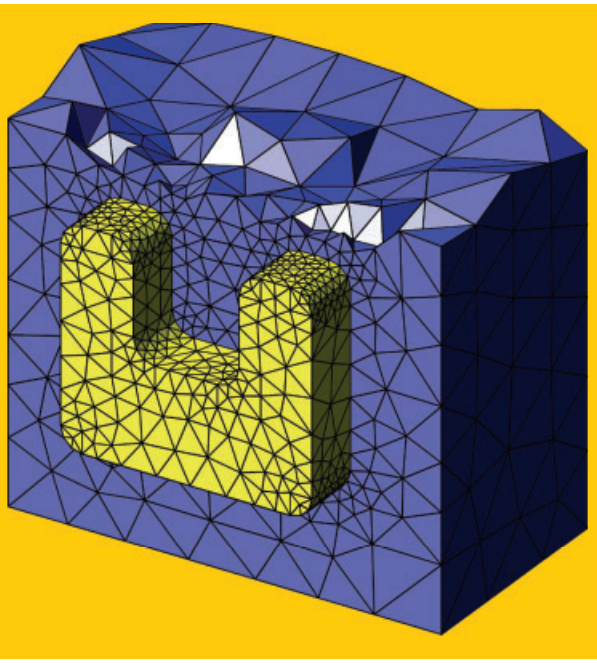
Early View publication on wileyonlinelibrary.com
(issue and page numbers not yet assigned;
citable using Digital Object Identifier – DOI)

LASER & PHOTONICS REVIEWS

Laser Photonics Rev., 1–37 (2011) / **DOI** 10.1002/lpor.201000045

Abstract Nanophotonic systems facilitate a far-reaching control over the propagation of light and its interaction with matter. In view of the increasing sophistication of fabrication methods and characterisation tools, quantitative computational approaches are thus faced with a number of challenges. This includes dealing with the strong optical response of individual nanostructures and the multi-scattering processes associated with arrays of such elements. Both of these aspects may lead to significant modifications of light-matter interactions.
This article reviews the state of the recently developed discontinuous Galerkin finite element method for the efficient numerical treatment of nanophotonic systems. This approach combines the accurate and flexible spatial discretisation of classical finite elements with efficient time stepping capabilities. We describe in detail the underlying principles of the discontinuous Galerkin technique and its application to the simulation of complex nanophotonic structures. In addition, formulations for both time- and frequency-domain solvers are provided and specific advantages and limitations of the technique are discussed. The potential of the discontinuous Galerkin approach is illustrated by modelling and simulating several experimentally relevant systems.



Discontinuous Galerkin methods in nanophotonics

Kurt Busch, Michael König*, and Jens Niegemann

1. Introduction

The last decades have seen amazing improvements in advanced nanofabrication techniques. Using tools such as electron beam lithography, it is possible to structure materials in the nanometer regime. Electronics has benefitted very much from these and other advances in technology and has brought them to everyday life. At the same time, nanostructuring has facilitated completely new ways of how to control light.

The most interesting effects occur when the feature sizes of the system are comparable to or much smaller than the wavelength of incident electromagnetic waves [1–4]. In particular, photonic crystals – which consist of periodically structured dielectric materials – prohibit the propagation of light along certain directions for certain frequencies via the formation of band gaps. Deliberately introduced deviations from the periodicity allow to design functional elements such as waveguides, splitters, interferometers, and more.

In a similar fashion, periodically arranged metallic building blocks such as nanorods, split-ring resonators, and fishnets can be combined to so-called metamaterials [1]: If these building blocks are much smaller than the wavelength of incident electromagnetic waves, then the composite material may be considered as an effective medium whose optical properties are largely determined by the building block. Intriguingly, this allows researchers to tailor material properties to their needs. For instance, metamaterials have been reported which show a distinct magnetic resonance at optical frequencies. For certain systems, even a negative effective refractive index has been found [5]. Possible applications and current research includes devices such as the perfect lens [6] and the optical cloak, which guides light around an obstacle in a way that it is invisible to an observer [7].

Nanostructures are also appealing for biological and chemical applications. For example, metallic nanostructures tend to enhance incident electric fields near tips and corners. The locally enhanced field drastically increases nonlinear effects such as the Raman effect. As a consequence, it is possible to measure and identify the Raman signal of single molecules [8]. Based on the same principle of local field enhancement, it is also possible to construct plasmonic tweezers [9] which trap and manipulate small particles. As the field enhancement goes along with a strong localisation, metallic nanostructures provide a way to overcome the diffraction limit at optical frequencies. Scanning near-field optical microscopes utilise metallic tips to regularly achieve resolutions in the sub-100 nm regime [10].

Furthermore, microscopic dielectric resonators show very pronounced resonances with enormous quality factors. At the same time, they are extremely sensitive to their

Institut für Theoretische Festkörperphysik and DFG-Center for Functional Nanostructures (CFN), Karlsruhe Institute of Technology (KIT), Wolfgang-Gaede-Straße 1, 76131 Karlsruhe, Germany
* Corresponding author: e-mail: mkoenig@tfp.uni-karlsruhe.de

environment – so sensitive, in fact, that even a single molecule can change their resonance frequency [11–13]. Consequently, such resonators are promising candidates for biological and chemical sensing applications.

The list of fascinating achievements and working devices is virtually endless. Despite the impressive bandwidth of these experiments, however, there is one commonality: Experimental research in the field of nanophotonics is challenging, expensive, and suffers from limited resources and infrastructure. Luckily, numerical simulation tools can support and ease experiments in a number of stages.

1.1. The need for numerical simulations

First of all, when planning a research project numerical experiments help to find promising systems and geometries. Proofs of concepts can be done without ever dealing with the fabrication process and the related difficulties. From such idealised experiments with optimal control, promising parameter ranges can be determined and the influence of different materials can be studied a-priori.

Once the systems of interest are fabricated, numerical simulations can be used to optimise the system parameters in order to tweak the performance, e. g., to increase the quality factor of a resonator structure. On the other hand, numerical simulations also allow to assess the quality of the fabrication process, as imperfections are usually not included in the numerical model.

Finally, simulations can simplify the interpretation. Using computer programs, it is possible to access quantities such as charge and electromagnetic field distributions or heat dissipation, which may not be accessible in the actual experiment. Physical effects can be switched on and off independently to investigate the dominant contribution to a phenomenon. Thus, we can learn for future designs. Simulation results can be visualised to obtain an intuitive understanding of the underlying physics. In some cases this might even help to understand unexpected effects which are not included in the physical model or have not been considered.

1.2. Common simulation methods

In nanophotonics it is usually sufficient to employ a classical description of the electromagnetic fields as given by Maxwell's equations. It is important to understand that the choice of the simulation tool is crucial for the quality of the results. Among the wide range of available methods we find a large number of techniques for specific applications. For example, the multiple multipole technique (MMP) is ideally suited to simulate spheroidal particles [14]. The Wannier function expansion can accurately describe large-scale photonic crystals with a comparatively small number of degrees of freedom [15, 16]. Layered periodic structures are easily investigated using the Fourier modal method [17–19].

Not surprisingly, such methods cannot fully display their strengths when being applied to systems other than the ones they were originally intended for. In such cases, converged numerical results require a disproportionately large amount of computational resources. In contrast, there exist a couple of general purpose solvers which can – some extensions provided – tackle an extremely diverse set of systems with only moderate requirements.

The Finite-Difference Time-Domain (FDTD) algorithm is the most popular simulation tool for nanophotonics [20]. Starting from an initial state, it propagates the electromagnetic fields in time. The field components are discretised at cleverly chosen positions within a rectangular lattice, the so-called Yee grid or staggered grid. Using a Taylor expansion for the derivatives and a leap-frog scheme in time, one ends up with a very simple procedure to evolve the fields step by step. Together with a multitude of extensions, FDTD has been successfully applied to countless systems, even nonlinear ones.

However, FDTD has a couple of weaknesses. It owes its simplicity to the comparatively inflexible Yee grid. Objects which are not parallel to the coordinate axes are subject to aliasing or the staircase effect, which is of special concern for metallic nanostructures. The accuracy of the spatial discretisation is further limited by the underlying Taylor expansion. As soon as material interfaces are present, the accuracy is reduced from second to first order because the electromagnetic fields are no longer smooth. Despite various efforts [21–23], it remains extremely challenging to improve the spatial accuracy beyond the limits of the basic algorithm.

As a second commonly used class of general purpose solvers we would like to mention the Finite Element Method (FEM) [24, 25]. FEM works on a non-uniform mesh which is adapted to the geometry of interest. At critical points, the mesh size can be reduced to improve the local resolution (h-refinement). Each element of the mesh holds an electromagnetic field representation of adjustable accuracy (p-refinement) and is coupled to its neighbours. Eventually, these building blocks lead to a sparse system of linear equations (SLE).

Solving such a system appears costly when compared to the computation of a single time step in FDTD. However, if one solves Maxwell's equations in the frequency-domain for just a small to medium (≈ 100) number of frequencies, this can be considerably faster than a corresponding time-domain simulation. Consequently, this is how FEM is most often applied. On the other hand, a finite element time stepping scheme is rather expensive, as one has to solve an SLE for each time step.

1.3. Overview of the discontinuous Galerkin method

It is desirable to find a way to combine the advantages of both FDTD and FEM. An adaptive mesh with adjustable order of accuracy should be woven into a scheme with reasonable efficiency concerning processor (CPU) time and RAM. Discontinuous Galerkin (DG) methods are one possibility to achieve this goal [26, 27].

The historical development of the DG method is rather intricate with important contributions by many researchers from different fields. Thus, we will restrict our historical account to a very few selected works relevant for Maxwell's equations. More detailled chronologies and information on other types of equations can be found in [26, 28, 29], for example.

The DG method has been first proposed and used in the context of steady-state neutron transport [30] in 1973. Later works have applied the method to other research areas like acoustics, plasma physics, and hydrodynamics. Combined with a Runge-Kutta scheme [31] the method showed very promising results for hyperbolic systems. After some previous efforts related to Maxwell's equations, Hesthaven and Warburton proposed a nodal DG time-domain scheme for the numerical solution of electrodynamic systems in 2002 [27]. Besides providing the algorithm itself, this paper also features rigorous mathematical proofs concerning numerical stability and convergence. Since then, a steadily increasing number of groups outside the mathematical community have adapted the DG method for Maxwell's equations, e. g., see [32–35].

In essence, the DG method is a variant of conventional FEM. Again, we divide the computational domain into a set of elements, the mesh. On each element, we expand the electromagnetic fields in terms of a set of basis functions. The main difference to FEM is that these basis functions are restricted to their respective elements; they are identically zero on all other – and especially the neighbouring – elements. Hence, at the interface between neighbours, we can have two *different* field values, one for each element. This is the origin of the word "discontinuous" in the method's name.

With the overlap of basis function being unavailable as a coupling mechanism, we can treat all elements individually. Any linear algebra we use acts only on the space of a single element and results in small, handable matrices. Once the expensive linear algebra has been performed, we reintroduce the coupling between neighbouring elements using the concept of the numerical flux, which is borrowed from finite volume methods. We obtain a semi-discrete form in which the position-dependence is already discretised while the time-dependence is not.

We can choose to discretise the time in terms of time steps similar to FDTD or we can solve Maxwell's equations in frequency-domain. The time-domain version indeed combines the advantages of both FDTD and FEM because it is as accurate as FEM and – for a comparable level of accuracy – faster than FDTD. The frequency-domain version is probably not as efficient as conventional FEM, but it can be derived from the time-domain algorithm with just a few modifications.

1.4. Outline of the review

The remainder of this article is structured as follows:

Section 2 describes the way from the physical system to the semi-discrete form for time-dependent expansion coefficients in some mathematical detail.

The semi-discrete form can either be used to evolve the fields in time or to obtain eigenmodes and stationary solutions in the frequency domain. Both trails are followed in Sect. 3.

Section 4 illustrates the capabilities of the DG method. After a general review of applications in the literature we focus on two practically relevant examples. The first one deals with the simulation of a two-dimensional ring resonator coupled to two waveguides. The second, three-dimensional example investigates the mutual interaction of two split-ring resonators. Both time- and frequency-domain techniques are employed. To illustrate the computational efficiency of the DG method, we provide details concerning computational time and required main memory.

Obviously, it is not possible to cover every aspect of the discontinuous Galerkin technique for Maxwell's equations within the scope of this review. Hence, we have compiled several topics, open questions and possible future developments in the outlook section to provide the reader with some suggestions for additional research.

The basic algorithm is capable of solving comparatively simple systems. More complicated setups involving open boundaries or dispersive materials require some extensions which are finally presented in the appendix. Readers familiar with FDTD will observe some similarities, but a few differences as well.

1.5. Target audience

This article is intended for readers interested in the numerical simulation of general nanophotonic systems. It does not assume specialised knowledge of numerical methods and, thus, should be accessible to physicists, electrical engineers, and applied mathematicians. Throughout the review we present the method and attached topics from an application-oriented perspective and focus on the numerical treatment of Maxwell's equations. In particular, we try to present the key ideas of the basic algorithm and its numerous extensions as intuitively as possible. At various points we give some hints on how to efficiently implement the DG method in a computer code. Where possible (and reasonable from our application point of view), we left mathematical details to the referenced literature to avoid unnecessary distraction.

More experienced readers might find some sections of particular interest. Readers familiar with FDTD will find an introduction into the finite element like discretisation in Sect. 2. Section 3 illustrates the freedom we have in discretising time, which is in stark contrast to FDTD, while many topics discussed in the appendix are mere variations of established FDTD techniques.

Experts on conventional finite element methods will find the key differences to the discontinuous Galerkin discretisation outlined in Sects. 2.2, 2.3, and 2.6. As FEM solvers usually work in the frequency-domain, Sect. 3.1 will be helpful to get a basic understanding of the time stepping process. Also, many of the extensions presented in the appendix deal with problems which only arise in the time-domain.

2. Spatial discretisation via the discontinuous Galerkin approach

In the remainder of this review we will concentrate on Maxwell's curl equations

$$\varepsilon(\vec{\mathrm{r}}) \cdot \partial_t \vec{\mathrm{E}}(\vec{\mathrm{r}}, t) = \vec{\nabla} \times \vec{\mathrm{H}}(\vec{\mathrm{r}}, t), \tag{1a}$$

$$\mu(\vec{\mathrm{r}}) \cdot \partial_t \vec{\mathrm{H}}(\vec{\mathrm{r}}, t) = -\vec{\nabla} \times \vec{\mathrm{E}}(\vec{\mathrm{r}}, t). \tag{1b}$$

Here, we have introduced the electric field $\vec{\mathrm{E}}$ and the magnetic field $\vec{\mathrm{H}}$. We will use the arrow-notation $\vec{\mathrm{a}}$ for all vectors which have an x-, y-, and z-component. The relative permittivity ε and the relative permeability μ are assumed to be linear, dispersionless and isotropic. It should be noted that we have absorbed the vacuum permittivity ε_0 and the vacuum permeability μ_0 into the electromagnetic fields to obtain dimensionless units. The speed of light is given by $c = 1/\sqrt{\varepsilon\mu}$. Defining the material matrix

$$\mathscr{Q}(\vec{\mathrm{r}}) = \begin{pmatrix} \varepsilon(\vec{\mathrm{r}}) & 0 \\ 0 & \mu(\vec{\mathrm{r}}) \end{pmatrix},$$

the state vector

$$\mathbf{q}(\vec{\mathrm{r}}, t) = \begin{pmatrix} \vec{\mathrm{E}}(\vec{\mathrm{r}}, t) \\ \vec{\mathrm{H}}(\vec{\mathrm{r}}, t) \end{pmatrix},$$

and the flux

$$\vec{\mathrm{F}} = \begin{pmatrix} \mathbf{F}_x \\ \mathbf{F}_y \\ \mathbf{F}_z \end{pmatrix}, \qquad \mathbf{F}_i(\mathbf{q}) = \begin{pmatrix} -\hat{\mathrm{e}}_i \times \vec{\mathrm{H}}(\vec{\mathrm{r}}, t) \\ \hat{\mathrm{e}}_i \times \vec{\mathrm{E}}(\vec{\mathrm{r}}, t) \end{pmatrix},$$

it is possible to reformulate Eq. (1) as the conservation law

$$\mathscr{Q}(\vec{\mathrm{r}}) \cdot \partial_t \mathbf{q}(\vec{\mathrm{r}}, t) + \vec{\nabla} \cdot \vec{\mathrm{F}}(\mathbf{q}) = 0. \tag{2}$$

In these definitions we have employed the Cartesian unit vectors $\hat{\mathrm{e}}_i$, $i = x, y, z$ (see Table 1).

Table 1 Notation for various types of vectors.

Notation	Meaning
$\hat{\mathrm{e}}_i$	Unit vector in i-direction ($i = x, y, z$)
$\hat{\mathrm{n}}$	Outwardly directed normal vector of unit length
$\vec{\mathrm{a}}$	Physical vector with x-, y-, and z-components
$\mathbf{a}$	Physical state vector with more than three components
$\tilde{\mathrm{a}}$	Vector of expansion coefficients for a single field component
$\vec{\tilde{\mathrm{a}}}$	A physical vector with x-, y-, and z-components, where each component is a vector of expansion coefficients $\tilde{\mathrm{a}}_x$, $\tilde{\mathrm{a}}_y$, and $\tilde{\mathrm{a}}_z$
$\tilde{\mathbf{a}}$	A physical state vector with more than three components, where each component is a vector of expansion coefficients

Please note that the state vector $\mathbf{q}$ represents the three components of both electromagnetic fields and, thus, has six components. The flux vector $\vec{\mathrm{F}}$ has three components (x, y, and z), where each component is a six-component vector itself. This notation seems confusing at first glance, but will turn out to be convenient in the following. Table 1 shows the most common vector notations in this paper.

We want to point out that we do not explicitly enforce the divergence conditions

$$\vec{\nabla} \cdot \vec{\mathrm{E}} = 0 \qquad \text{and} \qquad \vec{\nabla} \cdot \vec{\mathrm{H}} = 0.$$

As can be easily shown [27], the time-evolution given by Maxwell's curl equations conserves the divergence of the initial state. Hence, if the electromagnetic fields specified at $t = 0$ satisfy the divergence conditions, they will do so for all $t > 0$ as well. A similar statement holds for time-harmonic problems where the system is excited by external harmonic sources. Nonetheless, the negligence of the divergence conditions does lead to a large number of spurious modes in the spectrum of the DG operator. This becomes particularly problematic when trying to solve eigenvalue problems (see Sect. 3.2.2).

In the next few sections we will discuss how to deal with the spatial dependencies of Eq. (2). Section 2.1 introduces the computational domain and a corresponding decomposition into a set of elements. On each single element, Sect. 2.2 defines the properties of the numerical solution in a local sense, i. e., considering each element individually and not in the context of its neighbours. This context is subsequently reintroduced by the numerical flux in Sect. 2.3. In addition, the numerical flux is used to enforce boundary conditions as shown in Sect. 2.4. Finally, Sect. 2.5 introduces an expansion basis for the numerical solution. This yields the semi-discrete form, a spatially discretised version of Eq. (2) which includes time-dependent expansion coefficients. Some remarks regarding the efficiency and the error of the spatial discretisation conclude this section.

2.1. Tesselation of the computational domain

The DG method is a volume method, which means that we sample field values all over the volume as defined by the system and its surroundings. Because of limitations in available memory we have to restrict the simulation to a finite computational domain. On its borders, suitable boundary conditions need to be applied to mimic the physical behaviour of the part of the system which cannot be simulated. Boundary conditions within the DG method are discussed later in Sect. 2.4.

Depending on the dimensionality of the system, the computational domain is decomposed into mutually distinct elements which make up the mesh. For one-dimensional simulations these elements would be line segments. In two and three dimensions, the standard choices of triangles and tetrahedrons, respectively, lead to efficient methods. Other element types such as quadrilaterals, hexahedrons [36], prisms, pyramids, or meshes with multiple types are possible and

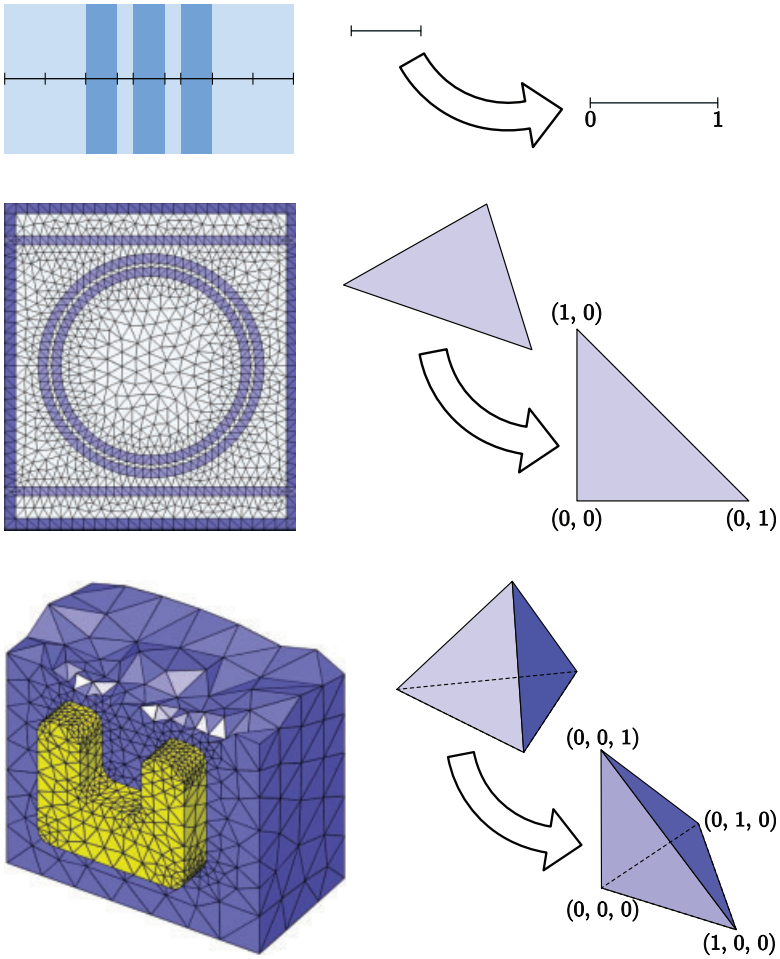


Figure 1 (online color at: www.lpr-journal.org) Tesselation of a few sample computational domains. Top panel: A series of Bragg layers (one-dimensional system). Middle panel: Two waveguides coupled to a slotted microresonator (two-dimensional system). Bottom panel: A split-ring resonator on a substrate (three-dimensional system). Each panel shows a mesh of the respective system which consists of several elements of different shapes and sizes. Furthermore, each panel features a zoomed view on an element and illustrates the transformation to a standardised reference element which significantly eases the computations (see Sect. 2.6).

sometimes advantageous. Even non-conforming discretisations can be used without too many difficulties [37, 38]. However, in the context of this article we will only consider conforming triangular and tetrahedral meshes. Either way, those elements need not be equally sized or shaped and, thus, create an unstructured mesh. This considerable freedom can be exploited to resolve spatial features of the computational domain with as many elements as needed to achieve a desired level of accuracy, while large homogeneous regions can be modelled with as few elements as possible. Figure 1 shows meshes for a couple of systems together with a sketch of typical elements.

The task of creating a mesh for a given geometric setup is a vast research field of its own. Luckily, there are quite a few meshing tools available. Among the more popular open source tools we find `NETGEN` [39], `TetGen` [40], and `Gmsh` [41].

2.2. Working on single elements

Let us now consider a single element of the computational domain. Our goal is to find a numerical approximation $\mathbf{q}_N$ to the correct solution $\mathbf{q}$ of Maxwell's equations in conservation form. In general, Eq. (2) will not hold exactly for $\mathbf{q}_N$, but will have to be modified to

$$\mathcal{Q}(\vec{\mathrm{r}}) \cdot \partial_t \mathbf{q}_N(\vec{\mathrm{r}}, t) + \vec{\nabla} \cdot \vec{\mathrm{F}}(\mathbf{q}_N) = \mathrm{res}\,, \tag{3}$$

where res is the residuum of the equation. Ideally, the residuum would be zero. As this is usually not the case, we have to find – and define – the best approximation. Within a given, finite, linear function space, there exists an optimal function which minimises the residuum. This residuum will then be orthogonal to the function space. Let $L_i(\vec{\mathrm{r}})$ represent a scalar basis function of the function space. The scalar product of Eq. (3) and L_i on the element $\triangle$ with volume $V_\triangle$ is given by

$$\int_{V_\triangle} \left(\mathcal{Q}(\vec{\mathrm{r}}) \cdot \partial_t \mathbf{q}_N(\vec{\mathrm{r}}, t) + \vec{\nabla} \cdot \vec{\mathrm{F}}(\mathbf{q}_N) \right) \cdot L_i(\vec{\mathrm{r}})\, \mathrm{d}^3 r$$
$$= \int_{V_\triangle} \mathrm{res} \cdot L_i(\vec{\mathrm{r}})\, \mathrm{d}^3 r \equiv 0\,. \tag{4}$$

The numerical solution must satisfy Eq. (4) for all test functions L_i.

At this point it should be noted that the last equation is completely local. It only involves field values and derivatives on the element $\triangle$. From physical reasoning it is clear that we cannot assume to get a correct solution of our problem from Eq. (4) alone, because Maxwell's equations are

formulated in a continuous space, not in a segmented one. Light waves propagate and, thus, must leave an element and enter the neighbouring one. We will account for this in the next section.

