The Discontinuous Galerkin Method
Applied to BOR Maxwell’s Equations

Elisabeth Blank, Kurt Busch, Fellow, OSA, Willy Dörfler, Michael König, Jens Niegemann

Abstract
The electromagnetic characterization of rotationally symmetric systems (bodies of revolution, BOR) plays an important role for a large number of technical applications. In numerical calculations, this symmetry can often be exploited to lower the effective dimensionality of the system which reduces the computational effort significantly. Here, we present an implementation of the discontinuous Galerkin time-domain (DGTD) method for the efficient solution of Maxwell’s equations in BOR structures. In particular, we derive an appropriate upwind flux and demonstrate how to calculate the local differentiation matrices efficiently from a set of pre-calculated global template matrices. In addition, we also discuss the implementation of suitable perfectly matched layers (PMLs) which allow to accurately simulate open systems. In a set of numerical experiments we find that our method is stable and shows optimal convergence properties.

1 Introduction
Structures which exhibit a cylindrical symmetry are omnipresent in the field of electromagnetics. Examples range from coaxial cables, antennas, and resonators in the microwave regime to lenses, tapered fibers, diffractive elements, and nano-plasmonic systems in the optical spectrum. Such structures are also called bodies of revolution (BOR) and their axial symmetry can be exploited to simplify the analytic or numerical treatment significantly. Indeed, over the past decades most of the commonly employed numerical techniques have been extended to also treat BOR systems efficiently. Besides the Method of Moments (MoM) [1] and the Finite Element Method (FEM) [2], this is also true for the Finite-Difference Time-Domain (FDTD) method [3]. In particular, the BOR–FDTD method has proven popular and been equipped with a number of important technical features such as efficient near-to-far-field transformation and plane wave excitation schemes [4,5]. BOR-FDTD has been applied to a large variety of systems including dielectric and metallic antennas [6,7] and resonators [8], optical lenses [9], diffractive elements [10], near-field probes [11], plasmonic nanostructures [12] and nano-laser [13].

While FDTD remains the most popular numerical method for time-domain simulation of Maxwell’s equations, in recent years a number of alternative methods were proposed. Possibly the most promising technique with respect to efficiency is the discontinuous Galerkin time-domain (DGTD) approach [14,15]. The DGTD approach has been successfully applied to such diverse tasks as ground-penetrating radar computations [16], specific absorption rate calculations in human tissues [17], and analyses of nano-photonic systems [18] (see also Refs. [19,20] for recent comparisons between DGTD and other time-domain approaches). In addition, DGTD has been combined with circuit simulation tools such as SPICE [21].

In this work, we demonstrate how to efficiently apply the nodal discontinuous Galerkin time-domain method to BOR structures, thus further increasing DGTD’s range of applicability. The paper is organized as follows: In Section 2 we will briefly recall the BOR Maxwell’s equations. We show that introducing cylindrical coordinates leads to equations with a singularity. This problem can be overcome by switching to the weak form. In Section 3 we then apply the DG method to discretize BOR Maxwell’s equations in space. An important ingredient of any DG scheme is the numerical flux, and we will derive an appropriate upwind flux for the BOR case. Spatial discretization leads to a semi-discrete scheme with local matrices varying from element to element. In Section 4 we will present an efficient way of computing some of these matrices from a set of global template matrices. Finally, in Section 5 we perform a number of numerical computations which demonstrate the stability, accuracy and versatility of our implementation.

2 BOR Maxwell’s Equations
Starting from the dimensionless Maxwell equations for isotropic materials with dielectric permittivity $\varepsilon$ and magnetic permeability $\mu$ in cylindrical coordinates $(r, \varphi, z)$,

\[
\begin{aligned}
\varepsilon \partial_t E_r - \frac{1}{r} \partial_r (r H_\varphi) + \partial_z H_r &= 0, \\
\varepsilon \partial_t E_\varphi + \partial_r (r H_r) - \partial_z H_\varphi &= 0, \\
\varepsilon \partial_t E_z - \frac{1}{r} (\partial_r (r H_\varphi) - \partial_\varphi H_r) &= 0, \\
\mu \partial_t H_r + \frac{1}{r} \partial_\varphi E_\varphi - \partial_z E_r &= 0, \\
\mu \partial_t H_\varphi - \partial_r E_r + \partial_z E_\varphi &= 0, \\
\mu \partial_t H_z + \frac{1}{r} (\partial_r (r E_\varphi) - \partial_\varphi E_r) &= 0,
\end{aligned}
\]

we initially encounter the problem of $1/r$-terms in the equations, which become singular at the symmetry axis $r = 0$. This is an artificial problem introduced by the chosen coordinate system. In the context of BOR-FDTD and BOR-FEM, several solutions have been suggested that overcome this problem, some of them are quite elaborate (see e.g. Ref. [22]).

