n(\alpha_\ell^m)]^2 \delta_{j,\ell} = \frac{1}{2}[J_n(\alpha_\ell^m)]^2 \delta_{j,\ell} \tag{D.9}$$

Case where $m = 0$ and $\ell = 0$. The proof of the two above cases used the relation 11.4.5 from [**?**], which is applicable as long as $\alpha_\ell^m > 0$. This is the case for $\ell \neq 0$ (i.e. the two above cases) but also for $m = 0$ and $\ell = 0$ (i.e. $J_0(0) \neq 0$ and thus $\alpha_0^0 > 0$). As a consequence, the above proofs are also valid for $m = 0$ and $\ell = 0$.

Case where $n = 0$, $m \neq 0$ and $\ell = 0$. In this case, $\alpha_\ell^m = 0$ and thus the relation 11.4.5 from [**?**] does not apply. However, for $j \neq 0$, the left-hand side of equation **??** reduces to

$$\int_0^1 t dt\; J_n(\alpha_\ell^m t) J_n(\alpha_j^m t) = J_0(0) \int_0^1 t dt J_n(\alpha_j^m t) = \frac{J_0(0)}{(\alpha_j^m)^2} \int_0^{\alpha_\ell^m} t dt J_n(t) = 0 \tag{D.10}$$

where we used relation 11.1.1 in [**?**]. On the other hand, for $j = 0$, one has $\alpha_j^m = \alpha_\ell^m = 0$ and thus

$$\int_0^1 t dt\; J_n(\alpha_\ell^m t) J_n(\alpha_j^m t) = [J_0(0)]^2 \int_0^1 t dt = \frac{1}{2}[J_0(0)]^2 \tag{D.11}$$