2.3. Connecting elements via the numerical flux

So far we have completely ignored the presence of neighbouring elements. However, solutions to Maxwell's equations must obey the continuity conditions. This leaves us with two options.

- We define boundary conditions on each element. The boundary conditions depend on the field values in neighbouring elements.
- We absorb the boundary conditions into modifications of the physical equations. The modifications will depend on fields in neighbouring elements as well.

For our purposes we choose the second option. Acknowledging that the coupling to neighbouring elements must be achieved across the element boundary, we integrate (4) by parts and obtain

$$\int_{V_\triangle} \left(\mathcal{Q}(\vec{\mathrm{r}})\, \partial_t \mathbf{q}_N(\vec{\mathrm{r}}, t) \cdot L_i(\vec{\mathrm{r}}) - \vec{\mathrm{F}}(\mathbf{q}_N) \cdot \vec{\nabla} L_i(\vec{\mathrm{r}}) \right) \mathrm{d}^3 r = -\int_{\partial V_\triangle} \left(\hat{\mathrm{n}} \cdot \vec{\mathrm{F}}(\mathbf{q}_N) \right) \cdot L_i(\vec{\mathrm{r}})\, \mathrm{d}^2 r.$$

In the right-hand side's integral over the element's surface $\partial V_\triangle$, $\hat{\mathrm{n}}$ represents the outwardly directed normal vector of unit length. We will now replace the flux $\vec{\mathrm{F}}$ on the right-hand side of this equation by the numerical flux, $\vec{\mathrm{F}}^*$, which we will have to define later. The seemingly random introduction of the numerical flux allows us to reintroduce the coupling to neighbouring elements which we have previously removed by defining the orthogonality in an element-wise fashion. Undoing the integration by parts then yields

$$\int_{V_\triangle} \left(\mathcal{Q}(\vec{\mathrm{r}}) \cdot \partial_t \mathbf{q}_N(\vec{\mathrm{r}}, t) + \vec{\nabla} \cdot \vec{\mathrm{F}}(\mathbf{q}_N) \right) \cdot L_i(\vec{\mathrm{r}})\, \mathrm{d}^3 r = \int_{\partial V_\triangle} \hat{\mathrm{n}} \cdot \left(\vec{\mathrm{F}}(\mathbf{q}_N) - \vec{\mathrm{F}}^*(\mathbf{q}_N) \right) \cdot L_i(\vec{\mathrm{r}})\, \mathrm{d}^2 r. \tag{5}$$

This is the strong variational formulation of Maxwell's curl equations. At this point, it is worth to mention that the left-hand side of Eq. (5) is still a local statement entirely confined to the element $\triangle$. The evaluation of the right-hand side requires both the flux and the numerical flux on the element's boundary. $\vec{\mathrm{F}}^*$ must introduce the coupling to neighbouring elements, and hence must incorporate field values from neighbouring elements. As a consequence, the right-hand side is a non element-local statement.

The proper choice of the numerical flux $\vec{\mathrm{F}}^*$ is essential for the correctness and the convergence of the scheme. Most interestingly, however, its choice is not unique. Hesthaven and Warburton have shown [27] that – for nodal schemes as presented later in Sect. 2.5 – a so-called upwind flux leads to a numerically stable and convergent scheme. It is given by the expression

$$\hat{\mathrm{n}} \cdot \left(\vec{\mathrm{F}}(\mathbf{q}_N) - \vec{\mathrm{F}}^*(\mathbf{q}_N) \right) = \begin{pmatrix} \frac{1}{\bar{Z}} \left(\alpha \left[\Delta\vec{\mathrm{E}} - \hat{\mathrm{n}}(\hat{\mathrm{n}} \cdot \Delta\vec{\mathrm{E}}) \right] + Z^+ \hat{\mathrm{n}} \times \Delta\vec{\mathrm{H}} \right) \\ \frac{1}{\bar{Y}} \left(\alpha \left[\Delta\vec{\mathrm{H}} - \hat{\mathrm{n}}(\hat{\mathrm{n}} \cdot \Delta\vec{\mathrm{H}}) \right] - Y^+ \hat{\mathrm{n}} \times \Delta\vec{\mathrm{E}} \right) \end{pmatrix}. \tag{6}$$

Employing definitions for the impedance

$$Z^\pm = \sqrt{\frac{\mu^\pm}{\varepsilon^\pm}},$$

the conductance

$$Y^\pm = \frac{1}{Z^\pm} = \sqrt{\frac{\varepsilon^\pm}{\mu^\pm}},$$

and their sums

$$\bar{Z} = Z^+ + Z^- \qquad \text{and} \qquad \bar{Y} = Y^+ + Y^-,$$

Eq. (6) includes material parameters from both the local element (index "−") and its neighbour (index "+") in the normal direction $\hat{\mathrm{n}}$. Furthermore, it includes the field differences

$$\Delta\vec{\mathrm{E}} = \vec{\mathrm{E}}^+ - \vec{\mathrm{E}}^- \qquad \text{and} \qquad \Delta\vec{\mathrm{H}} = \vec{\mathrm{H}}^+ - \vec{\mathrm{H}}^-$$

across the interface to each neighbour. Stated more clearly, $\vec{\mathrm{E}}^+$ is the limit of the electric field on the interface when approaching from the neighbouring element, while $\vec{\mathrm{E}}^-$ is the limit when coming from the interior of the local cell. As both limits are not necessarily identical, $\Delta\vec{\mathrm{E}}$ describes a jump discontinuity. Such discontinuities naturally occur at material interfaces, where the normal component of the electric field changes according to $\varepsilon^- \hat{\mathrm{n}} \cdot \vec{\mathrm{E}}^- = \varepsilon^+ \hat{\mathrm{n}} \cdot \vec{\mathrm{E}}^+$. As a consequence, expression (6) weakly enforces that

1. the fields are continuous where they ought to be and
2. satisfy the matching conditions at material interfaces.

This is illustrated in Fig. 2.

We have not yet discussed the upwind parameter α in Eq. (6). Any number in $[0, 1]$ yields a numerically stable and convergent scheme [26]. The value 1 represents the pure upwind flux, whereas we recover the central flux for $\alpha = 0$. The central flux is energy-conserving while the upwind flux is not. As it turns out, for a nodal expansion basis as introduced later in Sect. 2.5 the upwind flux is preferable because it strongly damps unphysical modes. Thus, the choice of α also influences the accuracy of the scheme. Optimal convergence rates are achieved for $\alpha = 1$, while the rates are less clear for other values of α [26, 27].

2.4. Boundary conditions

It has already been mentioned that volume methods cannot discretise an infinite space with a finite amount of memory.

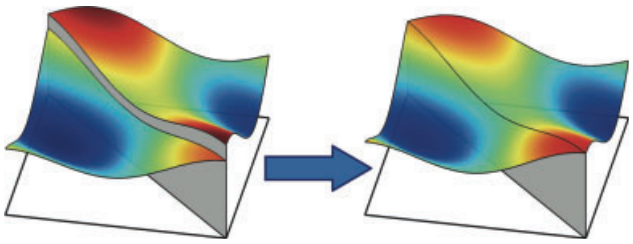


Figure 2 (online color at: www.lpr-journal.org) Illustration of the influence of the numerical flux. The panels show two neighbouring elements (triangles) and the component of the electric field tangential to the interface between both elements. The left panel depicts the situation where no numerical flux is present, i. e., where no coupling between the elements exists. As a result, the field distribution on the interface is not unique. The situation changes if one employs the upwind flux given by Eq. (6), which penalises jump discontinuities of the tangential fields. As a result, it weakly enforces the continuity of tangential fields across element boundaries, as shown in the right panel.

Hence, the computational domain must be truncated. We need to specify boundary conditions, i. e., how the electromagnetic fields should behave at the interface to the not simulated part of the universe. The numerical flux allows for an easy inclusion of the most important boundary conditions.

Periodic boundary conditions are a special case among the boundary conditions. In contrast to the other conditions presented below, each element on the boundary of the computational domain has a well-defined neighbour. Consequently, we can apply the very same numerical flux as we do for elements in the interior of the computational domain. Implementing this boundary condition is thus reduced to the search of the correct neighbouring element.

The situation is a bit more complicated for actual boundary elements, i. e., elements missing a neighbour. For such elements we have to provide information on the materials and fields in *virtual* neighbouring elements. To do so, we simply set

$$Z^+ \equiv Z^- \qquad \text{and} \qquad Y^+ \equiv Y^-$$

and modify the field differences as indicated in Table 2. Perfect electric (PEC) and perfect magnetic conductors (PMC) reflect all incident radiation and are used to create cavities or to introduce symmetry planes into the system. Silver-Müller boundary conditions [24, 25] mimic an infinite computational domain by partially absorbing outgoing radiation. However, only spherical waves which impinge normally to the boundary are (at least theoretically) perfectly absorbed. All other angles of incidence lead to spurious reflections. Thus, to minimise reflections one should place a spherical Silver-Müller boundary sufficiently far away from the radiation source. Still better results can be obtained by adding perfectly matched layers (see Sect. A.4).

Table 2 Modified field differences for commonly used boundary conditions.

Boundary condition	$\Delta\vec{\mathrm{E}}$	$\Delta\vec{\mathrm{H}}$
Perfect electric conductor (PEC)	$-2\vec{\mathrm{E}}^-$	0
Perfect magnetic conductor (PMC)	0	$-2\vec{\mathrm{H}}^-$
First order absorbing (Silver-Müller)	$-2\vec{\mathrm{E}}^-$	$-2\vec{\mathrm{H}}^-$

2.5. The semi-discrete problem

Equations (5) and (6) are the strong formulation of Maxwell's curl equations for numerical approximations of the electromagnetic fields. However, the numerical nature of the approximation has yet to be specified, which will be the task of this section.

To this end, we represent the electromagnetic fields in terms of the previously defined test functions L_i. Using the same function space for both the test functions and the field expansion is called the Galerkin choice. For each field component in each element $\triangle$, we obtain expressions analogous to

$$\mathrm{E}_x^{\triangle}(\vec{\mathrm{r}},t) = \sum_{j=1}^{n} \tilde{\mathrm{E}}_{x,j}^{\triangle}(t)\cdot L_j(\vec{\mathrm{r}}) \equiv \tilde{\mathrm{E}}_{x,j}^{\triangle}(t)\cdot L_j(\vec{\mathrm{r}})\,. \tag{7}$$

Here, $\tilde{\mathrm{E}}_x^{\triangle}$ was introduced as a vector of n expansion coefficients for the x-component of the electric field (see Table 1). Note the Einstein notation which implies summation for repeating indices. In general, one has to reconstruct the numerical solution using the expansion coefficients and the basis. However, there is a special basis which allows us to immediately connect the expansion coefficients with field values. The basis functions we use are called Lagrange polynomials or interpolating polynomials. Their defining property is given by

$$L_i(\vec{\mathrm{r}}_j) = \delta_{ij} \equiv \begin{cases} 0 & \text{for} \quad i \neq j \\ 1 & \text{for} \quad i = j \end{cases} \qquad i,\, j \in 1\ldots n\,, \tag{8}$$

where δ_{ij} is the Kronecker symbol. Given a number n of nodes $\vec{\mathrm{r}}_j$, each basis function will be zero for each but one of them. With this property it is easy to show that

$$\mathrm{E}_x^{\triangle}(\vec{\mathrm{r}}_k) = \tilde{\mathrm{E}}_{x,j}^{\triangle}(t)\cdot L_j(\vec{\mathrm{r}}_k) = \tilde{\mathrm{E}}_{x,j}^{\triangle}\cdot\delta_{jk} = \tilde{\mathrm{E}}_{x,k}^{\triangle}.$$

Thus, the expansion coefficients correspond to the field values at the nodes. Hence, using this special basis the scheme is called a nodal DG method. Usually, the more basis functions – and thus more coefficients – we use, the more accurate our expansion will be. With the general shape of Lagrange polynomial of order p being given by

$$L_i(\vec{\mathrm{r}}) = \sum_{k,l,m=0}^{k+l+m\leq p} a_{k,l,m}^{(i)}\cdot x^k y^l z^m\,,$$

we get a total of

$$n = p+1 \tag{1D}$$

$$n = \tfrac{1}{2}\cdot(p+1)\cdot(p+2) \tag{2D}$$

$$n = \tfrac{1}{6}\cdot(p+1)\cdot(p+2)\cdot(p+3) \tag{3D}$$

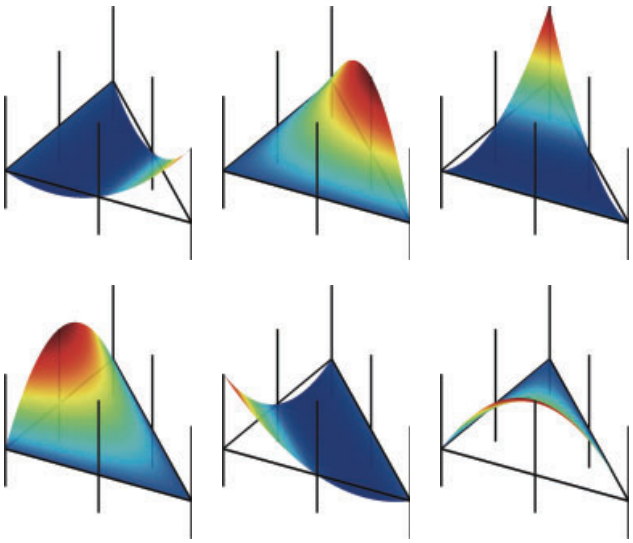


Figure 3 (online color at: www.lpr-journal.org) Two-dimensional Lagrange polynomials on a triangle. For $p = 2$ we obtain a total of six Lagrange polynomials which are depicted above. The triangle outline marks the x-y-plane, the values of the Lagrange polynomials are encoded in the z-direction. The node positions are given by the vertical lines which range from $z = -0.5$ to $z = 1$. Apparently, each Lagrange polynomial is assigned to a node where it has unit value; at all other nodes it is zero. In between the nodes Lagrange polynomials show a smooth interpolation behaviour.

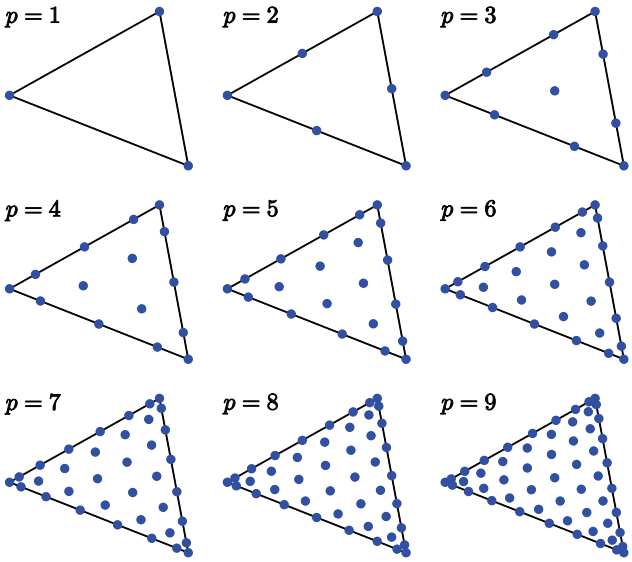


Figure 4 (online color at: www.lpr-journal.org) Node positions in two dimensions for various polynomial orders p. The optimal node distribution on the edges is given by an analytical formula. Using the Warp&Blend method nodes are smoothly arranged within the element. Please note that the nodes are not equally spaced. In particular, higher orders feature very small distances near the corners which heavily influence the maximum time step (see Sect. 3.1.2).

Lagrange polynomials and nodal points per element. The coefficients $a^{(i)}_{k,l,m}$ are entirely determined by the node positions and Eq. (8).

Adjusting p allows us to locally control the error of the spatial discretisation. An example of Lagrange polynomials for $p = 2$ and two-dimensional systems is given by Fig. 3. Table 3 shows numbers of nodal points and their distribution on the element boundaries for typical orders.

For one-dimensional problems analytical formulae exist to create both optimal node positions and a set of matching Lagrange polynomials. In two and three dimensions, the situation is more involved and one can employ the empirical Warp&Blend method to generate two- and three-dimensional point sets from one-dimensional ones [42]. In principle, more optimised point sets might slightly improve the interpolation error [43], but for most practical calculations the nodes generated via the Warp&Blend method perform sufficiently well.

Table 3 Number of nodes in one-, two-, and three-dimensional simplex elements. p denotes the polynomial order, n the total number of nodes, b the number of nodes on the element's boundary, and i the number of nodes within the element. For higher dimensions, the majority of the points reside on the element boundary.

	1D			2D			3D		
p	n	b	i	n	b	i	n	b	i
1	2	2	0	3	3	0	4	4	0
2	3	2	1	6	6	0	10	10	0
3	4	2	2	10	9	1	20	20	0
4	5	2	3	15	12	3	35	34	1
5	6	2	4	21	15	6	56	52	4
6	7	2	5	28	18	10	84	74	10
7	8	2	6	36	21	15	120	100	20

For simplicity, we assume $\mathcal{Q}(\vec{\mathrm{r}})$ to be constant within each element. Inserting the field expansion (7) into Eq. (5) leads to a separation of time-dependent expansion coefficients and position-dependent basis functions. The occurring integrals simplify to

$$\left(\mathscr{M}^{\triangle}\right)_{ij} = \int_{V_{\triangle}} L_i(\vec{\mathrm{r}}) \cdot L_j(\vec{\mathrm{r}})\, \mathrm{d}^3 r,$$

$$\left(\mathscr{S}^{\triangle}_k\right)_{ij} = \int_{V_{\triangle}} L_i(\vec{\mathrm{r}}) \cdot \partial_k L_j(\vec{\mathrm{r}})\, \mathrm{d}^3 r, \qquad k = x, y, z,$$

$$\left(\mathscr{F}^{\triangle}_f\right)_{ij} = \int_{f} L_i(\vec{\mathrm{r}}) \cdot L_j(\vec{\mathrm{r}})\, \mathrm{d}^2 r, \qquad \vec{\mathrm{r}}_j \in \text{face } f.$$

Here, we have introduced the mass matrix $\mathscr{M}^{\triangle}$, the stiffness matrices $\mathscr{S}^{\triangle}_k$ along the coordinate axes k, and the face mass matrices $\mathscr{F}^{\triangle}_f$ with respect to the element face f. Solving for

the time-derivatives yields

$$\partial_t \tilde{\vec{\mathrm{E}}}^{\triangle} = \frac{1}{\varepsilon^{\triangle}} \left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \left(\vec{\mathscr{S}}^{\triangle} \times \tilde{\vec{\mathrm{H}}}^{\triangle} + \mathscr{F}_f^{\triangle} \cdot \frac{\alpha \left[\Delta\tilde{\vec{\mathrm{E}}}_f^{\triangle} - \hat{\mathrm{n}}(\hat{\mathrm{n}} \cdot \Delta\tilde{\vec{\mathrm{E}}}_f^{\triangle})\right] + Z^{+} \hat{\mathrm{n}} \times \Delta\tilde{\vec{\mathrm{H}}}_f^{\triangle}}{\bar{Z}} \right), \tag{10a}$$

$$\partial_t \tilde{\vec{\mathrm{H}}}^{\triangle} = \frac{1}{\mu^{\triangle}} \left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \left(-\vec{\mathscr{S}}^{\triangle} \times \tilde{\vec{\mathrm{E}}}^{\triangle} + \mathscr{F}_f^{\triangle} \cdot \frac{\alpha \left[\Delta\tilde{\vec{\mathrm{H}}}_f^{\triangle} - \hat{\mathrm{n}}(\hat{\mathrm{n}} \cdot \Delta\tilde{\vec{\mathrm{H}}}_f^{\triangle})\right] - Y^{+} \hat{\mathrm{n}} \times \Delta\tilde{\vec{\mathrm{E}}}_f^{\triangle}}{\bar{Y}} \right). \tag{10b}$$

To shorten the notation, we have introduced

$$\vec{\mathscr{S}}^{\triangle} = \left(\mathscr{S}_x^{\triangle}, \mathscr{S}_y^{\triangle}, \mathscr{S}_z^{\triangle}\right)^T,$$

the vector of stiffness matrices, and $\Delta\tilde{\vec{\mathrm{E}}}_f^{\triangle}$ and $\Delta\tilde{\vec{\mathrm{H}}}_f^{\triangle}$ as vectors of differences of expansion coefficients across the face f of element $\triangle$. Please note that the inverse mass matrix and the face mass matrix are applied to physical vectors with x-, y-, and z-components, where each component itself is a vector of expansion coefficients (see Table 1). To such three-dimensional vectors, these matrices appear like scalar coefficients, i. e.,

$$\left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \tilde{\vec{\mathrm{a}}} \equiv \begin{pmatrix} \left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \tilde{\mathrm{a}}_x \\ \left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \tilde{\mathrm{a}}_y \\ \left(\mathscr{M}^{\triangle}\right)^{-1} \cdot \tilde{\mathrm{a}}_z \end{pmatrix}.$$

Eq. (10) is the semi-discrete formulation of Maxwell's curl equations. As compared to the original equations (1), the spatial dependence has been absorbed into a set of expansion coefficients and corresponding matrices. However, the expansion coefficients still depend on time. Strategies which either directly discretise the time in terms of time steps or take the Fourier transform to obtain a frequency-domain formulation are topics of Sect. 3.

2.6. Remarks regarding efficiency

Eq. (10) looks quite involved, especially in comparison with expressions obtained by standard finite-difference discretisations. It contains a number of matrices and even the inverse of the mass matrix. This section will explain why the discontinuous Galerkin discretisation is an efficient choice after all.

First of all, let us consider the dimensions of the matrices involved. The mass matrix $\mathscr{M}^{\triangle}$ is an $n \times n$ matrix, and so are the stiffness matrices $\mathscr{S}_k^{\triangle}$. We recall that n is the number of nodes per element in d dimensions for order p. The face mass matrices are $n \times n'$ matrices, where n' is the number of points on each face which coincides with the total number of nodes for dimension $d-1$ and order p. Typical values of n and n' for various orders and dimensions can be found in Table 3. In most cases, n and n' will be approximately 20. Such small matrices are easy to invert. Furthermore, they are easily stored in memory. The small matrices are an immediate consequence of the discontinuous basis functions. Conventional finite element methods yield large, sparse matrices which cannot be easily inverted or stored subsequently.

In most cases, it is not even necessary to store these matrices. Similarly to classical finite element methods, one computes the matrices on a reference element once. All other elements are mapped on this reference by affine transformations. However, this procedure is applicable for straight-sided simplices as shown in Fig. 1 only (for curvilinear elements please refer to Sect. A.5). The respective transformation follows immediately from the vertices of the reference element and the target element and does not depend on the order of the polynomial basis. The matrices on the target element are composed of scaled reference matrices, where the scaling factors consist of entries of the Jacobi matrix of the transformation. This is also valid for the inverse mass matrix. Hesthaven and Warburton provide technical details on the mapping to a reference element in [26, 27].

Hence, instead of storing a number of matrices with around 20×20 entries each for every single element, we just store the 3×3 Jacobi matrix for each element. As the reference matrices are usually small enough to be conveniently kept in the CPU cache, it is even faster to construct the matrices on a target element when needed. We conclude that we can efficiently evaluate the right-hand side of Eq. (10), because all the matrices are explicitly known and available.

The price we pay is that on each interface between neighbouring elements we store *two* distinct sets of field values; one for either element. Table 3 compares n, the total number of nodal points, with b, the number of nodal points on the element's boundary. In addition, the number of internal, non-boundary points i is listed. For two- and three-dimensional systems, most nodes are boundary nodes. Thus, we store most of the points twice. Higher orders reduce this overhead, but yield larger matrices. Nevertheless and especially in nanophotonics, higher orders are often favourable as will be discussed in the upcoming sections.

2.7. Error of the spatial discretisation

For numerical simulations it is essential to control the error of the approximate solution with respect to the correct solution of the original equation. In our case, we can increase the accuracy of the simulations by either decreasing the element size h or by increasing the polynomial degree p. For sufficiently smooth solutions and an upwind flux ($\alpha \equiv 1$), the deviation between the numerical and the analytical solution

is given by

$$\left\| \mathbf{q}^{\mathrm{ref}} - \mathbf{q}^{\mathrm{num}} \right\| = \mathscr{O}\left(h^{p+1}\right) . \tag{11}$$

Here, $\mathbf{q}^{\mathrm{ref}}$ represents the (not necessarily known) exact reference solution of the original, non-discretised equation, while $\mathbf{q}^{\mathrm{num}}$ is the numerical one. Given some arbitrary norm $\|\cdot\|$, the convergence is algebraic in h and exponential in p. Setting p, we can choose how fast the error shall decrease with the mesh size, with the cost of additional operations per element. In principle, both h and p can be different for each element (hp-adaptivity). In practice, it is convenient to use the same p for all elements and vary h locally. To control the error, one can globally increase p (p-refinement) or reduce h either globally or locally (h-refinement). Finding a balance between accuracy and computational effort is not always easy. However, in most of our calculations we find that orders of $p \in [3,4,5,6]$ offer a good compromise. For the mesh size h we then typically choose a maximal edge length which roughly corresponds to half of the wavelength of interest.

2.8. Advantages of the spatial discontinuous Galerkin discretisation

The spatial discretisation via the discontinuous Galerkin method has a number of merits of particular interest to nanophotonics. First of all, the method does not rely on regular grids like the FDTD method, but on flexible meshes with elements of various sizes and shapes instead. This freedom basically allows us to "adapt" the algorithm to a specific system. In particular, it is straightforward to handle non axis-aligned or curved material interfaces. Locally increasing the spatial resolution (via h- or p-refinement) allows to accurately model strong local variations of electromagnetic fields, e. g., near metallic surfaces. At the same time, one can save a substantial amount of degrees of freedom in regions with weakly varying fields. This is usually the case in the area surrounding a nanophotonic device with features often smaller than the wavelength of interest. Therefore, the DG method lends itself for multi-scale analysis.