2.1 Weak form
Here, we will give another possibility to solve the singularity problem, namely by switching to a weak formulation [23,24]. To do so, we start with Maxwell’s curl-equations in coordinate-independent form. To establish the weak form we then multiply by a smooth test function $\psi$ and integrate over the domain of interest $B \subset \mathbb{R}^3$. For the Maxwell–Faraday
Due to the periodicity of BOR systems in the Fourier Series Ansatz we can make a Fourier ansatz for the $\varphi$-variable of the fields as

\begin{align}
E(r, \varphi, z, t) &= \sum_{m=0}^{\infty} e^{im\varphi} E^{(m)}(r, z, t), \\
H(r, \varphi, z, t) &= \sum_{m=0}^{\infty} e^{im\varphi} H^{(m)}(r, z, t). 
\end{align}

Inserting this into (2), the $\varphi$-derivative drops out, and we obtain an infinite set of equations in the weak sense as

\begin{align}
\epsilon \partial_t E^{(m)}_x + \partial_z H^{(m)}_x &= \frac{im}{r} H^{(m)}_z, \\
\epsilon \partial_t E^{(m)}_y + \partial_z H^{(m)}_y &= \frac{im}{r} H^{(m)}_z, \\
\epsilon \partial_t E^{(m)}_z - \partial_x H^{(m)}_x &= \frac{1}{r} H^{(m)}_y - \frac{im}{r} H^{(m)}_z, \\
\mu \partial_t H^{(m)}_x - \partial_z E^{(m)}_z &= -\frac{im}{r} E^{(m)}_z, \\
\mu \partial_t H^{(m)}_y - \partial_z E^{(m)}_x &= -\frac{im}{r} E^{(m)}_z, \\
\mu \partial_t H^{(m)}_z + \partial_x E^{(m)}_x &= -\frac{1}{r} E^{(m)}_y + \frac{im}{r} E^{(m)}_z.
\end{align}

Thus, the set of six-dimensional equations is reduced to an infinite set of decoupled two-dimensional equations. Fortunately, depending on the excitation of the system, it is often sufficient to solve only for a single or very few values of $m$. A thorough theory about justification of this ansatz, convergence and error of the approximation can be found in Ref. [25].

3 The Discontinuous Galerkin Method Applied to BOR Maxwell’s Equations

In order to apply a discontinuous Galerkin discretization, it is advantageous to rewrite equations (4) in conservation law form as

\begin{equation}
Q \partial_t u + \nabla \cdot F(u) = \frac{1}{r} Bu, 
\end{equation}

with the state vector $u = (E^{(m)}, H^{(m)})^T \in \mathbb{R}^6$ for some $m \in \mathbb{N}$ (note: $u$ depends on $r = (r, z)$ and $\nabla = (\partial_r, \partial_z)^T$). $Q$ denotes the material matrix

\begin{equation}
Q := \begin{pmatrix} 0 & i \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{6,6}
\end{equation}

and $B$ is a sparse $6 \times 6$-matrix with non-zero entries

\begin{align*}
B_{1,6} &= im, & B_{3,4} &= -im, & B_{3,5} &= 1, \\
B_{4,3} &= -im, & B_{6,1} &= im, & B_{6,2} &= -1.
\end{align*}

Furthermore, the flux vector $F(u) \in \mathbb{R}^{6,2}$ is given by

\begin{equation}
F(u) := (A_r u, A_z u),
\end{equation}

where the $6 \times 6$-matrices $A_r$ and $A_z$ only have the following non-zero entries

\begin{align*}
(A_r)_{2,6} &= 1, & (A_r)_{3,5} &= -1, & (A_r)_{5,3} &= -1, & (A_r)_{6,2} &= 1; \\
(A_z)_{1,5} &= 1, & (A_z)_{2,4} &= -1, & (A_z)_{4,2} &= -1, & (A_z)_{5,1} &= 1.
\end{align*}

Note, that if the $(3 \times 3)$ material tensors $\xi$ and $\mu$ are symmetric, then so is $Q$. Therefore, since $Q$, $A_r$ and $A_z$ are symmetric, $F$ represents a hyperbolic system [20] Chap. 5. In our case of isotropic media, i.e., $\xi = \text{diag}(\epsilon, \epsilon, \epsilon)$ and $\mu = \text{diag}(\mu, \mu, \mu)$, we can calculate the eigenvalues explicitly and find them to be $\lambda_1 = 0$ and $\lambda_{2,3} = \pm 1/\sqrt{\mu}$. The material tensors are diagonal matrices $\Omega_k$, i.e., $\Omega = \bigcup_{k=1}^{N_k} \Omega_k$. The corresponding finite element space of discontinuous functions is then given by

\begin{equation}V_h := \{ u_h \in L^\infty(\Omega)^6 : u_h|_{\Omega_k} \in V(\Omega_k), \ k = 1, \ldots, K \}.
\end{equation}

$V(\Omega)$ is called the local approximation space, and we choose $V(\Omega_k) = P^\ell(\Omega_k)$ as the space of multivariate polynomials of total degree $p \in \mathbb{N}$. We approximate $u$ by $u_h \in V_h$ and use Lagrange interpolation to represent $u_h|_{\Omega_k}$. Thus, we obtain the nodal representation (cf. Ref. [23]) of the fields as

\begin{equation}u_h(r, t) = \sum_{j=1}^{N_p} u_h^k(r_j, t) \delta_h^k(r),
\end{equation}

where the $\delta_h^k$ are two-dimensional Lagrange polynomials on $\Omega_k$, $N_p = (p + 1)(p + 2)/2$ is the number of nodes, and $r_j^k = (r_j^k, z_j^k)$ are the chosen interpolation points. The Galerkin ansatz requires the residual $R_h := Q \partial_t u_h + \nabla \cdot F(u_h) - Bu_h$ to be orthogonal to all test functions $\psi_h \in V_h$ with respect to the measure $r dr = rd(r, z)$, i.e.,

\begin{equation}\int_{\Omega_h} R_h \cdot \psi_h r dr = 0.
\end{equation}

In the discontinous Galerkin method, we introduce a so-called numerical flux $F^*(u_h)$ which facilitates the coupling between neighboring elements. This leads to the expression [14] [23]

\begin{equation}\int_{\Omega_h} R_h \cdot \psi_h r dr = -\int_{\partial\Omega_h} (F(u_h^k) - F^*(u_h^k)) \cdot \psi_h r dr,
\end{equation}

where $n$ denotes the outer normal vector of the boundary $\partial\Omega_h$. Clearly, the numerical flux $F^*$ plays a crucial role for the
stability and accuracy of the scheme. While there are many choices of different numerical fluxes (see e.g. Refs [29,30]), the upwind flux is often considered most natural for hyperbolic problems. In the linear case, this choice produces convergent schemes, as was shown, e.g. in Ref. [29]. Moreover, for DG schemes applied to general hyperbolic equations an optimal convergence rate of $O(h^{p+1})$ for the $L^2$-error was shown in Ref. [31], and for Maxwell’s equations (in Cartesian coordinates) in Ref. [32]. As will be demonstrated in Section 5 we also find this convergence rate for our scheme.