Secondly, since the method employs discontinuous basis functions, discontinuities in the electromagnetic fields are inherently dealt with. Accuracy and convergence issues due to inhomogeneous material distributions as present in FDTD calculations are absent in DG calculations. At the same time, the special choice of basis functions also allows us to treat the elements separately. As a result, the DG discretisation lends itself for parallel computations on multiple cores, multiple computers, and graphic cards. Especially higher polynomial orders p result in excellent parallel performance, since the ratio of local-element operations and inter-element communication is more favourable.

Simultaneously, high interpolation orders also lead to very faithful representations of both magnitudes and phases of propagating electromagnetic fields [27]. Resonators, interferometers, and other, potentially nonlinear, phase-sensitive systems often encountered in nanophotonics greatly benefit from the accuracy of higher-order methods [44].

3. Solving the semi-discrete problem

Starting from Maxwell's curl equations, we have employed a discontinuous Galerkin technique to discretise the electromagnetic fields in space. To simplify the notation, we rewrite Eq. (10) as

$$\partial_t \begin{pmatrix} \tilde{\vec{\mathrm{E}}}(t) \\ \tilde{\vec{\mathrm{H}}}(t) \end{pmatrix} = \mathscr{H} \cdot \begin{pmatrix} \tilde{\vec{\mathrm{E}}}(t) \\ \tilde{\vec{\mathrm{H}}}(t) \end{pmatrix} + \begin{pmatrix} \tilde{\vec{\mathrm{E}}}^{\mathrm{source}}(t) \\ \tilde{\vec{\mathrm{H}}}^{\mathrm{source}}(t) \end{pmatrix} \tag{12a}$$

or, more concisely,

$$\partial_t \tilde{\mathbf{q}}(t) = \mathscr{H} \tilde{\mathbf{q}}(t) + \tilde{\mathbf{q}}^{\mathrm{source}}(t) . \tag{12b}$$

The system operator $\mathscr{H}$ acting on the current expansion coefficients represents the right-hand sides of Eq. (10). Please note that the expansion coefficients comprise all elements, and not just a single one. Furthermore, we have included source terms, which were absent in Eq. (10), and labelled them accordingly. Section A.1 will explain how to obtain these terms for a number of physical situations.

Equation (12) states a system of coupled, first-order, ordinary differential equations for the expansion coefficients of the electromagnetic field. There are two possibilities to solve this system. Either we discretise time itself and evolve initial fields in time, or we restrict ourselves to time-harmonic problems. Both approaches are discussed in Sects. 3.1 (time-domain) and 3.2 (frequency-domain), respectively.

3.1. The discontinuous Galerkin time-domain method

Before we come to the details on how to obtain the Discontinuous Galerkin Time-Domain (DGTD) method, let us outline a few general properties of time-domain methods first.

Time is a quantity human beings have an intuitive feeling for. Being accustomed to the concept that the present translates into the future, time-domain simulations can help us see the causes of visible effects. For example, given a simulation with multiple scatterers we can recognise the dominant scatterer. Such knowledge might help us to improve a design in order to reduce scattering losses. Animations of time-dependent field distributions most easily illustrate the behaviour of a physical system even for non-experts in the field (or even the general area of science).

At the end of the day, for a comparison with experiments we need to obtain spectral information about a system. Simulating the same system over and over again with plane waves of different frequencies is usually not a good idea. Instead, one launches ultra-short pulses into the system. Such pulses possess a very broad frequency spectrum. Using the

on-the-fly Fourier transform (Sect. A.6.1), one can obtain a spectrum over the complete bandwidth in a single simulation.

Simulating in the time-domain also allows us to naturally include nonlinearities. The latter prohibit the superposition of solutions. In particular, the superposition of two electromagnetic waves will lead to sum and difference frequency generation. As a result, nonlinear phenomena cannot easily be treated with frequency-domain methods.

Last but not least, time-domain methods are usually quite memory-efficient. In most cases, it is sufficient to store one or two vectors of expansion coefficients plus information about the physical system. Both the memory requirements and the computational time scale linearly with the number of degrees of freedom. Thus, time-domain methods can handle large systems even on computers with limited resources.

On the other hand, time-domain simulations have a number of conceptual drawbacks which include the comparison with real-world experiments, long computation times for high-quality resonators, and a non-trivial treatment of dispersive materials. Techniques to alleviate and dispose of these drawbacks are presented in the appendix.

3.1.1. Time-stepping and the Runge-Kutta method

Equation (12) represents a set of ordinary differential equations of first order in time. To solve it numerically, we divide the time axis into a number of (not necessarily equidistant) time steps. The fields at the first time step t_0 define the initial condition. Usually, one assumes vanishing fields but other initial conditions, e. g., cavity modes, are not uncommon. Using a time stepping scheme, we evolve the fields from t_n to t_{n+1}, where n indicates the time step. Let us consider the equation

$$\partial_t \tilde{\mathbf{q}}(t) = \tilde{\mathbf{f}}(\tilde{\mathbf{q}}(t), t), \tag{13}$$

which has the shape of Eq. (12). The choice of the time stepping scheme crucially affects the performance. In essence, on can either choose explicit or implicit methods. For explicit methods we need to evaluate the right-hand side $\tilde{\mathbf{f}}(\tilde{\mathbf{q}}(t), t)$, depending on the scheme even multiple times per time step. Prominent solvers are Adams-Bashforth and Runge-Kutta methods. However, if the time step is too large, then explicit solvers yield numerically unstable, i. e., exponentially growing, solutions. In contrast, implicit methods require the solution of a system of equations for each time step, but in turn provide unconditionally stable results. Nevertheless, the error increases with the time step size. As solving a system of equations is computationally expensive, time stepping with implicit solvers is often less efficient than using explicit solvers.

In the case of DGTD, the time evolution of the electromagnetic fields is most conveniently accomplished using low-storage Runge-Kutta (LSRK) methods for a number of reasons. First, it is desirable to accompany the higher-order accurate DG discretisation in space with a higher-order accurate time stepping scheme. Suitable LSRK schemes are available for up to fourth order. Secondly, given a total number of N expansion coefficients, LSRK methods require a total of $2N$ values stored in two registers $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{p}}$. For one time step from t_n to t_{n+1}, we have to adhere to the algorithm

$$\begin{aligned}
\tilde{\mathbf{q}}_0 &:= \tilde{\mathbf{q}}(t_n) \\
\tilde{\mathbf{p}}_i &:= A_i \cdot \tilde{\mathbf{p}}_{i-1} + \Delta t \cdot \tilde{\mathbf{f}}(\tilde{\mathbf{q}}_{i-1}, t_n + c_i \Delta t) && \text{(14a)} \\
\tilde{\mathbf{q}}_i &:= \tilde{\mathbf{q}}_{i-1} + B_i \cdot \tilde{\mathbf{p}}_i && \text{(14b)} \\
\tilde{\mathbf{q}}(t_{n+1}) &:= \tilde{\mathbf{q}}_s.
\end{aligned}$$

Steps (14a) and (14b) define the stages $i = 1 \ldots s$, where s is the number of stages. Together with this parameter, the coefficients A_i, B_i, and c_i define the properties (order of accuracy, stability contour) of the LSRK scheme. It is noteworthy, however, that the storage requirements of the algorithm neither depends on the order of accuracy nor the number of stages. The five stage, fourth order accurate LSRK scheme by Carpenter and Kennedy [45] is most commonly used for the field evolution in DGTD [27]. Nevertheless, other choices [46, 47] are possible and potentially advantageous. Hence, they are discussed in Sect. 3.1.3.

3.1.2. Eigenvalues, conditional stability and maximum time steps

Being explicit methods, LSRK schemes are subject to conditional stability, also known as the Courant-Friedrichs-Lewy condition (CFL condition). As soon as the time step Δt exceeds a critical time step, the numerical solution is subject to unphysical exponential growth. The critical time step, for which the numerical solution just does not grow exponentially, depends both on the time stepping scheme and on the system of equations to be integrated.

Let us first consider the influence of a specific LSRK scheme, which is characterised by the characteristic polynomial (or amplification factor)

$$R(z) = 1 + \gamma_1 z + \gamma_2 z^2 + \cdots + \gamma_s z^s, \quad z \in \mathbb{C}.$$

Here, the coefficients γ can be directly related to the coefficients A_i and B_i of the LSRK scheme [47]. In the absence of sources, the discretised physical system can be expressed as a matrix-vector product of a system matrix $\mathscr{H}$ and a number of unknowns, compare Eq. (12). We can only obtain a stable time-integration if the necessary condition

$$|R(\Delta t \cdot \lambda_i)| \leq 1$$

holds for *all* eigenvalues λ_i of $\mathscr{H}$. Geometrically speaking, the curve $|R(z)| = 1$ defines a stability contour in the complex plane. The complex eigenvalue spectrum of $\mathscr{H}$ can be scaled by the time step Δt until it is completely enclosed by the stability contour. This is illustrated in Fig. 5. The first time step for which this condition holds is called the maximum stable time step $\Delta t_{\max}$. It should be noted that this time step does not *guarantee* stability, since its derivation only relies on a necessary condition. Nevertheless, it serves

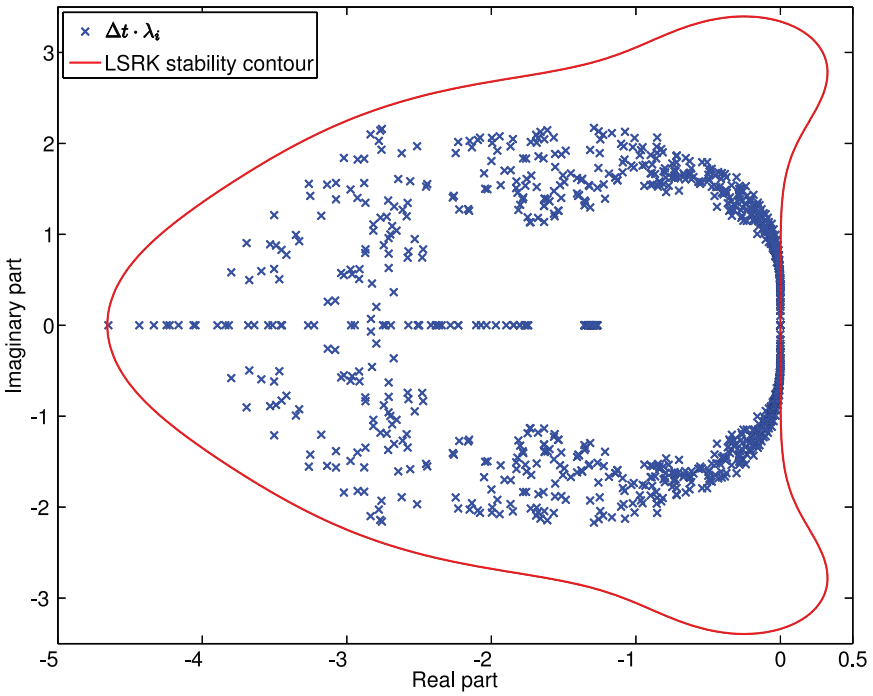


Figure 5 (online color at: www.lpr-journal.org) Low-storage Runge-Kutta stability contour and eigenvalues of a two-dimensional cavity. The red contour is the stability contour of the five-stage, fourth-order accurate low storage Runge-Kutta scheme as introduced by Carpenter and Kennedy. The blue crosses indicate eigenvalues λ_i of the system operator $\mathscr{H}$, which have been rescaled by the maximum time step Δt. If Δt were larger, at least one rescaled eigenvalue would lie outside of the stability contour. As a result, the numerical simulation would be numerically unstable.

as a good approximation to the time step obtained from a sufficient condition [26, 48].

Depending on the shape of the eigenvalue spectrum, the scaling factor required to squeeze it into the stability contour is different. Hence, it is worth to investigate the typical eigenvalue spectrum of our discretised Maxwell operator $\mathscr{H}$ as defined by Eq. (10). The spectrum is subject to a number of influences and can be tuned by changing the upwind parameter α. As discussed in Sect. 2.3, $\alpha = 0$ reduces the upwind flux to the energy-conserving central flux. Consequently, the spectrum only has purely imaginary values, provided the physical system does not include dissipative materials or absorbing boundary conditions. As α increases, the eigenvalues gain a negative real part. This effectively damps modes which do not fulfil the vanishing divergence condition or are not sufficiently resolved by the spatial discretisation. Auxiliary fields, for example to implement perfectly matched layers or dispersive materials (see appendix), further modify the spectrum.

The size of the time step is strongly related to the quality of the spatial discretisation as well. Similarly to the popular FDTD algorithm, where the maximum time step is governed by

$$\Delta t_{\max} \leq \frac{\text{mesh width}}{c},$$

the smallest distance between nodes within an element influences the DGTD time step. The higher the order and the more deformed an element is, the smaller the maximum time step will be. In practice, one can use the estimate

$$\Delta t_{\max} = s \cdot d_{\min}(p) \cdot \min_{\triangle} \left(r_{\text{in}}^{\triangle} \right), \qquad (15)$$

where $r_{\text{in}}^{\triangle}$ is the radius of the insphere of element $\triangle$, $d_{\min}(p)$ is smallest distance between two nodes on an edge of the reference element, and s is a constant factor of the order of 1. This heuristic approach works for a wide range of applications. However, it rarely provides the optimal time step. Especially for higher orders $p > 4$, the maximally allowed timestep is often underestimated and therefore performance is lost. The estimate (15) can be improved by making the factor s order-dependent and use a fitting to empirical data to improve the performance. As an example, in [49], a p-dependence of

$$s(p) = 0.8 + 0.27p - 0.011p^2$$

was proposed.

For a more advanced, system-dependent estimate of the optimal time step, one can calculate a few extremal eigenvalues of the system operator using an iterative eigenvalue solver such as `ARPACK` [50]. From these eigenvalues one can infer a scaling factor such that they lie within the stability contour of some given algorithm. Unfortunately, the computational cost associated with finding a sufficient number of eigenvalues can be quite considerable for large systems. Therefore, this approach is only feasible for medium-sized problems.

3.1.3. Optimised Runge-Kutta schemes

The dependence of the time step on the shape matching between the eigenvalue spectrum and the Runge-Kutta stability contour suggests that performance gains can be expected from optimised Runge-Kutta schemes. For memory efficiency we restrict ourselves to LSRK schemes. The parameters A_i, B_i, and c_i are subject to a number of conditions stemming from the desired order of accuracy and the low-storage property. Increasing the number of stages s above

the order introduces additional free parameters. As they affect the characteristic polynomial, they can be used to modify the shape of the stability contour.

To compare the performance of different schemes, it is worthwhile to consider the normalised time step

$$\Delta\bar{t} = \frac{1}{s} \cdot \Delta t \,.$$

The normalised time step represents the fraction of the total time step which can be attributed to a single stage. This is a reasonable measure because each stage requires one evaluation of the right-hand side $\tilde{\mathbf{f}}(\tilde{\mathbf{q}}_{i-1}, t_n + c_i \Delta t)$. Hence, the normalised time step also represents a measure for time step per computational effort (CPU time). Diehl *et al.* have compared over 50 established RK schemes to the commonly used Carpenter-Kennedy scheme [46]. For fourth-order accurate schemes, only marginal performance improvements could be found for α close to zero. For $\alpha \approx 0.6$, the five-stage third-order scheme by Spiteri and Ruuth [51] yields a speed-up factor of up to 1.6.

Very recently, Diehl *et al.* developed a numerical optimisation method to generate LSRK schemes with tailored stability regions [47]. With this approach, a fourth-order 14-stage scheme was generated, which outperforms the 5-stage scheme by Carpenter and Kennedy by around 40–50%. It is remarkable that these performance gains can be achieved by merely changing a couple of numbers in an existing DGTD implementation, which typically features several thousand lines of code.

3.1.4. Advantages and drawbacks

We conclude the discussion of DGTD by a short comparison with other existing time-domain methods. Let us start with the most established one. Compared to FDTD, DGTD requires significantly more computations per degree of freedom because we have a number of small matrix-vector products within each element. At the same time, DGTD computations usually require less degrees of freedom to obtain similarly converged results. This is an immediate consequence of the superior *h*- and *p*-refinement capabilities of DGTD. Small elements can be used to resolve small geometrical features. Even curvilinear elements (see Sect. A.5) can be employed where necessary. Depending on the system, DGTD can be considerably more efficient concerning both memory and CPU time than FDTD [44].

The finite volume (FVTD) and finite element time-domain (FETD) methods are conceptually quite similar to DGTD. In fact, the DGTD method discussed here reduces to a classical FVTD scheme for zeroth order ($p = 0$). However, in contrast to traditional finite volume methods, the DGTD method also offers a systematic approach to increase the order of the spatial discretisation to an arbitrarily high polynomial degree. This is also a big advantage over FETD, where the generation of higher order Nédélec elements [24, 25] is certainly not trivial. In addition, as mentioned before, a FETD approach requires the solution of a large (albeit sparse) system of linear equations at each time step. The main disadvantage of the DGTD method is that one has to deal with considerably more degrees of freedom due to the discontinuities across element interfaces. These discontinuities, however, also allow us to treat elements individually with a set of small matrix-vector multiplications. For this reason, DGTD is an ideal candidate for parallelisation on multiple cores, computers, and graphics cards [52].

To summarise, we find that DGTD is an explicit time-domain method which combines the efficiency of FDTD with unstructured meshes and higher order accuracy as known from finite element methods. Moreover, it can be combined with a number of time stepping algorithms, out of which customised low-storage Runge-Kutta algorithms appear particularly efficient schemes. As a consequence, DGTD can accurately simulate systems which involve fine geometric features, strongly varying local fields, and different length and time scales. Hence, DGTD is certainly an excellent method for the time-domain simulation of nanophotonic systems.

3.2. Discontinuous Galerkin frequency-domain methods

In the previous sections, we have introduced techniques to discretise the time derivatives in Eq. (12). The current section follows a different approach. Instead of looking at the full dynamics of the system, we restrict ourselves to time-harmonic solutions in the frequency domain.

Time-harmonic solutions are interesting for a number of applications. Experiments often show more or less pronounced spectral resonances. The field distribution at such a resonance may explain its physical origin. When coupling from a waveguide to another device, e. g., a photonic crystal waveguide, it is important to match the mode profiles in both devices for maximum transmittance. In such cases, the system needs to be characterised for only a few frequencies. Even in cases where we want to calculate a full spectrum a time-harmonic simulation often proves advantageous. This is especially the case for systems with high quality factors (Q-factors). Such systems require very long time-domain simulations in order to excite a resonance or let it decay. Spurious results are often obtained from prematurely terminated simulations.

Moreover, time-harmonic computations can include dispersion in a natural way. Instead of using analytical dispersion models (see Sect. A.2) one can immediately resort to experimental data. There are also a couple of frequency-dependent excitation profiles such as focussed laser beams and waveguide modes, which introduce further difficulties in the time-domain.

Hence, time-harmonic solvers provide an alternative route to obtain numerical results for a multitude of systems. Depending on the situation, either frequency-domain or time-domain simulations are better suited. In some cases, a time-domain simulation is used to calculate a broad-band spectrum. Once the resonances are identified, time-harmonic simulations are employed to evaluate mode profiles.

It should be noted that in this article we want to solve the very same *first-order* system as in the time-domain. In particular, we will not derive a wave equation for one of the electromagnetic fields as, for example, done in [53]. Therefore, our approach will not lead to optimal performance, but to maximum consistency with an existing time-domain code. In this sense, the frequency-domain algorithms as described in the upcoming sections complement the time-domain solver.

3.2.1. Transformation to the frequency domain

To transform the semi-discrete form (12) into the frequency-domain, we make the time-harmonic ansatz

$$\tilde{\mathbf{q}}(t) \to \tilde{\mathbf{q}}(\omega) \cdot e^{-\mathrm{i}\omega t} . \tag{16}$$

The (angular) frequency is denoted by ω. Please note that $\tilde{\mathbf{q}}(t)$ and $\tilde{\mathbf{q}}(\omega)$ are radically different functions and should not be confused with each other. As opposed to the usual quantities, $\tilde{\mathbf{q}}(\omega)$ is a vector of frequency-dependent expansion coefficients, where each expansion coefficient is a *complex* number. The physical solution for a specific time t is given by the real part of (16). For the time-derivative, this ansatz translates to

$$\partial_t \tilde{\mathbf{q}}(t) \quad \leftrightarrow \quad -\mathrm{i}\omega\tilde{\mathbf{q}}(\omega) . \tag{17}$$

Applying this ansatz to the sources as well and inserting it into Eq. (12) leads to

$$-\mathrm{i}\omega\tilde{\mathbf{q}}(\omega) = \mathscr{H}\tilde{\mathbf{q}}(\omega) + \tilde{\mathbf{q}}^{\mathrm{source}}(\omega) . \tag{18}$$

3.2.2. Eigenvalue problems

Let us first consider Eq. (18) without external sources, i. e.,

$$\tilde{\vec{\mathrm{E}}}^{\mathrm{source}}(\omega) \equiv \tilde{\vec{\mathrm{H}}}^{\mathrm{source}}(\omega) \equiv 0 .$$

In this case, we obtain

$$\mathscr{H} \cdot \begin{pmatrix} \tilde{\vec{\mathrm{E}}}(\omega) \\ \tilde{\vec{\mathrm{H}}}(\omega) \end{pmatrix} = -\mathrm{i}\omega \begin{pmatrix} \tilde{\vec{\mathrm{E}}}(\omega) \\ \tilde{\vec{\mathrm{H}}}(\omega) \end{pmatrix} . \tag{19}$$

This equations describes an eigenvalue problem of the system operator $\mathscr{H}$, where the eigenvalues are the complex frequencies $-\mathrm{i}\omega$ and the corresponding eigenvector consists of the field expansion coefficients. Thus, ω is the result of an eigenvalue problem.

Though this problem looks simple enough, its numerical solution using iterative eigenvalue solvers such as `ARPACK` [50] is rather challenging. In particular, the spectrum of the discretised Maxwell operator $\mathscr{H}$ features a large number of spurious, i. e., unphysical modes. For example, Fig. 5 shows the eigenvalues $\lambda = -\mathrm{i}\omega$ of $\mathscr{H}$ for a two-dimensional rectangular cavity. In principle, all the eigenvalues should lie on the imaginary axis, i. e., they should correspond to real-valued frequencies ω. In particular, there should be a minimum frequency which translates to the largest possible wavelength in the system.

Instead, we observe eigenvalues with negative real parts which appear due to the artificial damping introduced by the upwind flux (see Sect. 2.3). It can be difficult to distinguish the associated unphysical modes from physical modes if absorbing materials are present in the system. Furthermore, we even find eigenvalues on the real axis which correspond to purely complex ω. In particular, we find a very large number of zero eigenvalues, which correspond to the null space of the DG operator and belong to stationary solutions of Maxwell's curl equations with non-vanishing divergence.

As a result, iterative eigenvalue solvers have difficulties to find physical eigenvalues. In particular, the eigenvalues with the smallest frequencies are difficult to find because of the large spurious null space. On the other hand, eigenvalues with the largest magnitudes correspond to unphysical modes introduced by the spatial discretisation. Even when using the shift and invert transformation to find eigenvalues near a complex value σ, great care on the choice of σ has to be taken in order to obtain physical eigenvalues. The underlying reason for the formation of spurious modes is that the DG discretisation does not enforce the *global* divergence condition. To reduce the number of spurious modes, one could use a *locally* divergence-free basis on each element [54]. Combining such a scheme with a suitable numerical flux significantly reduces the null space and helps to distinguish physical modes from spurious ones for certain systems [26]. Nevertheless, calculating and identifying relevant eigenvalues still remains a non-trivial problem, for which the DG method is not ideally adapted.

3.2.3. Time-harmonic solutions

A time-harmonic solver can be used to calculate the field distribution for a specific, frequency-dependent illumination characterised by $\tilde{\mathbf{q}}^{\mathrm{source}}(\omega)$. In this case, we rewrite Eq. (18) as

$$(\mathscr{H} + \mathrm{i}\omega) \cdot \tilde{\mathbf{q}}(\omega) = -\tilde{\mathbf{q}}^{\mathrm{source}}(\omega) . \tag{20}$$

Apparently, the desired field distribution is the solution of a system of linear equations. Its right-hand side is given by the sources while the equations are characterised by the DG system matrix plus a diagonal matrix, which effectively shifts the diagonal of $\mathscr{H}$. In contrast to the eigenvalue problem, ω enters the calculation as a parameter, which determines the oscillation frequency of an external source.

It is worthwhile to consider the structure of the system matrix $\mathscr{H}' = \mathscr{H} + \mathrm{i}\omega$, whose detailed discussion is postponed to Sect. 3.2.4. For the time being, it is sufficient to know that $\mathscr{H}'$ is a non-symmetric sparse matrix where almost all matrix elements are zero. For such matrices, a number of efficient solver techniques are available and can be roughly divided into direct and iterative solvers.

Direct solvers attempt to decompose $\mathscr{H}'$ into a product of a lower triangular matrix L and an upper triangular matrix U. The main challenge is not to lose the sparsity property of $\mathscr{H}'$, i. e., to minimise the number of non-zero entries in

both L and U. Once the triangular matrices are known, the problem

$$\mathscr{H}'\tilde{\mathbf{q}} = \tilde{\mathbf{q}}^{\text{source}} \tag{21}$$

reduces to the – then trivial – forward and backward substitutions

$$L\tilde{\mathbf{p}} = \tilde{\mathbf{q}}^{\text{source}}$$
$$U\tilde{\mathbf{q}} = \tilde{\mathbf{p}}.$$

A number of fast direct solvers for unsymmetric matrices are available, such as `PARDISO` [55] and `UMFPACK` [56]. However, those are rather memory consuming and therefore limit the number of degrees of freedom one can handle. In practice only medium-sized systems with $N \approx 10^5$–10^6 unknowns can be treated when using direct solvers on typical computers.

In contrast, iterative solvers do not attempt to factorise $\mathscr{H}'$. Instead, starting from an initial guess $\tilde{\mathbf{q}}_0$ they try to reduce the residuum $\mathscr{H}'\tilde{\mathbf{q}}_i - \tilde{\mathbf{q}}^{\text{source}}$ of the solution in an iterative procedure. The system enters the scheme only via matrix-vector products between $\mathscr{H}'$ and some (temporary) vectors. In particular, storage of $\mathscr{H}'$ is not required as long as its effect on an arbitrary vector can be easily computed. As discussed in Sects. 2.6 and 3.1, an efficient matrix-vector product is the basis of the time-domain algorithm. The only differences to the time-domain are the complex-valued expansion coefficients and the diagonal shift, which can be implemented as a simple scaling. As a consequence, iterative methods are very memory-effective when combined with an existing DGTD code, as only a small number of vectors of expansion coefficients need to be stored. On the downside, the number of iterations required to get sufficiently converged results can be quite high. Acceleration techniques are discussed in Sect. 3.2.5. Popular iterative solvers include `BiCGstab(l)` [57] and restarted `GMRES` [58].

It should be noted that the choice of the initial guess $\tilde{\mathbf{q}}_0$ is crucial for the performance of iterative solvers. The naïve choice of all expansion coefficients being zero often represents a poor guess. For scattering problems it can prove advantageous to use the fields as created by the time-harmonic source in the absence of scatterers as an initial guess. This is especially the case in the limit of weak scattering. For example, a plane wave can be a good start when one is interested in Mie scattering. Similarly, when simulating a waveguide coupled to a resonator, one can ignore the resonator and start with a waveguide mode instead.

Due to the impact of the initial set of expansion coefficients on the solution time, it is desirable to obtain a system-dependent, automated, and educated guess for the starting point. This can be done using multigrid methods. The idea is to simulate the very same system using a set of grids, where each grid is finer than the previous one. The problem on the coarsest grid can be solved using direct methods. As we do not use as many degrees of freedom as for the finest grid, the additional memory-consumption is usually acceptable. In the simplest version of multigrid methods, the solution of the coarser grid is interpolated (or prolongated) to the next finer one. There, an iterative solver is applied to reduce the residual error. The procedure is reapplied until the desired level of (geometrical) accuracy is reached.

Finer grids can be created in two ways. First, one can refine the mesh, i. e., the size of the elements (h-refinement). Second, one can increase the number of degrees of freedom per element by using higher-order basis functions (p-refinement). Since h-refinement requires additional algorithms and p-refinement is one of DG's strengths, the latter is often the favourable choice. In our simulations, good initial guesses are usually obtained with a direct solver for $p = 2$.

A promising approach to the solution of the time-harmonic problem (20) is to combine direct and iterative solvers. For example, one can apply domain decomposition techniques and divide the computational domain into smaller subdomains [59]. Such a decomposition introduces virtual interfaces between neighbouring subdomains. Starting from an initial guess, one can exploit the continuity of fields to derive effective boundary conditions on these virtual interfaces. This effectively reduces the problem of solving Maxwell's equations on the entire domain to the problem of solving these equations on several smaller subdomains with well-defined boundary conditions. Provided that the subdomains are small enough, direct solvers can efficiently solve the resulting systems of linear equations. In an iterative procedure, the boundary conditions on the virtual interfaces are updated and a new solution on each subdomain is computed using direct solvers. Hence, the domain decomposition technique combines the strengths of iterative solvers (low memory consumption) and direct solvers (very fast) and allows to tackle even large three-dimensional problems [59]. Unfortunately, this comes at the price of increased code complexity, which is why we will not discuss this approach any further in this review.

3.2.4. The structure of the system matrix

As noted earlier, it is interesting to study the structure of the system matrix $\mathscr{H}' = \mathscr{H} + \mathrm{i}\omega$. Looking at the semi-discrete form of Eq. (10), $\mathscr{H}'$ has the following contributions:

- The degrees of freedom *within* an element $\triangle$ are coupled to each other via the discretised curl operator. The occurring spatial derivative matrices $\mathscr{M}^{-1}\mathscr{S}$ are square $n \times n$ matrices, where n is the number of nodes per element (see Sect. 2.5). Please note that these matrices are not symmetric. This is an immediate consequence of the first-order form of Maxwell's equations we want to solve.
- The numerical flux introduces some coupling *within* the element as well as some coupling to the *neighbouring* elements via the field differences. The matrices $\mathscr{M}^{-1}\mathscr{F}_f$ are of size $n \times n'$, where n' is the number of points per face (again, see Sect. 2.5).
- The term $\mathrm{i}\omega$ translates to a diagonal shift in the system matrix.
- Finally, some extensions for practical use (see appendix) may introduce auxiliary fields which are coupled to the

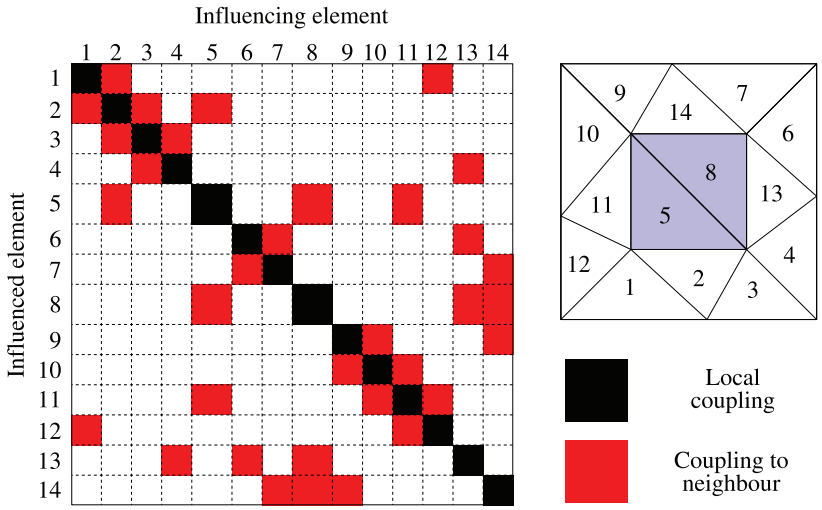


Figure 6 (online color at: www.lpr-journal.org) Sketch of the complete system matrix for a two-dimensional cavity as a sample system. The matrix $\mathscr{H}'$ on the left-hand side consists of a number of rectangular blocks, each of which couples degrees of freedom of a certain element (column) to those of another element (row). However, just a few of these blocks actually feature non-zero entries. Blocks with at least one non-zero entry are coloured black (if the block lies on the diagonal of $\mathscr{H}'$) or red (the block is off-diagonal). The right-hand side features the corresponding mesh used to create the system matrix. Of the fourteen numbered elements, elements 5 and 8 support more degrees of freedom due to being filled with a dispersive material (see Sect. A.2).

electromagnetic fields. If such a coupling is purely local, this corresponds to a diagonal $n \times n$ matrix within $\mathscr{H}'$. We see that matrix elements either connect degrees within an element or they link neighbouring elements with the local one (Fig. 6). Hence, the coupling is local. Especially in the limit of many elements, this leads to a sparse matrix, where almost all elements are zero. Even more, as the number of neighbours is limited to $d+1$ for simplices, where d is the dimensionality of the system, the number of non-zero entries per degree of freedom is roughly constant. Thus, the number of matrix elements grows linearly with N, the number of degrees of freedom.

In addition, we note that all contributions can be written as special matrices with n rows each (see Fig. 7). Depending on the type of the coupling, those matrices are either dense (derivatives), diagonal (local coupling, diagonal shift), or have a column-shaped structure (numerical flux). In particular, the numerical flux renders the system matrix structurally non-symmetric. The block structure of $\mathscr{H}'$ can be exploited to obtain an efficient sparse matrix storage scheme with minimum index overhead.

A last remark concerns the ordering of the field expansion coefficients in $\tilde{\mathbf{q}}$. It appears natural to store all expansion coefficient of a single field component for a single element in a contiguous memory area. Doing so leaves us with the task of mixing field components and elements (see Fig. 8). In principle, one can either group expansion coefficients of one field component for all elements, i. e.,

$$\tilde{\mathbf{q}} = \left(\tilde{\mathrm{E}}_x^{\Delta 1}, \tilde{\mathrm{E}}_x^{\Delta 2}, \ldots, \tilde{\mathrm{E}}_y^{\Delta 1}, \tilde{\mathrm{E}}_y^{\Delta 2}, \ldots, \tilde{\mathrm{E}}_z^{\Delta 1}, \tilde{\mathrm{E}}_z^{\Delta 2}, \ldots\right) ,$$

or one can group all field components of a single element, i. e.,

$$\tilde{\mathbf{q}} = \left(\tilde{\mathrm{E}}_x^{\Delta 1}, \tilde{\mathrm{E}}_y^{\Delta 1}, \tilde{\mathrm{E}}_z^{\Delta 1}, \ldots, \tilde{\mathrm{E}}_x^{\Delta 2}, \tilde{\mathrm{E}}_y^{\Delta 2}, \tilde{\mathrm{E}}_z^{\Delta 2}, \ldots\right) .$$

The second version is preferable as it keeps element-local data together. This improves CPU cache usage and eases data handling for computations on graphics cards. In addi-

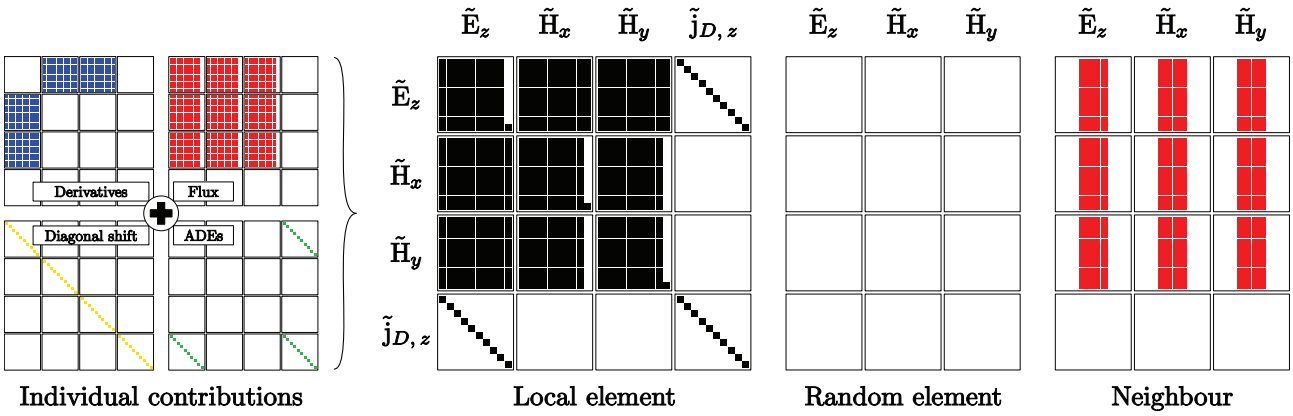


Figure 7 (online color at: www.lpr-journal.org) Details on the structure of the system matrix $\mathscr{H}'$. In principle, this figure is a more detailed view of row 8 of Fig. 6 for $p = 3$. In contrast to the latter, coloured rectangles indicate individual non-zero matrix entries here. The derivatives, the numerical flux, auxiliary differential equations and the diagonal shift contribute matrix values to the local block, which corresponds to the black 8–8 block in Fig. 6. Neighbours influence a given element via the numerical flux over the mutual interface. As an example, the red 8-13 block has been chosen. All other elements do not contribute non-zero elements to a particular row of the system matrix $\mathscr{H}'$.

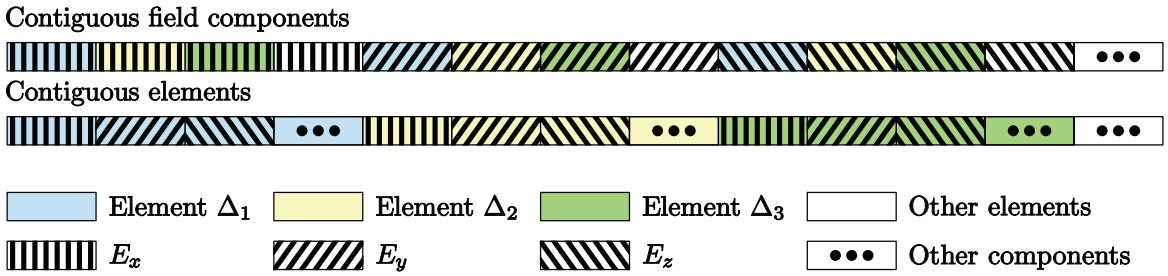


Figure 8 (online color at: www.lpr-journal.org) Storage order of expansion coefficients. Tasked with storing expansion coefficients for a number of field components on a number of elements, one can either store individual *field components* (top display) or all components of individual *elements* (bottom display) contiguously in memory. Usually, the latter is preferable (see Sect. 3.2.4).

tion, it simplifies the use of the Block-Jacobi preconditioner which will be presented in Sect. 3.2.5.

3.2.5. Preconditioning

Usually, large time-harmonic problems cannot be treated by direct solvers because of vast memory requirements. Iterative solvers, on the other hand, have very low memory requirements at the cost of increased run times. Besides the number of unknowns N and the quality of the initial guess, the run time essentially depends on the condition number $\kappa(\mathscr{H}')$ of the system matrix $\mathscr{H}'$. It can be expressed as

$$\kappa = \frac{\max_i |\lambda_i|}{\min_i |\lambda_i|},$$

the ratio between the maximum and the minimum absolute values of the system operator's eigenvalues λ_i. The larger the condition number, the more iterations it takes to reach a desired level of accuracy. The idea behind preconditioning is not to solve Eq. (21), but an alternative problem

$$\mathscr{P}^{-1}\mathscr{H}'\tilde{\mathbf{q}} = \mathscr{P}^{-1}\cdot\tilde{\mathbf{q}}^{\text{source}}$$

with the very same solution $\tilde{\mathbf{q}}$. The $N\times N$ matrix $\mathscr{P}$ is called a preconditioner. If

$$\kappa(\mathscr{P}^{-1}\mathscr{H}') \ll \kappa(\mathscr{H}')$$

and the cost of multiplying $\mathscr{P}^{-1}$ with a vector is sufficiently small, than the reduced number of iterations outweighs the additional computational cost per iteration. As a consequence, iterative solvers need considerably less time to obtain an approximate solution to the system of linear equations.

The minimum condition number is obtained for $\mathscr{P} = \mathscr{H}'$. However, in this case one would require the inverse of $\mathscr{H}'$ – which would be the solution of the actual problem (21) for any source vector. If such an inverse was readily available, one would not have to resort to iterative solvers. Instead, preconditioners should approximate the system operator in a way that allows an easy computation (and probably storage) of its inverse, while the product preconditioned matrix $\mathscr{P}^{-1}\mathscr{H}'$ should be as close to the unit matrix as possible.

Fulfilling and balancing these requirements – especially for non-symmetric problems like the present one – is not easy and often more art than science. The most simple preconditioner, the so-called Jacobi preconditioner, is given by

$$\mathscr{P}_{ii} = \mathscr{H}'_{ii}, \qquad \mathscr{P}_{ij} = 0 \quad \text{for} \quad i \neq j.$$

Unfortunately, the system operator of the DG method is not diagonally dominant and limits the effect of the Jacobi preconditioner. A block Jacobi preconditioner extends this approach by dividing the available indices into mutually distinct sets I_k:

$$\mathscr{P}_{ij} = \begin{cases} \mathscr{H}'_{ij} & \text{for} \quad i \in I_k,\, j \in I_k \\ 0 & \text{for} \quad i \in I_k,\, j \in I_l,\, l \neq k \end{cases}.$$

Thus, $\mathscr{P}$ represents the values of $\mathscr{H}'$ which lie on square blocks on its diagonal. The size of the blocks is given by the number of elements in each index set I_k. Naturally, the block size influences the performance of the preconditioner. For our DG scheme, there are two obvious choices.

For a start, the blocks might comprise the n expansion coefficients of a single field component in a single element. In this case, the entries in each block typically correspond to coupling of a field component with itself as introduced by the numerical flux (compare Fig. 7). Our experience shows that this choice only leads to minor speed-ups for iterative solvers.

The more advanced choice involves index sets which consist of *all* degrees of freedom of a *single* element. These blocks typically are $6n\times 6n$ matrices for three-dimensional simulations. In the presence of auxiliary fields they are even larger. Seen from a physical perspective, this preconditioner corresponds to the solution of Maxwell's equations in an element-local fashion. Hence, it is very similar to the original idea of the DG discretisation. As it turns out, the condition number of $\mathscr{H}'$ can be reduced by several orders of magnitude using this element Jacobi preconditioner – as one is tempted to call it. It should be pointed out, though, that the blocks can be quite large. For example, a $p = 3$ discretisation in three dimensions ($n = 20$) leads to 120×120 blocks, which amounts to 225 kB of memory to store the

complex entries in double precision. This may reduce the practicability of this preconditioner for larger systems.

3.2.6. Advantages and drawbacks

As compared to conventional FEM, the DG frequency-domain method as presented here has a couple of drawbacks. First of all, FEM solvers usually start from a wave equation instead of Maxwell's equations. This reduces the number of degrees of freedom by a factor of two. Furthermore, the field discretisation is usually done via so-called edge elements [24, 25]. These elements are constructed to be curl-conforming, i. e., the divergence of the electromagnetic fields is always zero. In contrast, since DG does not strictly enforce the divergence condition on the entire computational domain, we are burdened with more degrees of freedom than necessary. Furthermore, we are challenged by the non-symmetric nature of the system matrix $\mathscr{H}'$, which leads to issues regarding preconditioning and solving the related system of linear equations. Symmetric DG discretisations of the second-order wave equation, as discussed in [53], for example, seem preferable from an efficiency point of view.

It should be noted, though, that the key strength of the discontinuous Galerkin discretisation is the time-domain, where it allows for a truly explicit scheme. Its benefits are not as obvious in the frequency-domain. Nevertheless, especially the time-harmonic solver is a valueable addition to an existing time-domain code, and it is fairly easily derived from it. We have identified two key modifications to an existing code base:

1. The code must support complex field expansion coefficient in addition to real-valued ones. Appropriate routines to multiply real-valued matrices (derivative matrices, etc.) with complex-valued vectors must be provided as well.
2. Solvers for systems of linear equations must be provided. Fortunately, one can rely on a number of well-tested libraries.

Furthermore, to fully exploit the advantages of direct solvers, one needs an efficient routine for creating the matrix representation of the system operator. A dynamic block storage which automatically creates dense, diagonal, and column-shaped blocks when needed is convenient, especially in the light of possible extensions and auxiliary fields. Finally, a preconditioner is extremely useful when enhancing the performance of iterative solvers. Block-Jacobi techniques are easily implemented and show promising results.

4. Applications to nanophotonics

After all the abstract theory, it is time to provide examples of some real-world applications. The discontinuous Galerkin method has a number of strengths which are important for nanophotonics. Due to the adaptive mesh and higher-order basis functions, it is perfectly suited to tackle multi-scale problems. Tiny features as well as large volumes can be efficiently modeled. Strong local field enhancements in the vicinity of nanoantennas are easily resolved. Round geometries are accurately resolved by means of curved elements. On top of that, DG(TD) is very efficient in terms of CPU time and memory consumption and predestined for parallel computation. For these and other reasons, in particular the past few years have seen a steadily growing number of publications and conference contributions regarding the DG method for Maxwell's equations from numerous groups from all around the world.

Quite naturally, most authors validate their implementation with some simple test particles, mostly squares, cylinders, and spheres [32, 38, 59–61]. In this section, however, we have compiled some references which give a more application-oriented overview of what is possible with DG methods, especially in the time-domain.

Let us start with numerical experiments which involve dielectric materials only. Ji et al. have investigated the coupling between slab waveguides and one or two microring resonators in two dimensions [62]. In a similar study, Niegemann et al. have compared results on disc and ring resonators obtained using two numerical methods, namely DGTD and FDTD [44]. They have found that for comparable accuracy DGTD requires considerably less resources (CPU time and RAM) than FDTD. Another interesting example is provided by Tang et al., who have investiged the scattering of light by two-dimensional cylindrical and hexagonal ice crystals at optical and infrared frequencies [33].

Chun et al. have applied the DGTD method to the propagation of waves in layered, periodic structures with anisotropic material properties [63]. In particular, they have been able to analyse the finite size effect on the formation of the frozen mode phenomenon. Anisotropic material distributions have also been considered by König et al. to assess the quality of a cylindrical optical cloak [64].

Metallic nanostructures and metamaterials have found the broadest coverage within the DGTD literature. One of the first studies on metallic nanostructures has been published by Ji et al., who have investigated the coupling of silver nanowires [65]. Shi et al. have modified the DGTD method in order to simulate effective left-handed media with Lorentz dispersion and their focussing properties [35]. The anomalous transmittance through sub-wavelength silver apertures has been examined by Niegemann et al. [32]. Hille et al. have shown that curvilinear elements significantly improve the local field distribution in V-shaped grooves in silver strips [34]. Three-dimensional simulations of V-shaped particles have been performed by Stannigel et al. to evaluate the possibility of coherent control of spatio-temporal field distributions via chirped pulses [60]. Finally, the paper by Feth et al. provides a comparison between experimental and numerical results on the coupling of split-ring resonator dimers [66].

In addition, the DG method has been applied to other, more macroscopic electromagnetic systems as well. For example, Lu et al. have employed the DGTD method to simulate ground-penetrating radar measurements of objects and cavities embedded in soil [67]. Another radar application is the scattering of electromagnetic waves by aircraft, as for ex-

ample studied by Dolean et al. [68] or Montseny et al. [69]. Last but not least, Dolean et al. have investigated the interaction of mobile phone radiation with the human head [68].

In the upcoming sections we will provide more details on the performance of the DG method for two example systems.

4.1. Coupled resonator-waveguide systems

In this section we present some results on a two-dimensional system in transverse-electric polarisation. It consists of a ring resonator with radius $r = 4\,\mu\text{m}$ which is coupled to two waveguides of width $w = 400\,\text{nm}$ (see Fig. 9 for a sketch). The distance d between the waveguides and the resonator is just 50 nm. Both the waveguides and the ring resonator are made of a material with $\varepsilon = 9$, the rest of the computational domain is vacuum. A layer of PMLs surrounds the system to absorb outgoing radiation (see Sect. A.4). Finally, a waveguide mode is launched in the upper left waveguide port via the total-field/scattered-field technique (Sect. A.1.1).

For the numerical simulation a few challenges exist:

- The gaps between the waveguides and the ring resonator are rather small compared to the other dimensions of the systems. Hence, the discretisation comprises multiple scales.
- Perfectly round ring resonators feature very high quality factors. Numerical surface roughness – as introduced by straight-sided elements – leads to additional scattering and, thus, should be avoided.

As a first step, we model the system using straight-sided elements (Fig. 9). Since we are primarily interested in the transmittance into the upper right waveguide port (details on transmittance calculations can be found in Sect. A.6.3), we employ a time-domain simulation with a broad-band source. For the time-stepping we use a 14-stage 4th order LSRK scheme (see Sect. 3.1.3). The resulting spectra for various polynomial orders p (see Sect. 2.5) are shown in Fig. 10. Apparently, polynomial orders below $p = 5$ introduce too much dissipation for the given mesh, since the electromagnetic fields are not properly resolved. Additional calculations were performed for orders up to $p = 8$. While the resonance positions do not change noticeably, a 1% change occurs in the absolute transmittance values for $p = 6$ and $p = 8$.

In the second step, we then identify a resonance of interest at $\lambda \approx 1.5426\,\mu\text{m}$, for which we would like to know the magnetic field distribution. We could either start another time-domain simulation and obtain the field distribution via the on-the-fly Fourier transform (Sect. A.6.1), or we can just use the time-harmonic solver instead. Since our mesh merely comprises 1404 elements, the latter is a computationally efficient choice. Thus, we prepare another simulation with the same system, but with a time-harmonic solver instead of the time-domain algorithm. The resulting system of equations (20) is solved using the direct solver `UMFPACK` and yields the field distribution at resonance (Fig. 9).

The time-domain simulation with $p = 5$ required roughly 400,000 time steps to simulate 10,000 optical cycles at $\lambda = 1\,\mu\text{m}$. Using a 14-stage LSRK method amounts to 5.6×10^6 evaluations of the right-hand side of Eq. (10). On a single Intel Core 2 Quad processor with 2.5 GHz the total CPU time for this calculation is approximately 11 h. For the same order, the memory requirements are below 0.1 GB. During one time-domain simulation we evaluate the spectrum for 2001 frequencies.

The frequency-domain computation of the field distribution required the formation and solution of a system of linear equations with 93282 unknowns. The total time to obtain a solution for a single frequency is approximately 16 s. On the other hand, solving the system required roughly 1.6 GB peak memory.