Following the usual procedures of solving a linear Riemann problem by making use of the Rankine–Hugoniot jump conditions [29], we find the following explicit expression for the numerical flux for BOR Maxwell’s equations [4]

$$G_E := (F_E - F_E^c) n = -\frac{1}{Z} \hat{n} \times (Z^c \Delta H - \hat{n} \times E),$$

$$G_H := (F_H - F_H^c) n = \frac{1}{\nu} \hat{n} \times (Y^c \Delta E + \hat{n} \times \Delta H).$$

Here, $F_E \in \mathbb{R}^{3,2}$ denotes the projection of $\mathbb{P} \in \mathbb{R}^{6,2}$ onto the first 3 components, $F_H$ onto the last 3 components correspondingly, and $\hat{n} = (n_x, 0, n_z)^T$. “$+$” denotes the exterior of the local cell and “$-$” the interior. So $E^c$ is the value of the $E$-field in the interior of $\Omega_k$, $E^+$ the value in its exterior, and $\Delta E := E^+ - E^c$ are the jumps across the faces of the elements. $\Delta H$ is defined in an analogous manner. $Z^\pm = \sqrt{\mu/c^\pm}$ is the local impedance, and $Y^\pm = 1/Z^\pm$ is the local conductance. Finally, we also introduced $\bar{Z} := Z^+ + Z^-$ and $\bar{Y} := Y^+ + Y^-$. It should be noted that the expression (10) is again identical to the Cartesian case.

To obtain a semi-discrete system, we insert the expansion into [4] and choose test functions identical to the expansion basis $\{\psi_i\}$. We assume in the following that $\epsilon$ and $\mu$ are piecewise constant with respect to the chosen decomposition. After a few arithmetic transformations, the final semi-discrete expressions take the form

$$\epsilon \partial_t E_k^c = -D_k^c H_k^c + i\omega M_k^c H_k^c + \mathcal{L}^c G_k^c,$$

$$\partial_t H_k^c = -D_k^c \epsilon_{k} E_k^c + \partial_k H_k^c + \mathcal{L}^c G_k^c,$$

$$\partial_t \epsilon_{k} E_k^c = (\lambda^c + D_k^c \epsilon_{k}) H_k^c - i\omega M_k^c E_k^c + \mathcal{L}^c G_k^c,$$

$$\partial_t M_k^c E_k^c = -D_k^c \epsilon_{k} E_k^c - i\omega M_k^c E_k^c + \mathcal{L}^c G_k^c,$$

$$\partial_t H_k^c = (\lambda^c + D_k^c) E_k^c + i\omega M_k^c E_k^c - \mathcal{L}^c G_k^c,$$

which will be explained in more detail subsequently (here and below we will use the same symbol for the fields and the coefficient vectors). The local matrices $D_k^c$, $\epsilon_k$, $M_k^c$ and $\mathcal{L}^c$ can be expressed in terms of elementary matrices as

$$D_k^c = (M_k^c)^{-1} S_k^c, \quad D_k^c = (M_k^c)^{-1} S_k^c,$$

$$\lambda^c = (M_k^c)^{-1} M_k^c, \quad \mathcal{L}^c = (M_k^c)^{-1} \mathcal{L}^c.$$

Here, $M_k^c$ corresponds to the classical local mass matrix, while all other matrices now include an $r$-dependence. Explicitly, all matrices on the right-hand side of (12) can be expressed in terms of integrals of the basis functions as

$$\mathcal{M}_{ij} := \int_{\Omega_k} \psi_i^c (r) \psi_j^c (r) \, dr,$$

$$\mathcal{J}_{ij} := \int_{\Omega_k} \psi_i^c (r) \psi_j^c (r) \, dr,$$

$$\mathcal{S}^c_{ij} := \int_{\Omega_k} \psi_i^c (r) \psi_j^c (r) \, dr,$$

$$\mathcal{L}^c_{ij} := \int_{\Omega_k} \psi_i^c (r) \psi_j^c (r) \, dr.$$

Besides the local mass matrix $\mathcal{M}^c$, we call $\mathcal{M}^c$, the local BOR mass matrix, and $\mathcal{S}^c, \mathcal{L}^c$ the local BOR stiffness matrices.

In principle, one can evaluate the matrices (13) by means of a suitable quadrature rule for each element $k$. From those matrices, one can then easily pre-compute and store the matrices (12) which are required to evaluate the right-hand side of (11). However, such a quadrature-based approach requires the storage of four matrices (12) for each element. Therefore, such a procedure is not only demanding in terms of memory, it also negates much of the advantage that makes the discontinuous Galerkin time-domain approach so attractive for implementation on graphics processing units (GPUs) [32]. In the next section, we will present an alternative procedure to calculate the matrices (12), which at least partially mitigates the shortcomings of a naive quadrature-based procedure.

## 4 Efficient Implementation

A significant advantage of the DGTD approach in Cartesian coordinates is that all local matrices (12) can be expressed in terms of a few global template matrices. Here, we will demonstrate how to achieve something similar despite the explicit $r$-dependence of the overlap integrals in (13).

### 4.1 Transformation to a Reference Element

First, we introduce a reference triangle $I = \{a := (a,b) : a, b > -1, a + b \leq 0\}$. For straight-sided triangles, the transformation from $\Omega_k$ to $I$ is then given by the affine mapping

$$\psi^k : I \rightarrow \Omega_k,$$

$$a \rightarrow r = \psi^k(a) := v_1 + \frac{1}{2}(a + 1)(v_2 - v_1) + \frac{1}{2}(1 - 1)(v_3 - v_1)$$

with $J^k := \det(\nabla \psi^k) = \frac{1}{2} \text{vol}(\Omega_k)$. Here, $v_1 = (v_{11}, v_{12})$, $v_2 = (v_{21}, v_{22})$ and $v_3 = (v_{31}, v_{32})$ denote the vertices of the triangle $\Omega_k$ as depicted in Fig. [1]. In addition, we introduce the edge vectors $e_1 = v_2 - v_1$, $e_2 = v_3 - v_2$ and $e_3 = v_1 - v_3$. Furthermore, we obtain

$$\nabla \psi^k = \frac{1}{2} \begin{pmatrix} e_{11} & -e_{31} \\ e_{12} & -e_{32} \end{pmatrix},$$

and the inverse Jacobi matrix reads

$$(\nabla \psi^{-1})^k := \begin{pmatrix} \frac{1}{J^k} & 0 \\ 0 & \frac{1}{J^k} \end{pmatrix} \begin{pmatrix} e_{32} & e_{11} \\ -e_{31} & e_{12} \end{pmatrix}.$$
4.2 An Auxiliary Basis

As discussed in [28, Kap. 3.1], it is advantageous to employ an auxiliary orthogonal polynomial basis for the numerical construction of the local matrices.

The family of one-dimensional Jacobi polynomials \( \{ \phi_{n}^{(\alpha,\beta)} \}_{n \geq 0} \) on \([-1,1]\), for \( \alpha, \beta > -1 \), is constructed by the Gram-Schmidt orthonormalization procedure with respect to the weighted \( L^2 \)-scalar product on \((-1,1)\) using \( w(\tau) = (1-\tau)^{\alpha}(1+\tau)^{\beta} \) as a weight [28, App. A].

In Ref. [34] a family of two-dimensional Jacobi polynomials \( \{ \phi_{n,k}^{(\alpha,\beta,\gamma)} \}_{0 \leq k \leq n} \) on the simplex

\[ S := \{ (\xi, \eta) \in \mathbb{R}^2 : 0 < \eta < \xi < 1 \} \]

was derived, for \( \alpha, \beta, \gamma > -1 \), that is \( L^2(S) \)-orthogonal with respect to the weight function

\[ w_S(\xi, \eta) = (1-\xi)^{\alpha}(1-\eta)^{\beta}\eta^{\gamma}, \]

see Ref. [34] (3.10). The two-dimensional polynomials are obtained from the one-dimensional polynomials via

\[ \phi_{n,k}^{(\alpha,\beta,\gamma)}(\xi, \eta) = \phi_{n-k}^{(\alpha,\beta+2\gamma)}(2\xi-1)\phi_{k}^{(\beta,\gamma)}(2\eta - 1). \]

Since all our later operations are carried out on \( I \), we define a transformation from \( I \) to \( S \) by

\[ \Psi_I : I \rightarrow S, \quad \Psi_I(a, b) := \frac{1}{2}(1 - b, 1 + a). \]

Thus we obtain the two-variable analogues of the Jacobi polynomials on \( S \) by

\[ \tilde{\phi}_{n,k}^{(\alpha,\beta,\gamma)}(a, b) = \phi_{n,k}^{(\alpha,\beta,\gamma)} \circ \Psi_I^T(a, b) \]

\[ = \phi_{n-k}^{(\alpha,\beta+2\gamma)}(-b)\phi_{k}^{(\beta,\gamma)}(2(1 + a) - 2b - 1) \]

That are orthogonal with respect to the weight function

\[ w_I(a, b) = \frac{1}{2}\frac{1}{1+b+\gamma}(1+b)\alpha(1+b+\gamma)^{\alpha}, \]

We normalize the weighted \( L^2(I) \)-norm of \( \tilde{\phi}_{n,k}^{(\alpha,\beta,\gamma)} \) to unity to get the normalized set \( \{ \tilde{\phi}_{n,k}^{(\alpha,\beta,\gamma)} \}_{0 \leq k \leq n} \) of the two-variable polynomials [18]. For the choice \( \alpha = \beta = \gamma = 0 \) we recover the commonly used Dubiner polynomials [35], [34], [20], [28, Ch. A.1].

4.3 Local Mass Matrices

If the local polynomial degree is \( p \in \mathbb{N} \), the dimension of the local approximation space is \( N_p := (p + 1)(p + 2)/2 \). We take \( N_p \) distinct points \( \{ a_i \}_{1 \leq i \leq N_p} \subset I \) as in [28, Kap. 6.1] and let \( \{ l_i \}_{1 \leq i \leq N_p} \) be the corresponding nodal basis \( l_i(a) = \delta_{ij} \) of the polynomial space \( P_p \) on \( I \). A basis of \( P_p \) on \( \Omega_k \) is given by \( l_i^k := l_i \circ \Psi^{-1}_k \), \( \Psi_k \) from Sect. 4.1.