Finally, we investigate the quality of our geometrical model. So far, the ring resonator is meshed using straight-sided (linear) elements. Consequently, higher polynomial orders will lead to more converged results on a polygonal

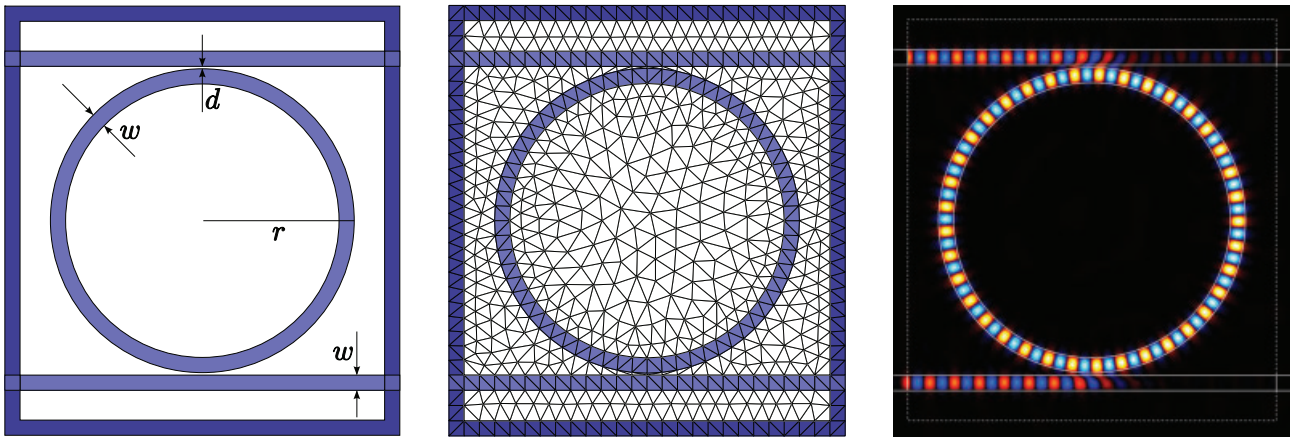


Figure 9 (online color at: www.lpr-journal.org) Two slab waveguides coupled to a cylindrical ring resonator. Left panel: Sketch of the physical system. A perfectly matched layer surrounds the computational domain to absorb outgoing radiation. Central panel: The mesh which has been used for the DG computations. Right panel: False colour plot of the H_z component of a mode at $\lambda \approx 1.5426\,\mu\text{m}$ computed with the time harmonic DG solver for $p = 5$.

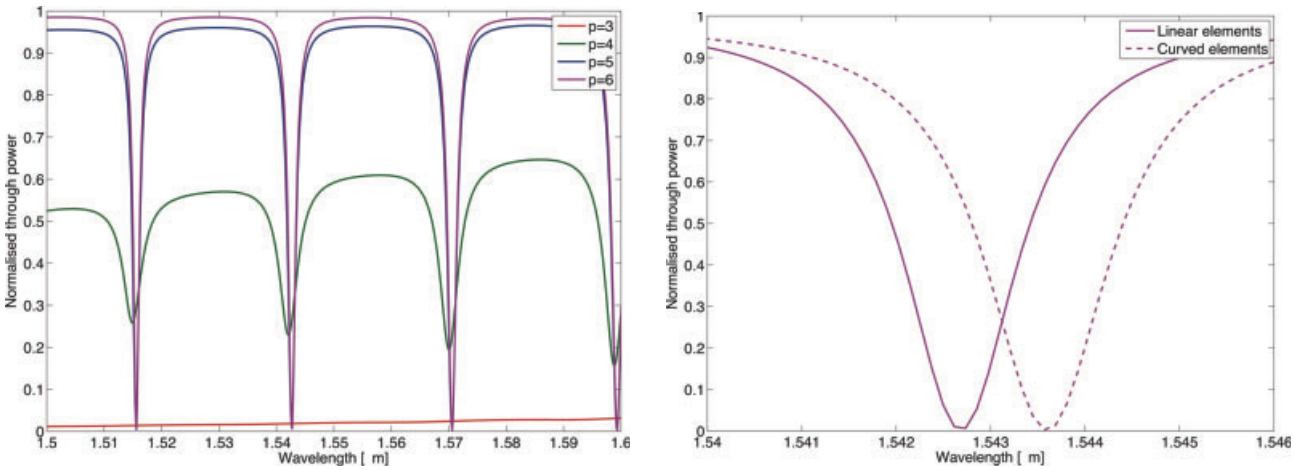


Figure 10 (online color at: www.lpr-journal.org) Transmittance into the upper right waveguide port of the system sketched in Fig. 9. Left panel: Transmittance for various polynomial orders p using linear meshes. Right panel: Transmittance for the polynomial order $p = 6$, where the resonator is modelled with linear and curvilinear elements, respectively. The corner points in both meshes are identical.

approximation of the resonator. A smoother model of the resonator either requires smaller elements or the utilisation of curvilinear elements (Sect. A.5). Figure 10 shows the influence of the different geometric discretisation. As compared to the exact model with curvilinear elements, the results obtained with linear elements shows a small blueshift.

4.2. Split-ring resonators

In our second example we investigate split-ring resonators (SRRs), which are often used as building blocks for metamaterials. As a first step to understand how such a metamaterial's properties arise from the properties of individual building blocks, it is interesting to study the coupling of just two SRRs which form an SRR dimer [66].

Since we investigate isolated structures instead of a periodic arrangement, we use a modified three-dimensional scattering setup as outlined in Sect. A.6.3. We place two equally shaped gold SRRs of height $h = 200\,\text{nm}$, gap height $g_h = 100\,\text{nm}$, width $w = 200\,\text{nm}$, gap width $g_w = 80\,\text{nm}$, and thickness $t = 30\,\text{nm}$ in the total-field region (Fig. 11). The dispersion of gold is modelled with a standard Drude model [70]. This model is a good approximation of the material parameters in the near-infrared region. Furthermore, we include a glass substrate ($\varepsilon = 2.25$) into the system. The remainder of the computational domain is filled with vacuum ($\varepsilon = 1$). Perfectly matched layers (Sect. A.4) surround the computational domain in both the vacuum and the substrate region. Illuminating the system with a plane wave source (normal incidence) via the TF/SF technique (see Sect. A.1.1) and integrating the Poynting vector on the TF/SF interface yields the scattering, absorption, and extinction cross-sections. The electric field of the incident wave is parallel to the axis from the first SRR to the second SRR, i. e., it corresponds to a horizontal polarisation in Fig. 11. The symmetry of the system can be exploited by enforcing PEC boundary conditions on the mirror plane. As a result, the computational effort is reduced by a factor of two.

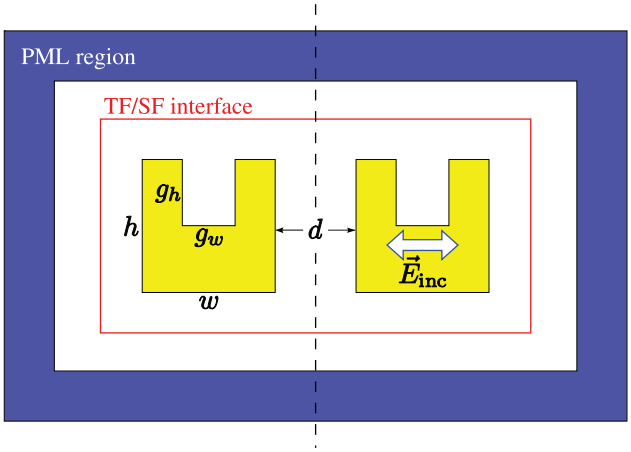


Figure 11 (online color at: www.lpr-journal.org) A two-dimensional sketch of a computational setup used to study the coupling between SRR dimers. To reduce the computational effort, a mirror plane with PEC boundary conditions is included. The incident plane wave is horizontally polarised.

To study the coupling between the two SRRs we vary the distance d between both scatterers. For each distance we create a tetrahedral mesh and calculate the cross sections using the DGTD method. Typical spectra for $d = 120\,\text{nm}$ are depicted in Fig. 12. Using a Lorentz fit, we locate the resonance wavelength from the extinction cross sections (Fig. 13). Apparently, a strong red shift is observed for small separations which can be attributed to electric dipole-dipole coupling [66].

For all computations we use fourth order ($p = 4$) polynomials. Depending on the distance between the SRRs, the corresponding meshes consist of $24,000$ to $30,000$ tetrahedrons. Each simulation requires less than 750 MB of memory. A simulation time which corresponds to 30 optical cycles for $\lambda = 1\,\mu\text{m}$ translates into about $15,000$ time steps with a 14-stage LSRK scheme (see Sect. 3.1.3). Hence, we need $210,000$ evaluations of the right-hand side of Eq. (10).

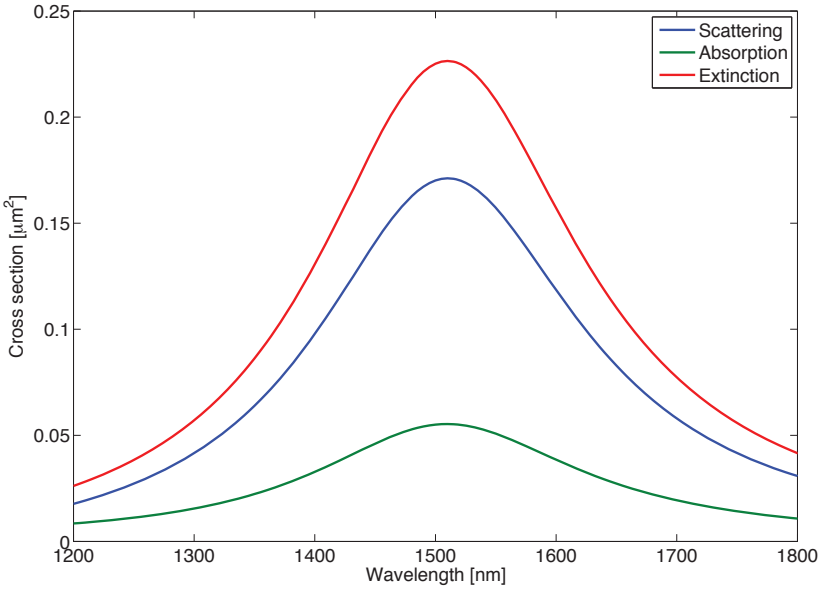


Figure 12 (online color at: www.lpr-journal.org) Scattering, absorption, and extinction cross sections for an SRR dimer with $d = 120\,\mathrm{nm}$. Please note that the cross sections have been normalised to the number of SRRs, i. e., by a factor of 0.5.

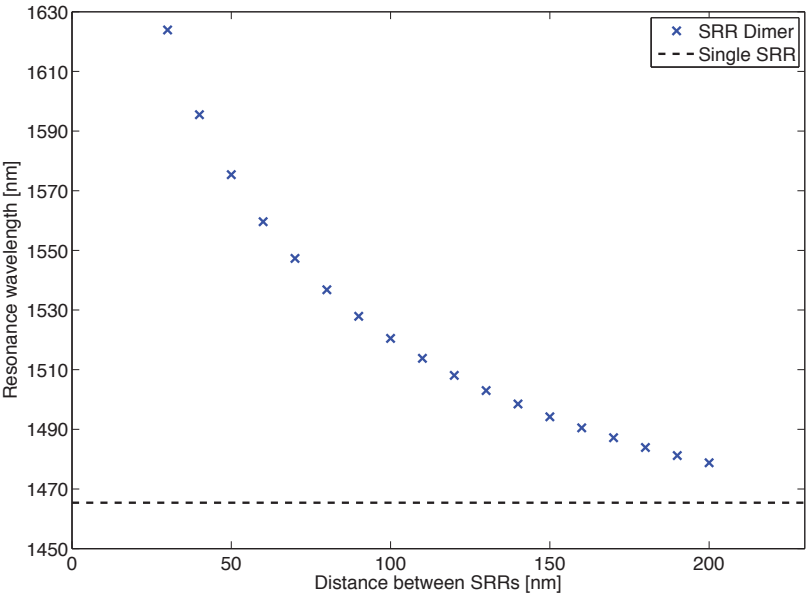


Figure 13 (online color at: www.lpr-journal.org) Resonance wavelengths as obtained from the extinction cross sections for SRRs separated by various distances d. The horizontal line indicates the resonance wavelength of a single SRR, which corresponds to a dimer with $d = \infty$.

In total, the required simulation times on a single core of an Intel Xeon X5570 CPU (2.93 GHz) lie between 28 and 34 h. Coarse computations with $p = 2$ yield surprisingly accurate results in less than 3 h.

Once the resonance positions are determined, we can investigate the field distribution on resonance. As opposed to the two-dimensional system in the previous section, we use another time-domain simulation in conjunction with the on-the-fly Fourier transform (Sect. A.6.1) to avoid the high memory consumption of the time-harmonic solvers. Figure 14 shows the absolute value of the time-harmonic electric field in a plane through the centre of both SRRs. One can clearly observe strongly localised electric fields near the arms of the SRRs. The mode profiles for different spacings d reveal that for small gaps something like a capacitor is formed between the right arm of the left SRR and the left arm of the right SRR. This capacitor has sig-

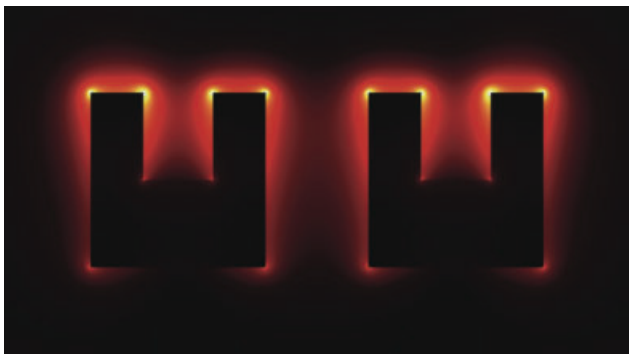


Figure 14 (online color at: www.lpr-journal.org) Absolute value of the time-harmonic electric field in a plane through the centre of the SRR dimer with $d = 120\,\mathrm{nm}$. Note the strongly localised fields near the sharp corners.

nificant influence on the resonance of each individual SRR and is a possible explanation for the resonance shift of the SRR dimer.

5. Outlook

Even though reading this review might have been exhausting to the reader, our discussion of the discontinuous Galerkin technique in nanophotonics is by no means exhaustive. In this concluding section we describe a couple of possible extensions, open problems, and potential areas of research.

5.1. Improving the spatial discretisation

The spatial discretisation we have presented in Sect. 2 is not ideal in a number of ways. Though tetrahedrons allow us to model arbitrary geometries, we usually do not require this freedom in the entire computational domain.

For example, a computational domain could be enclosed by an axes-aligned perfectly matched layer. While the actual physical domain with arbitrary scatterers will benefit from conventional tetrahedral meshes, such boundary regions could be easily tessellated into hexahedral elements [61]. Here, nodes would be positioned in terms of a tensor product of one-dimensional node distributions. As derivatives would only incorporate degrees of freedom along a line, performance gains are expected immediately. Furthermore, the surface-to-volume ratio of a cuboid is lower than that of the tetrahedrons it can be divided into. Hence, we have less degrees of freedom associated with element boundaries. As we have to store these values twice, hexahedrons lead to a more efficient scheme. Similarly, layered systems will benefit from prism-shaped elements.

Unfortunately, such hybrid meshes pose significant technological challenges to mesh generation and handling. Appropriate data structures are needed to allow a mesh to hold elements of different types and to relate them with each other. More fundamentally, sophisticated algorithms which create optimal meshes with mixed element types for arbitrary geometries are required.

Besides the element shape, the set of basis functions is another aspect of the DG discretisation to refine. While the nodal scheme is well tested and convenient due to the close relation between expansion coefficients and field values, it supports fields of non-vanishing divergence (see Sect. 3.2.2). Expressed in other terms, this means that the nodal basis contains superfluous degrees of freedom. To eliminate this redundant information one can construct a *modal*, vectorial basis which locally satisfies the divergence condition, i. e., in an element-wise fashion [53, 54, 71]. However, since the locally divergence-free basis functions are not invariant under affine transformations, such an implementation dramatically increases the memory consumption. In addition, the calculation of the numerical flux also requires more computational effort in a modal representation when compared to the nodal version. Thus, it is not clear yet whether a locally divergence-free basis actually leads to performance improvements.

Finally, one usually assumes that the fields in each element are discretised with the same local order of accuracy p. In addition to the local mesh refinement, DG also allows the local refinement (or coarsening) of p. Consider a structure with just a few small features which need to be resolved by a couple of small elements. Compared to the other elements, the feature region is discretised by considerably more degrees of freedom, and probably more than necessary. Hence, one could locally reduce the order of the basis functions. Following the discussion of Sect. 3.1.2 this will increase the time step, as the distance between adjacent nodes is larger for lower orders.

5.2. Time stepping

In contrast to FDTD, the spatial discretisation of DGTD is strictly separated from the time evolution of the fields. Though Runge-Kutta schemes show satisfactory performance, it might be worthwhile to investigate alternative time stepping techniques. The main motivation lies in the observation that in nanophotonics the time step is often limited by just a few exceptionally small or awkwardly shaped elements. If a time stepper could treat these elements separately, one could maintain a larger time step for the bulk of the elements. Hence, depending on the mesh dramatic performance boosts can be expected.

One possible strategy is to use explicit solvers with different time steps for differently sized elements [36, 69, 72]. For example, we might divide the elements into the categories small, medium, and large. Fields in small elements are evolved in steps of Δt, fields in medium elements in steps of $2\Delta t$, and the largest elements in steps of $4\Delta t$. Problems arise because the numerical flux links small and medium elements and we have to make sure that fields in both elements are known at the same time. Appropriate interpolation schemes or interwoven time stepping schemes are required. While such schemes are available and employed, e. g., in finite volume calculations [73], they are usually restricted to lower orders. Higher-order schemes [74] (or schemes of mixed order) appear more suitable to accompany the higher-order spatial discretisation of DGTD. Recently, higher-order Taylor approximations [75] and third-order Adams-Bashforth schemes [76] were employed to design local time-stepping schemes for large electromagnetic problems. Both methods are suited for an efficient implementation on GPUs. For selected large problems, the authors have demonstrated significant performance gains over standard LSRK techniques with a global time step.

Alternatively, one can employ hybrid implicit-explicit time stepping schemes [68]. Implicit time steppers require the solution of a system of linear equations, which is in contrast to the explicit nature of the usual DGTD update process. However, implicit schemes are unconditionally stable and allow large time steps with comparatively small numerical errors. By applying an implicit scheme to the smallest couple of elements only, one can increase the overall time step while keeping the computational effort per time step within

reasonable bounds. The fields in the remaining elements are evolved in the usual explicit manner.

5.3. Material models and coupled system dynamics

The material models presented in this article should be sufficient for most linear experiments in nanophotonics. However, nonlinear material responses [77] have attracted interest of a large number of research groups and become more important each day.

Classical nonlinearities of $\chi^{(2)}$ and $\chi^{(3)}$ type are responsible for the optical Pockels and Kerr effects, wave mixing phenomena, etc., and are often found in substrate materials such as GaAs. The inclusion of nonlinear effects in the DGTD framework is not easy. The main problem is to find an efficient and accurate expression for the numerical flux.

In addition, nonlinear system dynamics arise when dealing with more sophisticated material models, too. For example, the simple Drude theory of metals could be extended to a hydrodynamic model, which describes the free metal electrons in terms of charge and current densities [78]. One obtains additional equations which have to be solved concurrently with Maxwell's equations. The coupling between both types of equations via currents and driving fields introduces nonlinearities and leads to the generation of higher harmonics. Other examples of coupled systems include the interaction of classical light fields with quantum mechanical systems. For example, the radiation dynamics of light emitted from quantum wells or atoms can be significantly influenced by the surrounding media [77].

While the linear regime seems to be well understood, it appears that, so far, it has not been reported yet how to incorporate non-linear material models in the DG framework for Maxwell's equations. Once this issue is solved, it will open a whole new research area to the powerful machinery that is the DG method.

We believe that the discontinuous Galerkin method is on a good way to become one of the most efficient and versatile simulation tools for all kinds of nanophotonic systems. We are eager to see which further developments the future holds.

A. Appendix: Essential extensions for practical use

The algorithm presented in the main part of this article is able to deal with straight-sided objects made out of non-dispersive materials. These objects are enclosed in a computational domain whose boundaries either perfectly reflect or absorb a good portion of incoming light.

At this point, we are unable to simulate metallic structures in the near-infrared and visible spectrum, where a perfect electric conductor no longer presents a valid approximation to the dispersive material properties. Plasmonics, and thus a complete area of nanophotonic research, would be out of reach.

We might also be interested in the scattering properties of isolated particles. To access these in a well-defined way, we need control over both incident and outgoing radiation. A method to add energy to the system and, in particular, to inject arbitrary pulses is required. In addition, one has to improve on the absorbing boundary conditions to properly terminate the computational domain and avoid non-negligible reflections from the outer boundaries.

Furthermore, many realistic systems present us with significant modelling issues. Photonic crystals often use circular and spherical shapes as building blocks, e. g., circular holes in a silicon membrane or SiO_2 beads as building blocks of opals. Thus, a proper simulation requires an accurate description of round objects which quickly leads to a large number of rather small elements when using straight-sided elements.

Last but not least, we have not yet considered how to extract physical quantities of interest from mere field distributions. How can we efficiently obtain spectra from time-domain simulations?

The next few sections will address all these issues and present all the necessary techniques to obtain a versatile simulation tool which is applicable to a wide range of problems.

A.1. Sources

When modelling a physical system one usually thinks in terms of materials and geometrical structures. Equally important, though, is how the system is illuminated as the properties of the incident light decide whether we will observe interesting physics or not. Consequently, control over strength, polarisation, propagation direction, and frequency (spectrum) of an incident wave is desired for a number of possible illumination patterns, which include plane waves, waveguide modes, focussed laser beams, point sources, shaped beams [79] and many more.

In principle, almost arbitrary radiation patterns can be included via properly set initial conditions for $\vec{E}$ and $\vec{H}$. Unfortunately, even short pulses show significant field strengths in rather large volumes. Hence, additional computational effort is needed just to model the illumination. Furthermore, some patterns such as a dipole source in close vicinity to a scatterer cannot easily be represented using initial conditions. As soon as the initial fields are non-zero within the scatterer, wrong results will be obtained unless the scatterer is incorporated during the calculation of the initial field pattern. If this were the case, however, there would be no need for a simulation in the first place. Finally, frequency-domain simulations do not support initial conditions. Thus, we need another way to deal with incident fields. The method of choice is the total-field/scattered-field technique, which is efficiently implemented via the numerical flux.

A.1.1. The total-field/scattered-field technique

The total-field/scattered-field (TF/SF) technique is very popular for incorporating plane wave sources in FDTD simula-

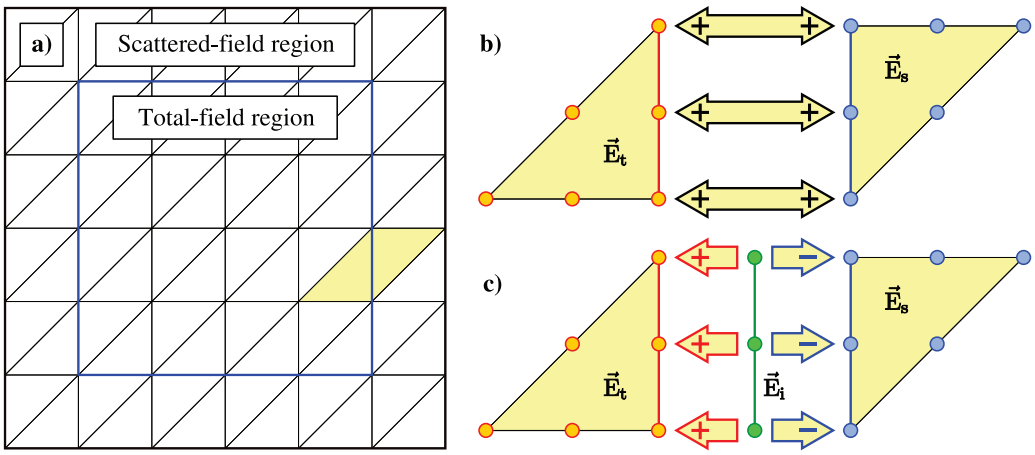


Figure 15 (online color at: www.lpr-journal.org) Illustration of the total-field/scattered-field technique as discussed in Sect. A.1. a) The computational domain is split into a total-field and a scattered-field region. The blue line indicates their mutual interface. b) Zoom on the elements highlighted in a). One element stores total field components $\vec{E}_t$ while the other stores scattered field components $\vec{E}_s$. According to Sect. 2.3, the field differences across element interfaces are given by the fields in the neighbouring element minus the field in the local one, as indicated by the arrows. c) After the field differences for all elements have been calculated according to b), the field differences on the total-field/scattered-field interface are modified. Using the incident field $\vec{E}_i$ we can convert $\vec{E}_s$ to $\vec{E}_t$ and vice versa.

tions [20, 80]. It relies on the linearity of Maxwell's equations, which allows us to split the electromagnetic fields into two contributions:

$$\begin{aligned} \vec{E}_t(\vec{r},t) &= \vec{E}_i(\vec{r},t) + \vec{E}_s(\vec{r},t)\,, \\ \vec{H}_t(\vec{r},t) &= \vec{H}_i(\vec{r},t) + \vec{H}_s(\vec{r},t)\,. \end{aligned} \tag{22}$$

The subscripts "t" indicate the *total*, i. e., physical fields, whereas "i" and "s" represent *incident* and *scattered* fields, respectively. The incident field is the known analytical expression for the desired illumination pattern, e. g., the field distribution of a plane wave. With its help, one can calculate scattered fields from total fields and vice versa.