Then the local mass matrix \( M^k \) in [12] can be computed as

\[ M^k = \int_{\Omega_k} l_i^k(a)l_j^k(a) \det(\nabla \Psi_k) \, da \]

with \( \nabla \Psi_k \) constant for straight-sided simplices. In order to compute the mass matrix \( M \) on the reference element, we choose an (arbitrary but fixed) enumeration of the two-variable Jacobi polynomials [18] of degree \( p \), \( \tilde{\phi}_{j}^{(\alpha,\beta,\gamma)} \) for \( j = 1, \ldots, N_p \), and obtain as in [28, Kap. 3.2] (using the orthonormality) that

\[ M = (YV^T)^{-1}. \]

Here, \( Y \) is the generalized Vandermonde matrix \( \tilde{V}_{ij} = \tilde{\phi}_{j}^{(\alpha,\beta,\gamma)}(a_i) \). Thus, the computational effort for each \( M^k \) reduces to the scaling of a global template matrix \( M \).

For the local BOR mass matrix \( M^k_b \) in [12], the integral contains an explicit \( r \)-dependence. Inserting \( \Psi_k \) to map to the reference element \( I \), we find

\[ (M^k_b)_{ij} := \int_{I} r l_i^b(r) l_j^b(r) \, dr \]

\[ = J_k \int_I \left( v_{11} + \frac{1+a}{2} e_{11} - \frac{1+b}{2} e_{31} \right) l_i(a)l_j(a) \, da. \]

Therefore, we can express the local BOR mass matrix as

\[ M^k_b = J_k(v_{11}M + e_{11}(M^1 - e_{31}M^2)), \]

with template matrices that can be written in terms of Vandermonde matrices

\[ M^{(1)} = (V^{(1)}(V^{(1)}))^{-1} \quad \text{with} \quad (V^{(1)})_{ij} := \tilde{\phi}_{j}^{(0,0,1)}(a_i), \]

\[ M^{(2)} = (V^{(2)}(V^{(2)}))^{-1} \quad \text{with} \quad (V^{(2)})_{ij} := \tilde{\phi}_{j}^{(1,0,0)}(a_i), \]

using the respective orthonormality relations.

4.4 The Differentiation Matrices

In order to construct the differentiation matrices \( D^k_x \) and \( D^k_y \) in [11], we need first to consider the BOR stiffness matrices. Again, we apply the transformation \( \Psi^k \) and find, with notation from Sect. 4.1,

\[ (S^k_x)_{ij} := \int_{I} r l_i^b(r) \partial_x l_j^b(r) \, dr \]

\[ = J_k \left( v_{11}(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) + 2v_{12}b_z(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) - 2J_z a_z(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) \right) \]

\[ = J^k(M^k_x)_{ij}(a_z(D_{az})_i + b_z(D_{az})_j), \]

\[ (S^k_y)_{ij} := \int_{I} r l_i^b(r) \partial_y l_j^b(r) \, dr \]

\[ = J_k \left( v_{11}(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) + 2v_{12}b_z(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) - 2J_z a_z(a_iS_{ij}^{(3)} + b_iS_{ij}^{(4)}) \right) \]

\[ = J^k(M^k_y)_{ij}(a_z(D_{az})_i + b_z(D_{az})_j). \]
Interestingly, upon insertion of (21) and (22) into (25), where we have introduced the matrices
\[ S^{(1)}_{ij} := \int_a^b l_i(a) \partial_a l_j(a) \, da, \]
\[ S^{(2)}_{ij} := \int_a^b l_i(a) \partial_b l_j(a) \, da, \]
\[ S^{(3)}_{ij} := \int_a^b \frac{1 + a}{2} l_i(a) \partial_a l_j(a) \, da, \]
\[ S^{(4)}_{ij} := \int_a^b \frac{1 + a}{2} l_i(a) \partial_b l_j(a) \, da, \]
\[ S^{(5)}_{ij} := \int_a^b \frac{1 + b}{2} l_i(a) \partial_a l_j(a) \, da, \]
\[ S^{(6)}_{ij} := \int_a^b \frac{1 + b}{2} l_i(a) \partial_b l_j(a) \, da. \]

Analogously to Ref. [28, Kap. 3.2] one can show
\[ S^{(4)} = \mathcal{MD}_a, \quad S^{(2)} = \mathcal{MD}_b, \quad S^{(3)} = \mathcal{M}_1^{(1)} D_a, \]
\[ S^{(4)} = \mathcal{M}_1^{(2)} D_b, \quad S^{(6)} = \mathcal{M}_2^{(2)} D_b, \]
so that (21) (and 20) leads to
\[ S^k = \mathcal{M}_1^k \mathcal{D}_a + b_r D_b, \]
\[ S^k = \mathcal{M}_2^k \mathcal{D}_a + b_r D_b, \]
The local differentiation matrices \( D_a \) and \( D_b \) are given by
\[ (D_a)_{ij} := \partial_a l_j(a), \]
\[ (D_b)_{ij} := \partial_b l_j(a), \]
where \( a_i = (a_i, b_i) \), exactly as in [28, Kap. 3.2]. They can be computed in the same way via \( \mathcal{D}_m = V_m \mathcal{V}^{-1}, \) \( m \in \{a, b\} \), with
\[ (V_m)_{ij} := \partial_a l_j(a), \]
\[ (V_m)_{ij} := \partial_b l_j(a), \]
for \( \tilde{P}^{(\alpha, \beta, \gamma)} \) as in the beginning of this section.