Acknowledging this, we exploit that elements only couple to their immediate neighbours and divide the computational domain into two regions. In the TF region we are interested in the total fields. Similarly, we solve for the scattered fields in the SF region. Because of the linearity of Maxwell's equations and ansatz (22), Maxwell's equations remain valid for both the TF and the SF region. Hence, we solve the equations

$$\begin{aligned} \mathcal{Q}(\vec{r})\cdot\partial_t \mathbf{q}_t(\vec{r},t) + \vec{\nabla}\cdot\vec{F}(\mathbf{q}_t) &= 0 \\ \mathcal{Q}(\vec{r})\cdot\partial_t \mathbf{q}_s(\vec{r},t) + \vec{\nabla}\cdot\vec{F}(\mathbf{q}_s) &= 0 \end{aligned}$$

in the respective domains. In the event of the DG discretisation, we will eventually encounter the field differences

$$\begin{aligned} \Delta\vec{E}_t(\vec{r},t) &= \vec{E}_t^{+}(\vec{r},t) - \vec{E}_t^{-}(\vec{r},t)\,, \\ \Delta\vec{E}_s(\vec{r},t) &= \vec{E}_s^{+}(\vec{r},t) - \vec{E}_s^{-}(\vec{r},t)\,, \end{aligned}$$

which we need for the computation of the numerical flux. As previously introduced in Sect. 2.3, "+" represents the value of the neighbouring element while "−" represents the local one. As long as we are in the interior of either the TF or the SF region, the evaluation of the field differences is straightforward. However, right at the interface between both regions we cannot immediately use the field values of the neighbouring element. For example, the neighbour of an element in the TF region stores $\vec{E}_{scat}$, and not $\vec{E}_{tot}$ as would be needed. Fortunately, we can use relation (22) to obtain

$$\begin{aligned} \Delta\vec{E}_t(\vec{r},t) &= \vec{E}_s^{+}(\vec{r},t) - \vec{E}_t^{-}(\vec{r},t) + \vec{E}_i(\vec{r},t)\,, \\ \Delta\vec{E}_s(\vec{r},t) &= \vec{E}_t^{+}(\vec{r},t) - \vec{E}_s^{-}(\vec{r},t) - \vec{E}_i(\vec{r},t)\,. \end{aligned} \tag{23}$$

Similar statements hold for the differences of the magnetic field. To include TF/SF sources in an existing code, we do not have to distinguish between total fields and scattered fields or even implement independent discretisations of Maxwell's equations in both regions. Instead, it is sufficient to update the field differences on the TF/SF boundary by either adding or subtracting the incident field according to (23). Hence, the TF/SF technique is an ideal extension to the DG method. The TF/SF procedure is illustrated in Fig. 15.

As long as analytical (or semi-analytical) expressions for both the electric and the magnetic components of the incident field are available, we can readily use the source. Field distributions for a number of relevant sources can be found in many text books, for example, see [81, 82].

A.1.2. Point sources

Typically, the mathematical trick of the TF/SF technique presented in the last section is used for beam sources such as plane waves. Alternatively, one might utilise the actual source terms in Maxwell's equations. This method has considerable appeal for the modelling of point sources, which are easily represented by singular current distributions

$$\vec{j}_s(\vec{r},t) = \vec{j}_0 \cdot f(t) \cdot \delta^3(\vec{r} - \vec{r}_0)\,,$$

where $\delta^3(\cdot)$ denotes the three-dimensional Dirac distribution. In addition, we have defined the polarisation vector $\vec{j}_0$, the source's time dependence $f(t)$, and its position $\vec{r}_0$. In principle, one can include such a source by adding $\vec{j}_0 \cdot f(t)$ to the expansion coefficients of the electric field at a single node.

As it turns out, Lagrange polynomials as used for the field expansion (see Sect. 2.5) are notoriously bad at resolving the – theoretically infinite – field values in the vicinity of $\vec{r}_0$. If one is really interested in the field distribution near a point source, one has to locally refine the mesh. Following the discussions of Sect. 3.1.2, this will drastically reduce the time step and is, thus, unfavourable.

Therefore, we have proposed to model current sources via the TF/SF technique as well [32]. In contrast to the usual layout of the computational domain, the total-field region encloses the scattered field region, which in turn contains the point source. For this setup, the field distribution on the TF/SF interface is semi-analytically computed using free space Green's functions and quadrature rules to numerically perform the required integrations. Following this approach, point sources can be simulated both accurately and efficiently.

A.1.3. Typical time dependencies

The total-field/scattered-field technique, which we apply to both point and beam sources, requires the incident field distribution $\vec{E}_i(\vec{r}, t)$ according to Eq. (23). The most common source, a plane wave, can be written as

$$\vec{E}_i(\vec{r}, t) = \vec{E}_0 \cdot \mathrm{Re}\left\{\exp\left(-i\omega_0 t' + \phi_0\right)\right\}, \qquad t' = t - \frac{\vec{k}\cdot\vec{r}}{|\vec{k}|\cdot c}. \tag{24}$$

Here, $\vec{E}_0$ represents the polarisation and amplitude of the incident electric field, $\vec{k}$ is the wave vector and characterises the direction of propagation, ω is the (angular) frequency of the incident wave, and ϕ_0 denotes the initial phase. The spatial shape of the plane wave is absorbed into the local time parameter t', wherein the speed of light, c, in dimensionless units is identical to the refractive index of the medium. Eq. (24) is a naïve approach to model a plane wave. Though clearly being a valid description, it is disadvantageous for numerical simulations for a number of reasons:

- The plane wave exists for all times, it has neither a well-defined beginning nor an end. As the sum of incident and scattered field must match the total field, the initial condition in the total-field region must match the incident wave. If it does not, artificial, non-physical scattering will be introduced by the TF/SF interface. However, modifying the initial condition is not trivial, especially in the presence of scatterers. If we knew how an initial field in the presence of a scatterer could be calculated, we would not need the simulation anyway.
- A Fourier transform is often employed for data acquisition and analysis (see Sect. A.6.1 for a detailed discussion). Accurate results require the fields of interest to decay until the end of the simulation. The plane wave as given by Eq. (24) does not decay at all.
- Obviously, it incorporates waves of only a single frequency ω_0. This eventually leads to a steady state. For such computations frequency domain simulations seem more appropriate.

To improve on our simple approach we replace the harmonic time dependence by

$$\vec{E}_i(\vec{r}, t) = \vec{E}_0 \cdot \mathrm{Re}\left\{f\left(t'(\vec{r}, t)\right)\right\}. \tag{25}$$

Again, t' ensures the spatial shape of a propagating plane wave is maintained. The newly introduced complex-valued scalar function $f(t)$ represents the time-dependence of the amplitude at the origin of the coordinate system.

The freedom of choosing $f(t)$ is essential for the success of time-domain simulations. For time-harmonic calculations, we are obviously stuck with $f(t) = \exp(-i\omega t + \phi_0)$. Let us consider the important choice

$$f(t) = i \cdot \exp\left(-i\omega_0 (t - t_0) - \frac{(t - t_0)^2}{2\sigma^2}\right). \tag{26}$$

It represents a harmonic oscillation with frequency ω_0 with a Gaussian envelope centred around t_0 with width σ. Due to this envelope, all problems which we have discussed earlier are accounted for:

- For $t = 0$, the electromagnetic fields are sufficiently close to zero for all positions $\vec{r}$ within the total-field region, provided that t_0 is sufficiently large. Hence, the artificial scattering introduced by the mismatch of the initial (zero) condition and the incident field is minimised.
- Similarly, for large values of t the Gaussian envelope will exponentially suppress the amplitude of the incident field. Only a finite amount of energy is injected into the system. Losses in the system, either via dispersion (Sect. A.2) or absorbing boundary conditions (Sects. 2.4 and A.4), eventually lead to a decay of the electromagnetic fields. Thus, errors due to residual fields in subsequent Fourier transformations are minimised.
- Last but by far not least, the Fourier transform of the *real part* of the time-dependence (26) is proportional to

$$\exp\left(-\frac{\sigma^2}{2}(\omega - \omega_0)^2\right) - \exp\left(-\frac{\sigma^2}{2}(\omega + \omega_0)^2\right).$$

Please note that the minus sign between both exponential functions is a consequence of the coefficient "i" in Eq. (26). Evidently, the energy of the incident pulse is distributed across a whole frequency band instead of just a single frequency. There is no contribution for $\omega = 0$, thus we avoid the creation of static fields. The larger σ, i. e., the longer the duration of the pulse, the narrower the band of significantly contributing frequencies will be. Conversely, low values of σ, which correspond to ultra-fast pulses in the time-domain, lead to very broad frequency bands.

Here, the last point is particularly important for time-domain methods. A single time-domain simulation with such a broad band excitation is sufficient to determine the system's response to a vast number of plane waves of different frequencies. Thus, using appropriate signal processing techniques, one time-domain simulation is equivalent to many frequency-domain simulations. Depending on the desired spectral resolution and the system, either calculation can be faster.

For spatial profiles other than a plane wave, dispersion is often a problem. For example, the shape of a focussed laser beam heavily depends on its frequency. Similar statements hold for waveguide modes and other profiles. In this case, one calculates the spatial field pattern for a central frequency ω_0 and uses a slowly ramped (large σ) Gaussian pulse (26) for the time-dependence. This leads to a very sharp, well-defined peak in the frequency spectrum while at the same time avoiding the problems which would be otherwise introduced by a standard harmonic time dependence.

Other time-dependencies might be interesting to match experiments involving ultra-fast phenomena or to achieve coherent control [60, 79, 83]. In particular, the application of time-reserved pulses to achieve spatio-temporal localisation of radiation is most appealing [84]. The higher-order spatio-temporal discretisation of the DGTD method appears to be ideally suited for such problems.

A.2. Dispersive media

Dispersive materials are often encountered when dealing with nanophotonics. The refractive indices of many dielectrics show only small variations with frequency. It is often a reasonable assumption to use a constant value for the permittivity. In contrast, metals such as silver and gold exhibit dramatic dispersion in the visible and near-infrared regime, where the real part of the permittivity changes its sign and absorption becomes a dominant effect.

Dispersive materials enter Maxwell's equations via the constitutive relation

$$\vec{\mathrm{D}}(\omega) = \varepsilon(\omega) \cdot \vec{\mathrm{E}}(\omega) \,. \tag{27}$$

This equation features the electric displacement $\vec{\mathrm{D}}$, which depends on both the permittivity ε and the electric field $\vec{\mathrm{E}}$. Note that all quantities are frequency-dependent! Apparently, this formulation is ideal for Eq. (20), the frequency-domain version of Maxwell's equations. There, it is sufficient to replace the constant ε (hidden in the system operator $\mathscr{H}$) by the frequency-dependent $\varepsilon(\omega)$. Apart from the system operator now being frequency-dependent and complex-valued, this does not change the algorithm at all.

On the other hand, Eq. (27) poses severe problems for the time-domain formulation. After Fourier transformation, the time derivative of (27) for frequency-independent $\varepsilon(\omega) \equiv \varepsilon$ is given by

$$\partial_t \vec{\mathrm{D}}(t) = \varepsilon \cdot \partial_t \vec{\mathrm{E}}(t) \,. \tag{28}$$

With this, we recover Eq. (1), Maxwell's curl equations for dielectric materials. In the general case, however, the constitutive relation will transform like

$$\vec{\mathrm{D}}(t) = \int_{-\infty}^{t} \varepsilon(t-t') \cdot \vec{\mathrm{E}}(t') \,\mathrm{d}t' \,.$$

Several methods to deal with dispersion have been developed for FDTD [20]. Here, we will present a technique based on auxiliary differential equations, which is best suited for inclusion into a DGTD framework [32, 65]. It requires only minimal changes to the frequency-independent case and is useful for a couple of other extensions as well.

A.2.1. Auxiliary differential equations

Our time-domain formulation relies on analytical models for the dispersion. Our goal will be to find an expression for

$$-\mathrm{i}\omega\vec{\mathrm{D}}(\omega) \quad \leftrightarrow \quad \partial_t \vec{\mathrm{D}}(t) \,,$$

which is easily transfered from the frequency- to the time-domain. As a first step, we separate the permittivity into

$$\varepsilon(\omega) = \varepsilon_\infty + \chi(\omega) \,,$$

where ε_∞ is a constant background permittivity and $\chi(\omega)$ is the frequency-dependent susceptibility. Inserting this relation into (27) yields

$$-\mathrm{i}\omega\vec{\mathrm{D}}(\omega) = -\mathrm{i}\omega\varepsilon_\infty \cdot \vec{\mathrm{E}}(\omega) + \vec{\mathrm{j}}(\omega) \,, \tag{29a}$$

$$\vec{\mathrm{j}}(\omega) = -\mathrm{i}\omega\chi(\omega) \cdot \vec{\mathrm{E}}(\omega) \,. \tag{29b}$$

Here, we have defined an auxiliary field $\vec{\mathrm{j}}(\omega)$ which represents a polarisation current. Transforming (29a) into the time-domain yields

$$\partial_t \vec{\mathrm{D}}(t) = \varepsilon_\infty \partial_t \vec{\mathrm{E}}(t) + \vec{\mathrm{j}}(t) \,.$$

Except for the newly introduced field $\vec{\mathrm{j}}(t)$, the last expression is just the time-derivative of Eq. (28). In particular, no convolution integral enters Maxwell's curl equations. The time-evolution of the auxiliary current is governed by the Fourier transform of (29b). For properly chosen susceptibility models, i. e., rational functions with respect to $\mathrm{i}\omega$, this will lead to auxiliary differential equations (ADEs) for $\vec{\mathrm{j}}(t)$ (and probably additional auxiliary fields), which have to be simulated in parallel to Maxwell's curl equations.

A.2.2. Drude model

In the course of the derivation of the Drude model, one assumes that the metal's conduction electrons behave as charged free particles which are subjected to an external electric field. In other words, they form an ideal classical gas. All interactions with other electrons and core ions are

absorbed in a phenomenological collision frequency γ_D. For the susceptibility we find

$$\chi_D(\omega) = -\frac{\omega_D^2}{\omega\cdot(\omega+\mathrm{i}\gamma_D)} \tag{30}$$

with the sign convention (16). In addition to the collision frequency, we have introduced the plasma frequency ω_D. In practice, one can consider γ_D, ω_D, and ε_∞ as free parameters. Using a fitting procedure, one can reproduce both real and imaginary parts of the permittivity for metals in the near-infrared regime with surprising accuracy. Inserting (30) into (29b) yields

$$-\mathrm{i}\omega\vec{\mathrm{j}}_D(\omega) = \omega_D^2\vec{\mathrm{E}}(\omega) - \gamma_D\vec{\mathrm{j}}_D$$

after a few algebraic manipulations. The necessary time-domain equations now read

$$\begin{aligned}\varepsilon_\infty\partial_t\vec{\mathrm{E}}(t) &= \vec{\nabla}\times\vec{\mathrm{H}}(t) - \vec{\mathrm{j}}_D(t)\,,\\ \partial_t\vec{\mathrm{j}}_D(t) &= \omega_D^2\vec{\mathrm{E}}(t) - \gamma_D\vec{\mathrm{j}}_D(t)\,.\end{aligned} \tag{31}$$

As soon as we want to simulate metals using a Drude model, we incur costs of one ADE per electric field component. For each additional Drude pole, we have to introduce yet another ADE per electric field component.

A.2.3. Lorentz oscillators

The Drude model is fairly limited when it comes to the description of metals in the visible regime. There, higher photon energies induce interband transitions of the electrons. Such transitions are not covered by the Drude model. Additional Lorentz oscillators provide a simple model for such processes as well as effects associated with bound charges. The resulting susceptibility

$$\chi_L(\omega) = \frac{\Delta\varepsilon_L\cdot\omega_L^2}{\omega_L^2 - \mathrm{i}\gamma_L\omega - \omega^2} \tag{32}$$

translates easily into the time-domain. In this expression, $\Delta\varepsilon_L$ represents the strength, ω_L the resonance frequency, and γ_L the damping coefficient of the oscillator. Again, we have used the sign convention (16). Inserting (32) into (29b), one obtains

$$-\mathrm{i}\omega\vec{\mathrm{j}}_L(\omega) = \Delta\varepsilon_L\omega_L^2\vec{\mathrm{E}}(\omega) + \vec{\mathrm{p}}_L(\omega)$$

with the abbreviation

$$\vec{\mathrm{p}}(\omega) = -\frac{\omega_L^2\vec{\mathrm{j}}(\omega) + \gamma_L\Delta\varepsilon_L\omega_L^2\vec{\mathrm{E}}(\omega)}{\gamma_L - \mathrm{i}\omega}\,.$$

The latter can be manipulated further to obtain

$$-\mathrm{i}\omega\vec{\mathrm{p}}(\omega) = -\omega_L^2\vec{\mathrm{j}}(\omega) - \gamma_L\Delta\varepsilon_L\omega_L^2\vec{\mathrm{E}}(\omega) - \gamma_L\vec{\mathrm{p}}(\omega)\,.$$

Again, applying a Fourier transform yields the time-domain formulation

$$\begin{aligned}\varepsilon_\infty\partial_t\vec{\mathrm{E}}(t) &= \vec{\nabla}\times\vec{\mathrm{H}}(t) - \vec{\mathrm{j}}_L(t)\\ \partial_t\vec{\mathrm{j}}_L(t) &= \Delta\varepsilon_L\omega_L^2\vec{\mathrm{E}}(t) + \vec{\mathrm{p}}_L(t)\\ \partial_t\vec{\mathrm{p}}_L(t) &= -\omega_L^2\vec{\mathrm{j}}_L(t) - \gamma_L\Delta\varepsilon_L\omega_L^2\vec{\mathrm{E}}(t) - \gamma_L\vec{\mathrm{p}}(t)\,.\end{aligned} \tag{33}$$

In contrast to the Drude model, we need to store *two* auxiliary fields $\vec{\mathrm{j}}_L$ and $\vec{\mathrm{p}}_L$. Yet, it is also possible to add an arbitrary number of Lorentz poles at the price of two additional auxiliary fields for each pole.

A.2.4. Implementation and efficiency considerations

The ADEs (31) and (33) can be readily included into the DG discretisation scheme. In analogy to the electromagnetic fields we expand the auxiliary fields in terms of Lagrange polynomials and obtain a set of expansion coefficients. As the auxiliary equations themselves do not include spatial derivatives, the spatial discretisation merely introduces the mass matrix $\mathscr{M}^{\triangle}$ on *both* sides. Multiplying with the inverse mass matrix recovers the very same auxiliary equations (31) and (33) we had for the continuous fields, but this time for the expansion coefficients. Hence, the system operator contains diagonal ADE blocks (compare Sect. 3.2.4 and Fig. 7). The absence of spatial derivatives in the ADEs also ensures that we can use the same flux as for the dielectric formulation.

At first glance it seems quite expensive to store additional field components. A single Lorentz pole, for example, doubles the number of unknowns in three-dimensional simulations. It is, however, important to note that in most simulations only a small fraction (typically below 10%) of the elements will be filled with a metal. A little book keeping allows us to restrict the necessary ADEs to these elements only. As a consequence, the storage of auxiliary fields can be avoided on all non-metallic elements. Furthermore, memory is usually not an issue for time-domain simulations, where the CPU time consumption generally is the limiting factor. However, because of the absence of matrix operations evaluating the Drude-Lorentz ADEs is very fast. As a consequence, the inclusion of metals in a DGTD framework merely leads to slight performance impairments.

To add these ADEs into an existing DG framework, two simple steps should be sufficient.

1. The expansion coefficients for the auxiliary fields must be integrated into the vector of expansion coefficients $\tilde{\mathbf{q}}$. As discussed previously in Sect. 3.2.4, we suggest to store all degrees of freedom of a single element in a contiguous block of memory.
2. For most elements, the system operator $\mathscr{H}$ without ADEs can be immediately applied. For all other elements, one first applies the existing operator. Then, one applies the corrections due to the dispersion ADEs in a post-processing step.

With the vector $\tilde{\mathbf{q}}$ and the system operator $\mathscr{H}$ thus modified, the existing Runge-Kutta solver can be used to do the

time stepping of both physical and auxiliary fields at the same time.

A.2.5. Material parameters

Quite a few publications have dealt with the experimental acquisition of material parameters for a wide range of metals. Johnson and Christy have tabularised the optical properties of gold, silver, and copper for various photon energies in the optical and near-infrared regime [70]. They also provide commonly used fit parameters for a Drude model. Ordal *et al.* have compiled tables for a number of metals, including non-noble metals like aluminium [85]. Fit parameters can also be found in their paper. Vial *et al.* improve on the simple Drude metal for gold using an additional Lorentz term [86]. They report a much better fit in the visible spectrum, especially for the imaginary part of the permittivity. Unfortunately, the present review cannot cover this topic in more detail, but the references may provide a starting point for a more detailed research.

A.2.6. Notes on the frequency-domain

As mentioned previously, auxiliary fields and differential equations are not needed in the frequency-domain. One can directly use tabulated parameters without the need to model a dispersion relation. In particular, one is not restricted to materials for which accurate analytical models are available.

However, $\mathscr{H}$ will now depend on the frequency. Eigenvalue problems as stated by Eq. (19) require a frequency-independent matrix $\mathscr{H}$. One can either approximate $\mathscr{H}(\omega)$ for some frequency of interest ω_0, or one can directly include the dispersion using the ADE technique. In the former case, one only considers solutions with frequencies close to ω_0 and hopes that the material parameters do not change too much in this regime. In the latter case, one obtains valid solutions in the whole spectral range, where "valid" means "with respect to the analytical model".

A.3. Optically anisotropic materials

For the derivation of the discontinuous Galerkin discretisation we have used Maxwell's equations (1) as its starting point. For simplification we have assumed the material properties to be both dispersionless and isotropic. In the previous section we have seen how to allow dispersive materials within the DG framework. In the following we will tackle the problem of anisotropic material parameters which are often encountered in the field of nanophotonics.

In particular, light propagation in crystals is often governed by optical anisotropy, e. g., birefringence. Liquid crystals are of special interest, as their optical axes can be realigned by applying external electric or magnetic fields. Another potential application is the simulation of effective materials. Certain structures interact with light on a subwavelength scale and, thus, may often be considered as effective media whose properties are determined by their respective building blocks. Specially tailored materials with effectively anisotropic permittivity tensors are potential candidates to facilitate optical cloaking, i. e., rendering an object "invisible" to an observer [7, 87].

A.3.1. Update equations

For simplicity we restrict our discussion to the important special case of two-dimensional systems in transverse-electric polarisation with relevant field components E_x, E_y, and H_z. We maintain an isotropic permeability μ and assume a tensorial permittivity

$$\underline{\varepsilon} = \begin{pmatrix} \varepsilon_{xx} & \varepsilon_{xy} \\ \varepsilon_{yx} & \varepsilon_{yy} \end{pmatrix}.$$

Retracing the steps of the spatial discretisation we finally arrive at [64]

$$\begin{aligned} \partial_t \tilde{\mathrm{E}}_x^{\Delta} = {} & \eta_{xx} \left(\mathscr{M}^{\Delta}\right)^{-1} \cdot \mathscr{S}_y^{\Delta} \tilde{\mathrm{H}}_z^{\Delta} \\ & + \eta_{xx} \left(\mathscr{M}^{\Delta}\right)^{-1} \cdot \mathscr{F}_f^{\Delta} \left[\hat{\mathrm{n}} \cdot \left(\vec{\mathrm{F}}_f - \vec{\mathrm{F}}_f^{*}\right)\right]_{E_x} \\ & - \eta_{xy} \left(\mathscr{M}^{\Delta}\right)^{-1} \cdot \mathscr{S}_x^{\Delta} \tilde{\mathrm{H}}_z^{\Delta} \\ & + \eta_{xy} \left(\mathscr{M}^{\Delta}\right)^{-1} \cdot \mathscr{F}_f^{\Delta} \left[\hat{\mathrm{n}} \cdot \left(\vec{\mathrm{F}}_f - \vec{\mathrm{F}}_f^{*}\right)\right]_{E_y}, \end{aligned} \tag{34}$$

where we have introduced the inverse permittivity tensor

$$\underline{\eta} = \begin{pmatrix} \eta_{xx} & \eta_{xy} \\ \eta_{yx} & \eta_{yy} \end{pmatrix} \equiv \underline{\varepsilon}^{-1}.$$

Equation (34) is the semi-discrete formulation for the field component E_x for anisotropic materials and, thus, represents a generalisation of Eq. (10). Please note that $\left[\hat{\mathrm{n}} \cdot \left(\vec{\mathrm{F}}_f - \vec{\mathrm{F}}_f^{*}\right)\right]_{E_x}$ represents the contribution due to the numerical flux across the element's face f which would be added exclusively to E_x under normal conditions, i. e., for isotropic materials.

An expression similar to Eq. (34) can be found for E_y. For H_z we find no modifications as compared to the original semi-discrete form. However, the numerical flux undergoes some changes which we describe in the next section.

A.3.2. Numerical flux

The original expression for the numerical upwind flux (6) involves field differences across the element interface and *scalar* material properties ε and μ via the impedance Z and the conductance Y. As we have a *tensorial* permittivity, it is clear that the numerical flux needs to be modified. To this end, one can either choose a different kind of flux, e. g., a central flux [63], or derive a modified expression for the upwind flux. As it turns out, the tensorial permittivity – at

a given interface between elements – can be reduced to an effective scalar value

$$\varepsilon_{\text{eff}} = \frac{\det \underline{\varepsilon}}{\hat{\mathrm{n}}^T \underline{\varepsilon} \hat{\mathrm{n}}} .$$

As before, $\hat{\mathrm{n}}$ denotes the outwardly oriented normal vector of an element's face. Using ε_{eff} instead of ε in the definitions of Z and Y while keeping the remaining formulae leads to a numerically stable scheme which preserves the convergence properties of the DG discretisation [64]. Thus, wave propagation in systems with anisotropic materials is properly accounted for.