In the semi-discrete expressions (11), the stiffness matrices only appear indirectly in the differentiation matrices
\[ D^k = (\mathcal{M}_1^k)^{-1} S^k, \]
\[ D^k = (\mathcal{M}_2^k)^{-1} S^k. \]
Interestingly, upon insertion of (21) and (22) into (23), we find
\[ D^k = a_r D_a + b_r D_b, \]
\[ D^k = a_s D_a + b_r D_b. \]

Therefore, identically to the Cartesian case, we can calculate the differentiation matrices from the entries of the Jacobi matrix of the element and two global template matrices. This saves a considerable amount of computer memory in contrast to storing two individual matrices for each element. In addition, it increases cache locality, which is particularly relevant for implementation of the algorithm on GPUs.

4.5 Surface Integrals for the Numerical Flux

We now consider the computation of the flux expression on the right hand side of the semi-discrete scheme (11). There, we have to evaluate surface integrals of the form
\[ \int_{\partial T_k} r G^k l_i^k (r) \, dr = \sum_{m=1}^3 \int_{\tau_m} r G^k l_i^k (r) \, dr, \]

where \( e_1, e_2, \) and \( e_3 \) denote the three edges of the triangle and \( G^k \) is one of the flux terms on the left in (10). Inserting the expansion of the fields into basis functions \( l_i^k \), we find
\[ \int_{\tau_m} r G^k (r) l_i^k (r) \, dr = \sum_j \mathcal{G}^k_j \int_{\tau_m} r l_i^k (r) l_j^k (r) \, dr. \]

This expression can be written as a matrix-vector product of the discrete flux vector with
\[ \mathcal{M}^{(k, e_m)} := \int_{\tau_m} r l_i^k (r) l_j^k (r) \, dr, \]
which is very similar to the BOR mass matrix, with the exception that the integration only runs over a certain edge of the element instead over its volume. Using the transformation \( \mathcal{G}^k (r) \equiv v_m + \frac{i}{2} \mathcal{G}^k (r) \), we can map the edges onto the reference interval \([-1, 1]\) and find
\[ \mathcal{M}^{(k, e_m)} := \frac{\mathcal{G} \int_{\tau_m} l_i^k (r) l_j^k (r) \, dr. \]

We have chosen the interpolation points such that there are \( p + 1 \) points on each edge with equidistant spacing. Thus we can replace the element-specific two-dimensional Lagrange polynomials by corresponding one-dimensional Lagrange polynomials on the reference interval, i.e.
\[ \mathcal{M}^{(k, e_m)} := \frac{\mathcal{G} \int_{\tau_m} l_i^k (r) l_j^k (r) \, dr. \]

Here we have to consider a one-to-one correspondence between the indices \( i, j \) \( \in \{1, \ldots, N_e\} \) for the nodes \( a_i \) in \( e_m \) to indices \( i', j' \) \( \in \{1, \ldots, p + 1\} \) for the nodes \( \tau_{i'} \) \( \in \{-1, 1\} \) with \( a_i = \tau_{i'} \)

Similarly to the procedure of the BOR mass matrix, we can express the Lagrange polynomials in terms of Jacobi polynomials and exploit the orthogonality relations to find
\[ \mathcal{M}^{(k, e_m)} := \frac{\mathcal{G} \int_{\tau_m} l_i^k (r) l_j^k (r) \, dr. \]

In order to have a one-to-one correspondence between the indices \( i, j \) \( \in \{1, \ldots, N_e\} \) for the nodes \( a_i \) in \( e_m \) to indices \( i', j' \) \( \in \{1, \ldots, p + 1\} \) for the nodes \( \tau_{i'} \) \( \in \{-1, 1\} \) with \( a_i = \tau_{i'} \)

Similarly to the procedure of the BOR mass matrix, we can express the Lagrange polynomials in terms of Jacobi polynomials and exploit the orthogonality relations to find
\[ \mathcal{M}^{(k, e_m)} := \frac{\mathcal{G} \int_{\tau_m} l_i^k (r) l_j^k (r) \, dr. \]

With the matrices \( \mathcal{M}^{(k, e_m)} \) at hand, we can easily construct the face matrix \( F^k \) for each element and we therefore managed to express all local matrices from (13) in terms of global template matrices. Unfortunately, we cannot easily construct the actual lift matrix \( \mathcal{L}^k = (\mathcal{M}_2^k)^{-1} F^k \) without explicit inversion of the BOR mass matrix for every element. Thus, for a practical implementation, it makes sense to pre-compute and store both \( \mathcal{M}^k \) and \( \mathcal{L}^k \). In contrast to a quadrature-based implementation, it still gain a factor of two in terms of memory. If memory is crucial, it is also possible to only pre-compute and store the inverse BOR mass matrix \( (\mathcal{M}_2^k)^{-1} F^k \) and construct the face and mass matrices \( \mathcal{M}^k \) and \( \mathcal{M}^k \) on-the-fly. We thereby reduce the memory requirements roughly by yet another factor two, but at the cost of the construction and two additional matrix-vector products. The latter procedure is especially attractive for the implementation on GPUs, where memory often is a particularly scarce resource.
5 Numerical Tests

5.1 Testing Stability and Convergence

As a first test to assess the convergence and stability behaviour of the scheme, we consider a homogeneous cylindrical resonator of radius $R = 1$ and length $L = 1$ with perfect electric conductor boundary conditions. The domain $\Omega = [0, 1]^2$ is meshed by regular triangles. The eigenmodes of this system are known analytically \[37\]. For our initial tests, we pick the TM-mode with $m = 1$ as an initial condition and let the fields evolve for 10 periods of oscillation. The actual time-stepping is performed with an optimized low-storage Runge–Kutta scheme \[38\]. At each timestep, we record the maximal deviation of the $E_z$-component from the analytical solution as a measure of the error. In Fig. 2 we then plot the maximal deviation over the entire time, for increasing polynomial order $p$ and decreasing maximal edge length $h$. We observe the expected behaviour of $O(h^{p+1})$, as in the Cartesian case. In addition, we run simulations with several, randomly chosen values of $p$ and $h$ for 10000 periods to check stability with respect to the relative error in the $L^2$-norm,

$$
\epsilon_{rel}(E_z) := \frac{||E_z - E_{z,an}||_2}{||E_z||_2}.
$$

All simulations remain stable and the error grows only linearly with time (not shown here).

5.2 Simulation of an Open System

For many realistic cases, it is necessary to simulate open systems. Unfortunately, it is non-trivial to formulate and implement exact open boundary conditions. As a common alternative, one can add an absorbing layer around the computational domain. If this layer is designed in such a way as to absorb outgoing radiation without any reflections at its interface, it is called a perfectly matched layer (PML). An efficient and stable way of implementing PMLs in BOR–DGTD is provided in Appendix A.

As a more realistic test setup, we simulate a semi-infinite dielectric fiber of radius $r = 1\,\mu m$ and with permittivity $\varepsilon_1 = 1.527$ in $z = [0, \infty]$. Below the fiber, at the distance of $1\,\mu m$, we put a sphere with radius $r_s = 1\,\mu m$ and with $\varepsilon_s = 12$. The computational domain was chosen to be $4.5\,\mu m \times 10\,\mu m$ in size and is surrounded by a PML of width $0.5\,\mu m$. The system is excited by an injection of a pulse within the fiber, travelling from the top downwards. The pulse has a carrier frequency of $\nu_0 = \frac{2\pi}{\lambda}$ and a Gaussian envelope of width $\sigma = 8$. The corresponding mesh was generated with NETGEN 4.9.9 \[39\] and is shown in Fig. 3. Some snapshots of the pulse travelling downwards and scattering at the sphere are shown in Fig. 4. The simulation was performed with polynomials of order $p = 5$ and we find, that the PMLs absorb the outgoing radiation without any visible reflection. Furthermore, the system remains stable for very long simulation times.

6 Summary and Outlook

In this work, we presented an efficient implementation of the nodal discontinuous Galerkin time-domain (DGTD) method for the solution of Maxwell’s equations in axi-symmetric (body of revolution, BOR) systems. In contrast to a naive, quadrature-based approach, we showed how all elementary matrices can be constructed by exploiting a suitable set of orthogonal polynomials. Most importantly, we demonstrated how the differentiation matrices can be directly assembled from two global template matrices.

While our approach requires to pre-compute and invert the matrix $M^e_k$ for each element, it still reduces the required memory by at least a factor of two when compared to the quadrature-based approach. Since BOR systems are effectively two-dimensional, in most cases this memory requirement does not lead to significant limitations in terms of applicability. For cases where memory is very scarce (e.g. extremely large systems or for computations on graphics processing units), our method allows to reduce the memory by roughly a factor of four at the price of some additional matrix-vector products.
Finally, in a set of numerical experiments, we demonstrated that our implementation yields optimal $hp$-convergence and is a promising method for solving the time-dependent Maxwell equations in BOR systems.

Acknowledgments

E. Blank is financially supported by the DFG Research Training Group 1294 "Analysis, Simulation and Design of Nano-technological Processes" at the KIT.

A Perfectly Matched Layers

The basis for our implementation of PMLs in BOR-DGTD is the formulation of UPMLs in cylindrical coordinates [40]. In $r$-direction we have the PML region $[R, r_1]$ and in positive and negative $z$-direction the regions $[Z, z_1]$ and $[−Z, −z_1]$, respectively (see Fig. 3 for an illustration). We assume constant $\epsilon$ and $\mu$ in each element. After Fourier transform (with the convention $\partial_i \leftrightarrow −i\omega$), we can write Maxwell’s equations with UPMLs as

$$-i\omega \Delta \mathbf{E}(r, \omega) = \nabla \times \mathbf{H}(r, \omega),$$
$$i\omega \mu \Delta \mathbf{H}(r, \omega) = \nabla \times \mathbf{E}(r, \omega).$$

Here, $\mathbf{E}, \mathbf{H}$ denote the electromagnetic fields $E, H$ in frequency domain and the diagonal $3 \times 3$-matrix $\Delta$ is given by

$$\Delta = \text{diag}\left(\begin{matrix} s_\phi & 0 & 0 \\ 0 & s_r & 0 \\ 0 & 0 & s_\phi \end{matrix}\right).$$

The parameters $s_r(r), s_\phi(z)$ and $s_r(r)$ are given by

$$s_r(r) = \frac{\kappa_r(r)}{\omega}, \quad s_\phi(z) = \kappa_\phi(z) - \frac{\sigma_r(z)}{\omega},$$
$$s_{r}(r) = \frac{\tilde{r}(r)}{r}, \quad \text{where} \quad \tilde{r}(r) = r_1 + \int_{r_1}^{r} s_{r}(r')dr'. $$

Throughout we choose $\kappa_r(r) = \kappa_\phi(z) = 1$, so that the PML parameters are constant, $\sigma_r(r) \equiv \sigma_r, \sigma_\phi(z) \equiv \sigma_\phi$, and

$$\tilde{r}(r) = \frac{\sigma_r}{\omega} r_1 + \left(1 - \frac{\sigma_r}{\omega}\right) r.$$