A.4. Absorbing boundary conditions

Being a volume method, the discontinuous Galerkin technique cannot inherently solve infinitely extended systems. In contrast to, e. g., Green's functions techniques, the basis functions are spatially localised. In particular, there are no dedicated basis functions for radiation into the infinity of space. Though this property basically leads to the explicit time-stepping scheme and the sparsity of the system matrix, it complicates the simulation for many physical situations.

In nanophotonics, one is often interested in studying isolated particles, i. e., particles which are surrounded by free space and perhaps lie on a substrate. One might also be interested in a transmittance spectrum for a periodic array of building blocks. There, the incident illumination is both transmitted and reflected into free space.

All energy which is radiated into infinity will be lost for the actual system of interest, because infinity does not contain any scatterers. Hence, if we have to terminate our computational domain, its boundary should perfectly absorb incident light irrespective of its angle of incidence, wavelength, and polarisation. Analytical absorbing boundary conditions such as the Silver-Müller condition (see Sect. 2.4) locally enforce boundary conditions in order to inhibit backwards propagating, i. e., reflected waves. Such a procedure works well for only a limited number of angles of incidence, e. g., for normal incidence. Other angles undergo non-negligible reflections.

To minimise the deviation of the angle of incidence from some ideal value, say 0°, one can increase the distance between the boundary of the computational domain and the scatterer. For example, one might place a metallic nanoantenna in the centre of a large spherical computational domain. Thus, outgoing spherical waves will impinge on the boundary of the computational domain almost perpendicularly. The downside of this approach is that it significantly increases the number of unknowns, where most elements will basically simulate free space. For this reason analytical absorbing boundaries often do not suffice in practice.

An alternative technique has been found most useful to solve this problem. The idea is to divide the computational domain into an actual region of interest and a surrounding layer made of a strongly absorbing artificial material. This layer's material properties are designed to perfectly match the impedance of the region of interest, thus avoiding reflection at their mutual interface regardless of the angle of incidence. Hence, this layer is called a perfectly matched layer (PML).

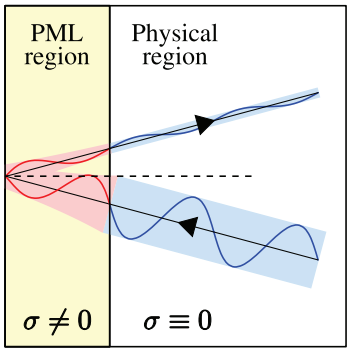


Figure 16 (online color at: www.lpr-journal.org) Sketch of the PML concept. The computational domain is divided into a physical region and an absorbing layer. Impinging waves pass the mutual interface of both regions without reflections. Within the PML the electromagnetic waves undergo continued attenuation – even after they have eventually been reflected at the computational domain's boundary. To the electromagnetic fields in the physical region the re-entering waves merely present minor perturbations.

Any wave which comes from the interior and hits the PML will pass the interface unobstructed (see Fig. 16). In the PML itself, the wave will continue to propagate until it impinges on the boundary of the computational domain. Depending on the boundary condition, e. g., a Silver-Müller absorbing boundary, the wave will be reflected at least partially. Again, it propagates through the PML and passes the interface to the region of interest. At this point, the wave's amplitude must have decayed sufficiently to avoid substantial influence on the physical system. Hence, the PML must consist of a lossy material. Nowadays, PMLs are a standard technique for the terminating open systems. Numerous formulations are available. In the following, we will present two different approaches.

A.4.1. Uniaxial perfectly matched layers

The uniaxial perfectly matched layer (UPML) approach is very convenient for the DG method [32, 61]. Consider an element with (scalar) material properties ε and μ. If this element were to be an element within a UPML layer, one would change its material properties to

$$\underline{\varepsilon}' \equiv \underline{\Lambda}\varepsilon, \quad \underline{\mu}' \equiv \underline{\Lambda}\mu, \quad \underline{\Lambda} \equiv \begin{pmatrix} \frac{s_y s_z}{s_x} & 0 & 0 \\ 0 & \frac{s_x s_z}{s_y} & 0 \\ 0 & 0 & \frac{s_x s_y}{s_z} \end{pmatrix} . \tag{35}$$

The PML parameters s_i, $i = x, y, z$ are chosen as

$$s_i(\omega) \equiv 1 - \frac{\sigma_i}{\mathrm{i}\omega} , \tag{36}$$

where σ_i controls the damping of a wave propagating along the i-direction. A numerical study to determine optimised UPML parameters was carried out in [32]. We observe that s_i is both complex-valued and dispersive. Please note that Eq. (36) also obeys the sign convention (16).

As this ansatz introduces a modified tensorial material constant $\underline{\varepsilon}'(\omega)$ with dispersive properties, we consequently apply the ADE technique as introduced in Sect. A.2.1. We are looking for an expression for

$$-\mathrm{i}\omega\vec{\mathrm{D}}(\omega) = -\mathrm{i}\omega\underline{\varepsilon}'(\omega)\vec{\mathrm{E}}(\omega)$$

which easily transforms into the time-domain. In the following, we will only consider the x-component of the electric field to simplify the notation. Analogous procedures lead to expressions for the other two components and for the magnetic field. Inserting (35) yields

$$-\mathrm{i}\omega D_x(\omega) = -\mathrm{i}\omega\varepsilon E_x + j_x(\omega) \tag{37a}$$

$$j_x(\omega) = -\mathrm{i}\omega\varepsilon\cdot\left(\frac{s_y s_z}{s_x} - 1\right)\cdot E_x(\omega)\,. \tag{37b}$$

Using definition (36) we can rewrite (37b) to obtain

$$-\sigma_x j_x(\omega) = -\mathrm{i}\omega\left[j_x(\omega) + \varepsilon\cdot(\sigma_x - \sigma_y - \sigma_z)\cdot E_z(\omega)\right] - \varepsilon\sigma_y\sigma_z E_x(\omega)\,.$$

With the auxiliary field

$$p_x(\omega) = j_x(\omega) + \varepsilon\cdot(\sigma_x - \sigma_y - \sigma_z)\cdot E_z(\omega)$$

we can eliminate j_x from both equation (37a) and the previous one. Applying the – then trivial – Fourier transform, one obtains

$$\partial_t E_x(t) = \varepsilon^{-1}\left[\vec{\nabla}\times\vec{\mathrm{H}}(t)\right]_x + (\sigma_x - \sigma_y - \sigma_z)\cdot E_x(t) - \varepsilon^{-1}p_x(t)$$

$$\partial_t p_x(t) = \left(\sigma_x^2 - \sigma_x\sigma_y - \sigma_x\sigma_z + \sigma_y\sigma_z\right)\cdot\varepsilon E_x(t) - \sigma_x p_x(t)\,.$$

Apparently, UPMLs introduce one additional field component per electromagnetic field component. Please note that in the region of interest, i. e., the domain which is surrounded by the PML, we have $\sigma_x = \sigma_y = \sigma_z = 0$. In this case, we do not need to store the auxiliary fields as both the initial condition and the time derivative of the field is zero. Following the DG discretisation procedure, one obtains the matrix form

$$\partial_t\tilde{\mathrm{E}}_x(t) = \varepsilon^{-1}\left(\mathscr{M}^{\triangle}\right)^{-1}\left[\vec{\mathscr{F}}^{\triangle}\times\tilde{\vec{\mathrm{H}}}(t)\right]_x + (\sigma_x - \sigma_y - \sigma_z)\cdot\tilde{\mathrm{E}}_x(t) - \varepsilon^{-1}\tilde{\mathrm{p}}_x(t)\,, \tag{38}$$

$$\partial_t\tilde{\mathrm{p}}_x(t) = \left(\sigma_x^2 - \sigma_x\sigma_y - \sigma_x\sigma_z + \sigma_y\sigma_z\right)\cdot\varepsilon\tilde{\mathrm{E}}_x(t) - \sigma_x\tilde{\mathrm{p}}_x(t)\,.$$

Please note the convenient absence of additional matrix-vector products in Eq. (38). This is an immediate consequence of the locality of the UPML ADEs which do not feature spatial derivatives.

Though UPMLs are fairly easily formulated for simple dielectric materials, the derivation is more complicated for dispersive materials. In this case, it is useful to write

$$\begin{aligned}
&-\mathrm{i}\omega D_x(\omega)\\
&= -\mathrm{i}\omega\left[\varepsilon_\infty + \chi(\omega)\right]\cdot\left[1 + \left(\frac{s_y s_z}{s_x} - 1\right)\right]\cdot E_x(\omega)\\
&= -\mathrm{i}\omega\varepsilon_\infty E_x(\omega) - \mathrm{i}\omega\left(\frac{s_y s_z}{s_x} - 1\right)\cdot E_x(\omega)\\
&\quad - \mathrm{i}\omega\chi(\omega)E_x(\omega) - \mathrm{i}\omega\left(\frac{s_y s_z}{s_x} - 1\right)\cdot\chi(\omega)E_x(\omega)\,.
\end{aligned}$$

The first three terms are the contribution from the dielectric material properties, the UPML for dielectric materials, and the dispersive material part as already known from previous discussions. The last term can be generically written as

$$\begin{aligned}
k_x(\omega) &= -\mathrm{i}\omega\left(\frac{s_y s_z}{s_x} - 1\right)\cdot\chi(\omega)E_x(\omega)\\
&= \left(\frac{s_y s_z}{s_x} - 1\right)\cdot j_x(\omega)\,,
\end{aligned}$$

where $j_x(\omega)$ is the polarisation current as given by Eq. (29b). Eventually, the combination of dispersive materials and a UPML layer leads to an additional auxiliary field for each electric field component. This is also true if the dispersion is modelled using multiple Drude and Lorentz poles.

A.4.2. Stretched coordinate formulation

Alternatively, PMLs can be implemented in a stretched-coordinate formulation [88, 89]. In essence, in the PML region we map real-valued x, y, z onto the complex plane. As a result, the product of a real-valued wave vector and the – now complex-valued – position vector gains an imaginary part. Thus, formerly propagating waves are converted into evanescently decaying ones. Stretched coordinates can be introduced into Maxwell's equations via the substitutions

$$\frac{\partial}{\partial x}\to\frac{1}{s_x}\frac{\partial}{\partial x}\,,\quad \frac{\partial}{\partial y}\to\frac{1}{s_y}\frac{\partial}{\partial y}\,,\quad \frac{\partial}{\partial z}\to\frac{1}{s_z}\frac{\partial}{\partial z}\,.$$

Similarly to the UPML case in the previous section, we define a complex-valued stretching factor

$$s_i(\omega) \equiv 1 - \frac{\sigma_i}{\mathrm{i}\omega - \alpha_i}\,. \tag{39}$$

As compared to Eq. (36), we have added the real number α which corresponds to a complex frequency shift. The above stretching factor is also commonly used for FDTD simulations [20], where one often replaces the "1" in Eq. (39) by a real stretching factor κ_i. However, instead of including a real stretching in the equations, one can simply stretch the mesh itself in the PML region. As this is easily accomplished during mesh generation, we can always set $\kappa_i = 1$ without

sacrificing generality. The remaining parameters must be carefully optimised to minimise the numerical errors. One such study was carried out in [89] for a three-dimensional reference system.

To highlight the essential points, we again restrict the derivation to the necessary changes to the E_x-component. In the frequency domain, the relevant component of Maxwell's equations read

$$-\mathrm{i}\omega\varepsilon E_x(\omega) = \frac{1}{s_y}\partial_y H_z(\omega) - \frac{1}{s_z}\partial_z H_y .$$

Using a similar trick as in the derivation of the UPML formulation, we obtain

$$\begin{aligned} -\mathrm{i}\omega\varepsilon E_x(\omega) &= \partial_y H_z(\omega) - \partial_z H_y(\omega) - G_{xy}(\omega) - G_{xz}(\omega), \\ G_{xy}(\omega) &= -\left(\frac{1}{s_y} - 1\right)\cdot \partial_y H_z(\omega) , \\ G_{xz}(\omega) &= \left(\frac{1}{s_z} - 1\right)\cdot \partial_z H_y(\omega) . \end{aligned}$$

Here, we have introduced two auxiliary fields G_{xy} and G_{xz}. They incorporate the effect on the x-component of the electric field due to a transformation along the y- or z-direction, respectively. Inserting the stretching factor (39) yields

$$\begin{aligned} -\mathrm{i}\omega G_{xy}(\omega) &= \sigma_y\partial_y H_z(\omega) - (\alpha_y+\sigma_y)\cdot G_{xy}(\omega) , \\ -\mathrm{i}\omega G_{xz}(\omega) &= -\sigma_z\partial_z H_y(\omega) - (\alpha_z+\sigma_z)\cdot G_{xz}(\omega) . \end{aligned}$$

Using the Fourier transform, one obtains the time-domain formulation

$$\begin{aligned} \partial_t E_x(t) &= \varepsilon^{-1}\cdot\left(\partial_y H_z(t) - \partial_z H_y(t) - G_{xy}(t) - G_{xz}(t)\right), \\ \partial_t G_{xy}(t) &= \sigma_y\partial_y H_z(t) - (\alpha_y+\sigma_y)\cdot G_{xy}(t) , \\ \partial_t G_{xz}(t) &= -\sigma_z\partial_z H_y(t) - (\alpha_z+\sigma_z)\cdot G_{xz}(t) . \end{aligned}$$

For each component of the electromagnetic fields we need to store two additional field components, i. e., the state vector $\mathbf{q}$ contains two additional components. Furthermore, we note that both auxiliary differential equations contain *spatial* derivatives. In the event of the DG discretisation spatial derivatives lead to coupling terms between neighbouring elements. Hence, the numerical flux has to be modified as well. Consequently, one obtains the semi-discrete scheme

$$\begin{aligned} \varepsilon^{\triangle}\partial_t\tilde{\mathrm{E}}_x^{\triangle} &= \left(\mathscr{M}^{\triangle}\right)^{-1}\left[\mathscr{S}^{\triangle}\times\tilde{\vec{\mathrm{H}}}^{\triangle}\right]_x - \left(\tilde{\mathrm{G}}_{xy}^{\triangle}+\tilde{\mathrm{G}}_{xz}^{\triangle}\right) \\ &\quad + \left(\mathscr{M}^{\triangle}\right)^{-1}\mathscr{F}_f^{\triangle}\cdot\left[\hat{\mathrm{n}}\cdot\left(\tilde{\vec{\mathrm{F}}}_f - \tilde{\vec{\mathrm{F}}}_f^{*}\right)\right]_{E_x} , \\ \partial_t\tilde{\mathrm{G}}_{xy}^{\triangle} &= \sigma_y\left(\mathscr{M}^{\triangle}\right)^{-1}\mathscr{S}_y^{\triangle}\cdot\tilde{\mathrm{H}}_z^{\triangle} - \left(\alpha_y^{\triangle}+\sigma_y^{\triangle}\right)\cdot G_{xy}^{\triangle} \\ &\quad + \sigma_y\left(\mathscr{M}^{\triangle}\right)^{-1}\mathscr{F}_f^{\triangle}\cdot\left[\hat{\mathrm{n}}\cdot\left(\tilde{\vec{\mathrm{F}}}_f - \tilde{\vec{\mathrm{F}}}_f^{*}\right)\right]_{G_{xy}} , \\ \partial_t\tilde{\mathrm{G}}_{xz}^{\triangle} &= -\sigma_z\left(\mathscr{M}^{\triangle}\right)^{-1}\mathscr{S}_z^{\triangle}\cdot\tilde{\mathrm{H}}_y^{\triangle} - \left(\alpha_z^{\triangle}+\sigma_z^{\triangle}\right)\cdot G_{xz}^{\triangle} \\ &\quad + \sigma_z\left(\mathscr{M}^{\triangle}\right)^{-1}\mathscr{F}_f^{\triangle}\cdot\left[\hat{\mathrm{n}}\cdot\left(\tilde{\vec{\mathrm{F}}}_f - \tilde{\vec{\mathrm{F}}}_f^{*}\right)\right]_{G_{xz}} . \end{aligned} \tag{40}$$

Note that $\hat{\mathrm{n}}\cdot\left(\tilde{\vec{\mathrm{F}}}_f - \tilde{\vec{\mathrm{F}}}_f^{*}\right)$ represents a state vector similar to $\tilde{\mathbf{q}}$. Hence, the indices E_x, G_{xy}, and G_{xz} correspond to the respective components of $\tilde{\mathbf{q}}$.

The missing parts in the stretched-coordinate formulation are the auxiliary components of the numerical fluxes. Observing that the spatial derivatives in the ADEs represent both terms of the original curl operator, it follows that

$$\left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{E_x} = \left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{G_{xy}} + \left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{G_{xz}} .$$

Rewriting the definition (6) and identifying the origin of the flux as the curl operator, we have

$$\left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{\vec{\mathrm{E}}} = \hat{\mathrm{n}}\times\frac{Z^{+}\cdot\Delta\vec{\mathrm{H}} - \alpha\cdot\hat{\mathrm{n}}\times\Delta\vec{\mathrm{E}}}{Z^{+}+Z^{-}} .$$

Having the definition of the numerical flux in mind, it is evident that the operator $\hat{\mathrm{n}}\times$ represents the curl operator. Splitting this operator into two parts [89] yields

$$\begin{aligned} \left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{G_{xy}} &= n_y\left(\frac{Z^{+}\cdot\Delta\vec{\mathrm{H}} - \alpha\cdot\hat{\mathrm{n}}\times\Delta\vec{\mathrm{E}}}{Z^{+}+Z^{-}}\right)_z , \\ \left[\hat{\mathrm{n}}\cdot\left(\vec{\mathrm{F}}-\vec{\mathrm{F}}^{*}\right)\right]_{G_{xz}} &= -n_z\left(\frac{Z^{+}\cdot\Delta\vec{\mathrm{H}} - \alpha\cdot\hat{\mathrm{n}}\times\Delta\vec{\mathrm{E}}}{Z^{+}+Z^{-}}\right)_y . \end{aligned}$$

In conclusion, employing a stretched coordinate formulation into the DG framework requires us to split the numerical flux. This is conceptually similar to the split field approach originally used by Berenger to introduce PMLs.

A.4.3. Implementation and discussion

From an analytical point of view, both the UPML and the stretched-coordinate formulation should be equivalent. In their numerical performance, however, they show significant differences. First of all, the UPML formulation introduces one additional auxiliary field for each electromagnetic field component. In comparison, the stretched coordinate ADEs require *two* additional fields per component of the electric and magnetic fields. Consequently, UPMLs are more memory efficient.

However, this is only true if the PMLs are used to terminate dielectric materials. Since absorption is included via a modified material tensor in the UPML formulation, additional auxiliary equations are necessary for the termination of dispersive materials. On the other hand, the coordinate stretching approach is material-independent since the spatial derivatives in Maxwell's curl equations are directly modified. Hence, stretched coordinates can be combined with dispersive (and in principle even nonlinear) materials without further changes to the numerical scheme.

Concerning the computational effort it is helpful to compare Eqs. (38) and (40). Obviously, the inclusion of auxiliary fields in the UPML formulation does not lead to additional matrix-vector products. Instead, field components at any

given node i only depend on other fields at the very same node. Hence, UPMLs lead to a local formulation. In contrast, stretched-coordinate ADEs feature spatial derivatives which lead to matrix-vector products for both the actual derivatives and the associated flux through the element boundaries. Hence, the stretched-coordinate formulation requires significantly more computational effort.

Apart from the material independence of the stretched-coordinate PMLs another point works in their favour. The complex frequency-shifting parameter α in Eq. (39) presents an additional parameter which can be tuned for optimal performance. In particular, $\alpha \neq 0$ greatly improves the absorption of low-frequency waves. Compared to the UPML formulation, this can reduce numerical reflection errors by one order of magnitude [89]. Both formulations, however, lead to significantly more accurate results than a simple termination of the computational domain using a Silver-Müller boundary condition (see Sect. 2.4). Nevertheless, it is advantageous to use a Silver-Müller boundary condition *in conjunction* with PMLs, which helps to slightly reduce the reflection errors. More importantly, the error becomes less sensitive to the PML parameters [89]. Thus, we conclude that PMLs should always be terminated with a Silver-Müller boundary condition, which is easily (and efficiently) implemented via a modified numerical flux (see Sect. 2.4).

A.5. Curvilinear elements

One of the key strengths of the DG discretisation is that it does not rely on structured grids, but employs geometry-adapted meshes instead. This allows to accurately represent thin films, small features, tilted planes and more. However, curved objects remain challenging to model, particularly since we strive for meshes with as few elements as possible. Specifically, we wish to avoid very small elements which drastically limit the maximum stable time step. For this reason, it makes sense to introduce curvilinear elements [34,90] as a means to reduce the number of elements while at the same time increasing the accuracy of the geometrical representation.

When revisiting the derivation of the semi-discrete system (10), we find that no assumptions were made on the specific shape of the elements. The only place where the shape of an element actually enters is in the construction of the matrices (9). As discussed in Sect. 2.6, for straight-sided elements we can generate the matrices of each element from only five template matrices, evaluated on a straight-sided reference element V_{ref}.

If we wish to represent a curved element, this necessarily leads to a non-affine mapping between the coordinates $\vec{\mathrm{r}} = (x, y, z)$ of the curved element and the coordinates $\vec{\mathrm{s}} = (u, v, w)$ of the straight-sided reference element. This mapping can be described by a spatially dependent Jacobian matrix $\mathscr{J}^{\triangle}(\vec{\mathrm{s}})$.

For a given non-affine mapping, we then have to evaluate the matrices individually for every curved element. In practice, one typically calculates the volume and surface integrals numerically by employing suitable cubature rules [91]. Then, the mass, stiffness and face-mass matrices need to be stored individually for every curved element. This leads to a dramatic increase in the memory consumption of the calculation, but also significantly improves the accuracy of the geometric description. Fortunately, in most systems only a small fraction of the elements actually needs to be curvilinear. Therefore, even rather large three-dimensional systems can be treated without running into memory constraints.

An alternative approach which requires considerably less memory than the conventional scheme presented here was recently proposed by Warburton [92]. In essence, he replaces the Lagrange polynomials by Lagrange polynomials which are weighted by one over the inverse square of the Jacobian:

$$L_i(\vec{\mathrm{r}}) \rightarrow \frac{L_i(\vec{\mathrm{r}})}{\sqrt{\det \mathscr{J}^{\triangle}}}.$$

Employing this substitution renders the mass matrices independent of the element. Thus, only a single mass matrix needs to be stored. On the other hand, this modification also introduces correction terms which eventually require numerical integrations. As a result, this approach trades memory against speed.

The key problem in using curvilinear elements is finding an appropriate mapping between the curved elements and the reference element. There are multiple approaches to tackle this problem: For simple geometries such as an individual cylinder or a sphere, it might be possible to find an analytic expression for the coordinate transformation. For more complex systems, one can try to numerically modify the interpolation nodes such that they lie on the curved surface [26,34]. This approach allows an almost exact treatment of arbitrary structures, but it significantly increases code complexity since one needs access to the geometric description of the physical system during the setup of the DG calculation. Finally, the most common approach is to employ a mesh generator which generates higher-order elements. Such higher-order elements usually contain additional vertices, for example quadratic elements contain one additional vertex at the centre of every edge. For such a higher-order element, one can then explicitly generate a polynomial mapping [24,93].

While the last approach only yields a (usually lower-order) polynomial approximation to the curved structure, it represents the most general and convenient method. In practice, already quadratic elements are often sufficient to reach the desired accuracy.

A.6. Data recording and analysis

The natural output of discontinuous Galerkin simulations are either time- or frequency-dependent expansion coefficients. Using the local expansion basis, it is easy to convert these coefficients into field distributions.

For comparisons with experiments it is important to obtain quantities which have actually been measured. The

field distribution itself is usually not accessible. Most experiments cannot even resolve the time-dependence of the electromagnetic fields. Instead, frequency spectra of integrated quantities like the transmittance of a material slab are often desired. This section briefly describes a few recurring tasks related to simulation data recording and analysis.

A.6.1. On-the fly Fourier transform

Most systems are experimentally characterised in the frequency-domain. Obviously, such quantities are naturally obtained from frequency-domain computations. This section, however, illustrates how to convert time-dependent data into spectral data.

Obviously, the Fourier transform

$$f(\omega) = \int_{-\infty}^{\infty} f(t) \cdot \exp(\mathrm{i}\omega t)\, \mathrm{d}t \tag{41}$$

will do the trick. More precisely, as we deal with discretised time steps a discrete Fourier transform is needed, i. e.,

$$f(\omega) \approx \sum_{i=1}^{T} f(t_i) \cdot \exp(\mathrm{i}\omega t_i) \cdot \Delta t_i\,. \tag{42}$$

Here, $f(t)$ is an arbitrary time-dependent quantity, $f(\omega)$ its Fourier transform for frequency ω, T is the number of time steps, and Δt_i represents the i-th time step. The popular fast Fourier transform (FFT) [94, 95] is not the ideal choice to evaluate Eq. (42). Though it is arguably the fastest method to extract the complete information from a given time series, it is subject to a few limitations:

- The time steps must be equally large. Time stepping schemes with adaptive time steps cannot be implemented if we want to use conventional FFT. In principle, however, generalisations to non-equidistant time steps are available, e. g., see [96].
- Due to the divide and conquer strategy used by FFT, it is essential that the complete time series is available at the same time. For a single data point this is not a problem. If one is, however, interested in recording fields on a plane or in a volume, one often ends up with tens of thousands of data points, each with its own series of time steps. As it turns out, storing these series can be a severe issue, even if the simulation itself easily fits into the main memory.