Thus, in frequency domain, Maxwell’s equations [4] with UPMLs can be written component-wise as

$$-i\omega E_r = -r \partial_r H_\phi + \frac{im}{r} H_z + \frac{1}{r} J_z^{(E)},$$
$$-i\omega E_\phi = -r \partial_r H_z + \partial_r H_\phi + \frac{1}{r} J_r^{(E)},$$
$$-i\omega E_z = \partial_r H_\phi + \frac{1}{r} H_r + \frac{im}{r} H_\phi + \frac{1}{r} J_r^{(E)},$$
$$-\mu i\omega H_r = \partial_r E_\phi - \frac{im}{r} E_z + \frac{1}{r} J_z^{(H)},$$
$$-\mu i\omega H_\phi = \partial_r E_z - \partial_r E_\phi + \frac{1}{r} J_r^{(H)},$$
$$-\mu i\omega H_z = -\partial_r E_\phi - \frac{1}{r} E_\phi - \frac{im}{r} E_z + \frac{1}{r} J_r^{(H)},$$

with the polarization currents $J_z^{(E)}$ and $J_r^{(H)}$. As an example, we will state the explicit expressions for the $r$-component of

![Figure 4: Simulation plots of $E_r$ travelling in negative z-direction along a fiber, ending in $z = 0$. The pulse is then scattered by a sphere with $c = 12$ and the outgoing radiation is absorbed by surrounding PMLs. (a) At time unit 30.0341, (b) at time unit 38.8765, (c) at time unit 56.8291, (d) at 75.3795.](image)

![Figure 5: Sketch of a BOR–PML-region (gray) with PML-width $r_1 - R = z_1 - Z$. The physical region is $[0, R] \times [−Z, Z]$.](image)
\( \bar{j}^{(E)} \), while all other components can be obtained analogously. So, with
\[
\bar{j}_r^{(E)} = i\omega \epsilon_r \left( \frac{s_s s_z}{s_r} - 1 \right) \bar{E}_r
\]  
we make use of the identity
\[
\frac{s_s s_z}{s_r} - 1 = \frac{1}{r(i\omega - \sigma_r)} \left( \frac{\sigma_r s_z}{i\omega} (r - r_1) - \sigma_z + r_1 \sigma_r \right)
\]
to find
\[
\bar{j}_r^{(E)} = \frac{i\omega \epsilon_r}{\omega - \sigma_r} \left( \frac{\sigma_r s_z}{i\omega} (r - r_1) - \sigma_z + r_1 \sigma_r \right) \bar{E}_r.
\]
Introducing the new variable
\[
\bar{P}_r^{(E)} := \bar{j}_r^{(E)} - \epsilon(r_1 \sigma_r - \sigma_z) \bar{E}_r,
\]
we can write
\[
-\omega \bar{P}_r^{(E)} = -\bar{\sigma}_r \bar{P}_r^{(E)} - \epsilon \sigma_z r_1 (\sigma_r - \sigma_z) \bar{E}_r.
\]
Solving (30) for \( \bar{\sigma}_r^{(E)} \) and insertion into (28) then eliminates the dependence from \( \bar{j}_r^{(E)} \). Performing an inverse Fourier transform finally yields the time-dependent equations for all components as
\[
\begin{align*}
re \partial_t E_r &= -r \partial_r H_\varphi + im H_z + P_r^{(E)} + \epsilon(r_1 \sigma_r - \sigma_z) E_r, \\
\partial_t P_r^{(E)} &= -\bar{\sigma}_r P_r^{(E)} - \epsilon \sigma_z r_1 (\sigma_r - \sigma_z) E_r, \\
re \partial_r H_\varphi &= -r \partial_r H_\varphi + \partial_r H_\varphi + P_r^{(E)} - \epsilon(r_1 \sigma_r - \sigma_z) E_r, \\
\partial_t P_r^{(E)} &= -\bar{\sigma}_r P_r^{(E)} - \epsilon \sigma_z r_1 (r_1 - r) \sigma_r - \sigma_z) E_r, \\
re \partial_t H_z &= r \partial_r E_\varphi - im E_z + P_z^{(H)} + \mu(r_1 \sigma_r - \sigma_z) H_z, \\
\partial_t P_z^{(H)} &= -\bar{\sigma}_z P_z^{(H)} - \mu \sigma_z r_1 (\sigma_r - \sigma_z) H_z, \\
r \mu \partial_r H_\varphi &= r \partial_r E_r - r \partial_r E_\varphi + P_r^{(H)} - \mu(r_1 \sigma_r + \sigma_z) H_\varphi, \\
r \partial_t P_r^{(H)} &= -\bar{\sigma}_r P_r^{(H)} + r_1 \sigma_z + P_r^{(H)} - \mu \sigma_z r_1 (r_1 - r) \sigma_r + \sigma_z) H_\varphi, \\
r \mu \partial_t H_\varphi &= -\bar{\sigma}_z \partial_r (E_r) + im E_\varphi + P_z^{(H)} + \mu((r_1 - 2) r) \sigma_r + \sigma_z) H_\varphi, \\
\partial_t P_z^{(H)} &= -\bar{\sigma}_z P_z^{(H)} + \mu \sigma_z^2 (r_1 - r) H_z + \mu \sigma_z ((r_1 - 2) r) \sigma_r - \sigma_z) H_z, \\
\end{align*}
\]
where we have one auxiliary differential equation for each component within the PML. It should be noted that these auxiliary differential equations do not contain spatial derivatives, which means that no modification of the numerical flux is needed.

References