Furthermore, the full spectrum is not always needed. Most often, one is interested in a certain frequency range with a certain resolution. For example, a metamaterial might show an interesting behaviour between 800 and 1800 nm. A resolution of 5 nm leaves us with a total of 201 wavelengths and, thus, frequencies of interest. As the number of frequencies is orders of magnitudes smaller than the number of time steps, it is convenient to perform an on-the-fly Fourier transform.

To this end, we specify frequencies ω_j for which the fields f should be recorded. After a time step, one updates the Fourier coefficients according to

$$f(\omega_j) := f(\omega_j) + f(t_c) \cdot \exp(\mathrm{i}\omega_j t_c) \cdot \Delta t_c\,.$$

Here, the subscript "c" denotes the current time step. Using this iterative update procedure, it is not necessary to store the complete time series, as we only need one value a time. In particular, the memory consumption of the on-the-fly Fourier transform does not depend on the number of time steps.

However, its accuracy does. The transition from continuous time (41) to discrete time (42) caps the limits of the integration. If the conditions

$$f(t) \equiv 0 \quad \text{for} \quad t \in (-\infty, 0]\,,$$
$$f(t) \equiv 0 \quad \text{for} \quad t \in [t_f, \infty)$$

are satisfied, the error of the discrete Fourier transform only depends on Δt, which is usually sufficiently small due to the explicit time stepping scheme. The first condition is easily fulfilled by setting all initial expansion coefficients to zero. The second condition states that the fields must decay during the simulation, where t_f is the final simulation time. Even though this can never be achieved perfectly, the system needs to be simulated long enough to allow for a sufficient decay. Problems arise especially for systems with high quality factors because they require extremely long simulation times. If it is too short, oscillations appear in the spectrum.

In essence, the Fourier transform – whichever way implemented – allows us to obtain a spectrum from just a single time-domain simulation. This is an important point, as we do not have to launch multiple simulations with incident waves of slightly different frequencies. It is often cited as a key benefit of time-domain methods.

A.6.2. Harmonic inversion techniques

In the previous section we have discussed the Fourier transform as a means to extract frequency-domain quantities out of time-domain simulations. Inconveniently, the accuracy of the discrete Fourier transform heavily depends on the simulation time. The underlying reason is that the Fourier transform applies to *any* time signal irrespective of its physical origin and, thus, leads to very general and conservative results. In the context of nanophotonics, however, we are usually interested in resonances and modes, each characterised by complex oscillation frequencies ω_k (which include finite life time effects via the imaginary part) and amplitudes a_k. Hence, we decompose the electric field, or any other field, as

$$E_x(\vec{r}_0, t) = \sum_{k=1}^{K} a_k \cdot \exp(-\mathrm{i}\omega_k t)\,.$$

The harmonic inversion problem is to obtain the unknown parameters for a given time series. By specifying a finite number K of modes within a certain frequency range, it is

possible to reduce this problem to the solution of a generalised eigenvalue problem [97]. Sophisticated implementations such as `Harminv` are freely available [98] and provide resonance frequencies and corresponding quality factors for the input time signal.

Harmonic inversion techniques are most useful for high-Q resonators, where reliable results concerning resonance frequencies and quality factors are available after just a few optical cycles [44]. On-the-fly Fourier transforms are better suited for recording fields on planes or in volumes.

A.6.3. Power flux, transmittance, and cross-sections

So far, we have only considered how to obtain spectral information from time-domain simulations. However, to be able to directly compare our simulations with most experiments one crucial element is still missing. Even though exact field distributions are easily obtained from both time- and frequency-domain calculations, corresponding measurements are usually not available. Instead, nanophotonic experiments typically provide us with intensity or power spectra taken by collecting a beam with a lens and focussing it on a detector plane. Normalising the resulting spectrum for an array of scatterers against a reference spectrum leads, for example, to transmittance and reflectance spectra. More sophisticated setups even allow the measurement of scattering, absorption, and extinction cross-sections of individual particles [66, 99]. This section explains how to obtain such data from DG simulations.

The time-averaged Poynting vector

$$\vec{\mathrm{S}}(\omega) = \tfrac{1}{2}\,\mathrm{Re}\left\{\vec{\mathrm{E}}(\omega) \times \vec{\mathrm{H}}^{*}(\omega)\right\} \tag{43}$$

describes the power flow per time and area averaged over a full period $T = 2\pi\omega^{-1}$. To calculate the time-averaged Poynting vector one needs *frequency-dependent* electromagnetic fields. In particular, the time-averaged Poynting vector cannot be obtained as a Fourier transform of $\vec{\mathrm{E}}(t) \times \vec{\mathrm{H}}(t)$. In case of DGTD simulations, the techniques presented in Sect. A.6.1 must be applied to the individual field components to obtain frequency-domain quantities.

The time-averaged rate at which energy flows through a (detector) surface D is given by

$$W(\omega) = \int_{D} \hat{\mathrm{n}} \cdot \vec{\mathrm{S}}(\omega)\, \mathrm{d}^2 r. \tag{44}$$

As before, $\hat{\mathrm{n}}$ is the unit normal of surface D. We now want to calculate the transmittance of a periodic array of scatterers, which is modelled by a single scatterer and periodic boundary conditions normal to the direction of propagation (see Fig. 17). Along the latter, we terminate both ends of the system with perfectly matched layers to absorb transmitted and reflected light. A plane wave is injected using the total-field/scattered-field (TF/SF) technique (see Sect. A.1.1), whose Poynting vector we denote by $\vec{\mathrm{S}}_{\mathrm{inc}}(\omega)$. A planar detector surface D is included behind the array. On

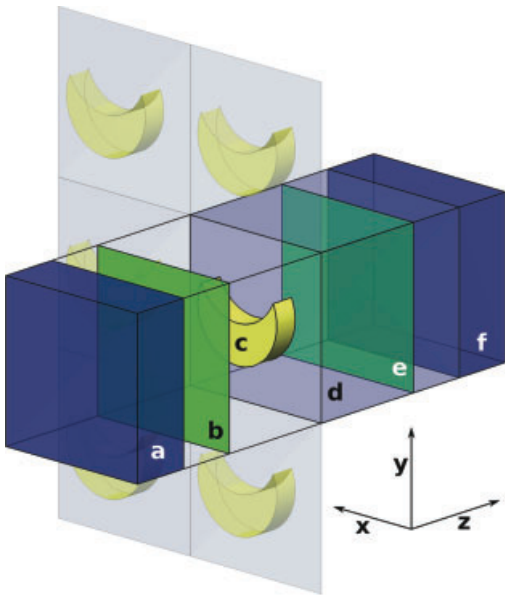


Figure 17 (online color at: www.lpr-journal.org) Sketch of a typical setup for transmittance calculations. A scatterer, in this case a crescent moon shaped particle (c) is placed on a substrate (d) in a unit cell, which is periodically arranged along the x- and y-directions via periodic boundary conditions. The illumination is provided via the total-field/scattered-field injection plane (b). The reflected and transmitted waves are absorbed by the perfectly matched layers (a) and (f). Integration and subsequent normalisation of the fluxes on the planes (b) and (e) yields the reflectance and transmittance, respectively.

this surface we calculate the Poynting vector $\vec{\mathrm{S}}_{\mathrm{trans}}(\omega)$ of the transmitted light. With this information the transmittance T is easily obtained via

$$T(\omega) = \frac{\int_D \hat{\mathrm{n}} \cdot \vec{\mathrm{S}}^{\mathrm{trans}}(\omega)\, \mathrm{d}^2 r}{\int_D \hat{\mathrm{n}} \cdot \vec{\mathrm{S}}^{\mathrm{inc}}(\omega)\, \mathrm{d}^2 r}. \tag{45}$$

Similarly, scattering, absorption, and extinction cross sections can be determined with a slightly modified setup (see Fig. 18). Instead of using periodic boundaries, we surround the whole computational domain by perfectly matched layers (see Sect. A.4). Within the computational domain we define a closed surface D as the TF/SF interface. Within the total-field region a scatterer interacts with an injected plane wave, whose Poynting vector is again given by $\vec{\mathrm{S}}_{\mathrm{inc}}(\omega)$. The scattering and absorption cross sections C^{scat} and C^{abs}, respectively, are given by

$$C^{\mathrm{scat}}(\omega) = \frac{\int_D \hat{\mathrm{n}} \cdot \vec{\mathrm{S}}^{\mathrm{scat}}(\omega)\, \mathrm{d}^2 r}{\left|\vec{\mathrm{S}}_{\mathrm{inc}}(\omega)\right|},$$

$$C^{\mathrm{abs}}(\omega) = -\frac{\int_D \hat{\mathrm{n}} \cdot \vec{\mathrm{S}}^{\mathrm{tot}}(\omega)\, \mathrm{d}^2 r}{\left|\vec{\mathrm{S}}_{\mathrm{inc}}(\omega)\right|}.$$

In the previous formula we have used the *outwardly* oriented normal vector $\hat{\mathrm{n}}$ of the TF/SF interface. The scattered-field and total-field Poynting vectors $\vec{\mathrm{S}}^{\mathrm{scat}}$ and $\vec{\mathrm{S}}^{\mathrm{tot}}$ on the same

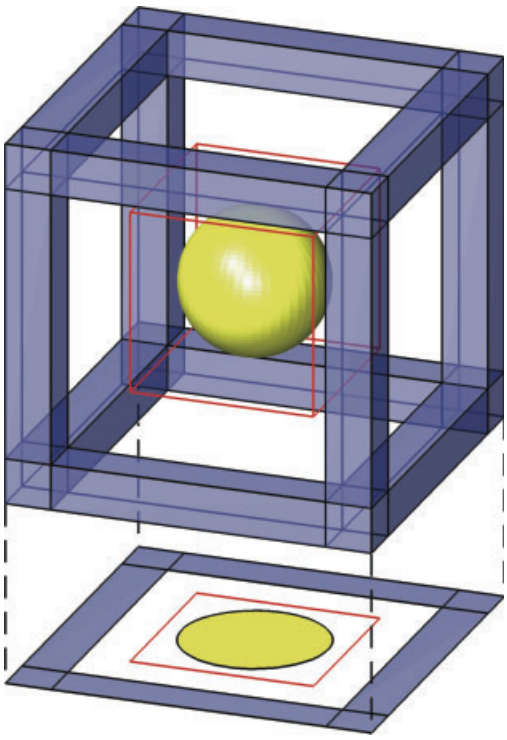


Figure 18 (online color at: www.lpr-journal.org) Sketch of a typical scattering simulation setup. A scatterer, in this case a sphere, is placed within the total-field region, i. e., inside the inner box. The box itself is used as an integration surface according to Sect. A.6.3. The computational domain is surrounded by PMLs along all directions to absorb outgoing scattered waves. For better visibility only corner and edge regions of the boundary layer is shown. The plane below shows a slice through the centre of the system.

interface can be directly taken from the simulation, as the discontinuous Galerkin technique comprises field values on both sides of an interface. The extinction cross section C^{ext}, finally, is given by

$$C^{\mathrm{ext}}(\omega) = C^{\mathrm{scat}}(\omega) + C^{\mathrm{abs}}(\omega) .$$

Acknowledgements. We acknowledge financial support by the Center for Functional Nanostructures (CFN) of the Deutsche Forschungsgemeinschaft (DFG) within subproject A1.2. The research of MK is supported through the Karlsruhe School of Optics & Photonics (KSOP) at the Karlsruhe Institute of Technology (KIT) and by the Studienstiftung des Deutschen Volkes. The Young Investigator Group of JN received financial support by the "Concept for the Future" of the KIT within the framework of the German Excellence Initiative. The authors would like to thank the anonymous reviewers of this article for numerous helpful suggestions.

Received: 23 December 2010, **Revised:** 4 March 2011, **Accepted:** 18 March 2011

Published online: 3 May 2011

Key words: Nanophotonics, plasmonics, light-matter interaction, Maxwell's equations, time-domain simulations, frequency-domain simulations, discontinuous Galerkin methods.

Kurt Busch received his Ph. D. degree from the Physics Department of Universität Karlsruhe (TH), Karlsruhe, Germany, in 1996. He was a postdoctoral researcher at the University of Toronto from 1997 to 1999 and head of a Junior Research Group within the Emmy-Noether Programme of the Deutsche Forschungsgemeinschaft (DFG) at Institut für Theorie der Kondensierten Materie at Universität Karlsruhe (TH) from 2000 to 2003. In 2004 he joined the University of Central Florida, Orlando, FL, USA, as an associate professor with a joint appointment between the Department of Physics and the College of Optics & Photonics: CREOL & FPCE. Since 2005 he has been a professor at Institut für Theoretische Festkörperphysik at Universität Karlsruhe (TH). He has been a member of the DFG-Center for Functional Nanostructures (CFN) since 2001 and of the Karlsruhe School of Optics & Photonics (KSOP) since 2006. His research interests lie in the areas of computational photonics and the theory of light propagation and light-matter interaction in strongly scattering systems. This research has lead to the Carl Zeiss Research Award of 2006.

Michael König received his diploma degree from the Physics Department of the Universität Karlsruhe, Germany in 2007. Since then he has been working on his PhD project at the Karlsruhe Institute of Technology (KIT). His research interests comprise numerical simulation tools for nanophotonics and metamaterials.

Jens Niegemann received his Dipl. Phys. degree in 2004 from the Universität Karlsruhe, Germany. In 2009 he received a PhD from the Karlsruhe Institute of Technology (KIT). Since 2009 he is the leader of the Young Investigator Group "Computational Nano-Photonics" at the KIT. His research interests include numerical methods in electromagnetics and wave propagation in nanostructured materials.

References

[1] K. Busch, G. von Freymann, S. Linden, S. F. Mingaleev, L. Tkeshelashvili, and M. Wegener, Phys. Rep.-Rev. Sec. Phys. Lett. **444**, 101 (2007).

[2] J. D. Joannopoulos, S. G. Johnson, J. N. Winn, R. D. Meade, Photonic Crystals: Molding the Flow of Light, 2nd ed. (Princeton Univerity Press, Princeton, 2008).

[3] K. Sakoda, Optical properties of photonic crystals, 2nd ed. (Springer, Berlin, 2005).

[4] J.-M. Lourtioz, H. Benisty, V. Berger, J.-M. Gérard, D. Maystre, A. Tchelnokov, and D. Pagnoux, Photonic crys-

tals: Towards nanoscale photonic devices, 2nd ed. (Springer, Berlin, 2008).
[5] G. Dolling, M. Wegener, C. M. Soukoulis, and S. Linden, Opt. Lett. **32**, 53 (2007).
[6] J. B. Pendry, Phys. Rev. Lett. **85**, 3966 (2000).
[7] W. Cai, U. K. Chettiar, A. V. Kildishev, and V. M. Shalaev, Nature Photon. **1**, 224 (2007).
[8] M. E. Stewart, C. R. Anderton, L. B. Thompson, J. Maria, S. K. Gray, J. A. Rogers, and R. G. Nuzzo, Chem. Rev. **108**, 494 (2008).
[9] A. N. Grigorenko, N. W. Roberts, M. R. Dickinson, and Y. Zhang, Nature Photon. **2**, 365 (2008).
[10] B. Hecht, B. Sick, U. P. Wild, V. Deckert, R. Zenobi, O. J. F. Martin, and D. W. Pohl, J. Chem. Phys. **112**, 7761 (2000).
[11] A. M. Armani, R. P. Kulkarni, S. E. Fraser, R. C. Flagan, and K. J. Vahala, Science **317**, 783 (2007).
[12] J. Hu, X. Sun, A. Argawal, and L. C. Kimerling, J. Opt. Soc. Am. B **26**, 1032 (2009).
[13] Y. Sun and X. Fan, Anal. Bioanal. Chem. **399**, 205 (2011).
[14] C. Hafner, Phys. Stat. Sol. B **244**, 3435 (2007).
[15] K. Busch, C. Blum, A. M. Graham, D. Hermann, M. Köhl, P. Mack, and C. Wolff, J. Mod. Opt. **58**, 365 (2011).
[16] C. Blum, C. Wolff, and K. Busch, Opt. Lett. **36**, 307 (2011).
[17] L. Li, J. Opt. Soc. Am. A **13**, 1024 (1996).
[18] L. Li, J. Opt. Soc. Am. A **14**, 2758 (1997).
[19] S. Essig and K. Busch, Opt. Express **18**, 23258 (2010).
[20] A. Taflove and S. C. Hagness, Computational Electrodynamics: The Finite-Difference Time-Domain Method, 3rd ed. (Artech House, Boston, 2005).
[21] A. Farjadpour, D. Roundy, A. Rodriguez, M. Ibanescu, P. Bermel, J. D. Joannopoulos, and S. G. Johnson, Opt. Lett. **31**, 2972 (2006).
[22] A. Deinega and I. Valuev, Opt. Lett. **32**, 3429 (2007).
[23] A. F. Oskooi, C. Kottke, and S. G. Johnson, Opt. Lett. **34**, 2778 (2009).
[24] J. Jin, The finite element method in electromagnetics, 2nd ed. (John Wiley & Sons, New York, 2002).
[25] P. Monk, Finite element methods for Maxwell's equations (Oxford Science Publications, Oxford, 2003).
[26] J. S. Hesthaven and T. Warburton, Nodal Discontinuous Galerkin Methods – Algorithms, Analysis, and Applications (Springer, Berlin, 2007).
[27] J. S. Hesthaven and T. Warburton, J. Comput. Phys. **181**, 186 (2002).
[28] B. Cockburn and C.-W. Shu, J. Comput. Phys. **141**, 199 (1998).
[29] Y. Xu and C. W. Shu, Commun. Comput. Phys. **7**, 1 (2010).
[30] W. H. Reed and T. R. Hill, Los Alamos Scientific Laboratory Report, LA-UR-73-479 (1973).
[31] B. Cockburn, S. Hou, and C.-W. Shu, Math. Comput. **54**, 545 (1990).
[32] J. Niegemann, M. König, K. Stannigel, and K. Busch, Photon. Nanostruct. Fundam. Appl. **7**, 2 (2009).
[33] G. Tang, R. L. Panetta, and P. Yang, Appl. Opt. **49**, 2827 (2010).
[34] A. Hille, R. Kullock, S. Grafström, and L. M. Eng, J. Comput. Theor. Nanosci. **7**, 1581 (2010).
[35] Y. Shi and C.-H. Liang, Prog. Electromagn. Res. **63**, 171 (2006).
[36] G. Cohen, X. Ferrieres, and S. Pernet, J. Comput. Phys. **217**, 340 (2006).
[37] H. Fahs, Int. J. Numer. Anal. Model. **6**, 193 (2009).
[38] H. Fahs and S. Lanteri, J. Comput. Appl. Math. **234**, 1088 (2010).
[39] J. Schöberl, Comput. Visual. Sci **1**, 41 (1997), http://sourceforge.net/projects/netgen-mesher/.
[40] H. Si, Tetgen, http://tetgen.berlios.de.
[41] C. Geuzaine and J.-F. Remacle, Int. J. Numer. Methods Eng. **79**, 1309 (2009); http://geuz.org/gmsh/.
[42] T. Warburton, J. Eng. Math. **56**, 247 (2006).
[43] R. Pasquetti and F. Rapetti, Numer. Algorithms (Netherlands) **55**, 349–366 (2010).
[44] J. Niegemann, W. Pernice, and K. Busch, J. Opt. A, Pure Appl. Opt. **11**, 114015 (2009).
[45] M. H. Carpenter and C. A. Kennedy, NASA Technical Memorandum 109112 (1994).
[46] R. Diehl, K. Busch, and J. Niegemann, J. Comput. Theor. Nanosci. **7**, 1572–1580 (2010).
[47] J. Niegemann, R. Diehl, and K. Busch, submitted (2010).
[48] H. O. Kreiss and L. Wu, Appl. Numer. Math. **12**, 213–227 (1993).
[49] J. Niegemann and K. Busch, AIP Conf. Proc. **1147**, 22–29 (2009).
[50] R. B. Lehoucq and D. C. Sorensen, SIAM J. Matrix Anal. Appl. **17**, 789 (1996), http://www.caam.rice.edu/software/ARPACK/.
[51] R. J. Spiteri and S. J. Ruuth, SIAM J. Matrix Numer. Anal., **40**, 469 (2002).
[52] A. Klöckner, T. Warburton, J. Bridge, and J. S. Hesthaven, J. Comput. Phys. **228**, 7863 (2009).
[53] D. Sármány, F. Izsák, and J. J. W. van der Vegt, J. Sci. Comput. **44**, 219 (2010).
[54] B. Cockburn, F. Li, and C.-W. Shu, J. Comput. Phys. **194**, 588 (2004).
[55] O. Schenk and K. Gärtner, Electron. Trans. Numer. Anal. **23**, 158 (2006).
[56] T. A. Davis, ACM Trans. Math. Softw. **30**, 165 (2004).
[57] G. L. G. Sleijpen and D. Fokkema, Elec. Trans. Numer. Anal. **1**, 11 (1993).
[58] Y. Saad and M. H. Schultz, SIAM J. Sci. Stat. Comput. **7**, 856 (1986).
[59] V. Dolean, S. Lanteri, and R. Perrussel, J. Comput. Phys. **227**, 2044 (2008).
[60] K. Stannigel, M. König, J. Niegemann, and K. Busch, Opt. Express **17**, 14934 (2009).
[61] T. Lu, P. Zhang, and W. Cai, J. Comput. Phys. **200**, 549 (2004).
[62] X. Ji, T. Lu, W. Cai, and P. Zhang, J. Lightwave Technol. **23**, 3864 (2005).
[63] S. Chun and J. S. Hesthaven, Commun. Comput. Phys. **2**, 611 (2007).
[64] M. König, K. Busch, and J. Niegemann, Photon. Nanostruct. Fundam. Appl. **8**, 303 (2010).
[65] X. Ji, W. Cai, and P. Zhang, Int. J. Numer. Methods Eng. **69**, 308 (2006).
[66] N. Feth, M. König, M. Husnik, K. Stannigel, J. Niegemann, K. Busch, M. Wegener, and S. Linden, Opt. Express **18**, 6545 (2010).
[67] T. Lu, W. Cai, and P. Zhang, IEEE Trans. Geosci. Remote Sens. **43**, 72 (2005).
[68] V. Dolean, H. Fahs, L. Fezoui, and S. Lanteri, J. Comput. Phys. **229**, 512 (2010).
[69] E. Montseny, S. Pernet, X. Ferriéres, and G. Cohen, J. Comput. Phys. **227**, 6795 (2008).

[70] P. B. Johnson and R. W. Christy, Phys. Rev. B **6**, 4370 (1972).
[71] M. J. Grote, A. Schneebeli, and D. Schötzau, IMA J. Numer. Anal. **28**, 440 (2008).
[72] S. Piperno, ESAIM-Math. Model. Numer. Anal.-Model. Math. Anal. Numer. **40**, 815 (2006)
[73] C. Fumeaux, D. Baumann, P. Leuchtmann, and R. Vahldieck, IEEE Trans. Microwave Theory Tech. **52**, 1067 (2004).
[74] M. Dumbser, M. Käser, and E. F. Toro, Geophys. J. Int. **171**, 695 (2007).
[75] S. Schomann, N. Gödel, T. Warburton, and M. Clemens, IEEE Trans. Magn. **46**, 3504 (2010).
[76] N. Gödel, S. Schomann, T. Warburton, and M. Clemens, IEEE Trans. Magn. **46**, 2735 (2010).
[77] P. Mandel, Theoretical problems in cavity nonlinear optics (Cambridge University Press, Cambridge, 2005).
[78] J. E. Sipe, V. C. Y. So, M. Fukui, and G. I. Stegeman, Phys. Rev. B **21**, 4389 (1980).
[79] M. Aeschlimann, M. Bauer, D. Bayer, T. Brixner, F. J. García de Abajo, W. Pfeiffer, M. Rohmer, C. Spindler, and F. Steeb, Nature **446**, 301 (2007).
[80] G. Mur, IEEE Trans. Electromagn. Compat. **23**, 377 (1981).
[81] L. Novotny and B. Hecht, Principles of Nano-Optics (Cambridge University Press, New York, 2006).
[82] J. A. Kong, Electromagnetic Wave Theory (EMW Publishing, Cambridge, 2005).
[83] M. I. Stockman, S. V. Faleev, and D. J. Bergman, Phys. Rev. Lett. **88**, 067402 (2002).
[84] X. Li and M. I. Stockman, Phys. Rev. B **77**, 195109 (2008).
[85] M. A. Ordal, L. L. Long, R. J. Bell, S. E. Bell, R. R. Bell, R. W. Alexander Jr, and C. A. Ward, Appl. Opt. **22**, 1099 (1983).
[86] A. Vial, A.-S. Grimault, D. Macías, D. Barchiesi, and M. L. de la Chapelle, Phys. Rev. B **71**, 085416 (2005).
[87] J. Li and J. B. Pendry, Phys. Rev. Lett. **101**, 203901 (2008).
[88] W. C. Chew and W. H. Weedon, Microwave Opt. Technol. Lett. **7**, 599 (1994).
[89] M. König, C. Prohm, K. Busch, and J. Niegemann, Opt. Express **19**, 4618–4631 (2011).
[90] H. Fahs, Appl. Math. Sci. (Ruse) **4**, 943 (2010).
[91] L.-B. Zhang, T. Cui, and H. Liu, J. Comput. Math. **27**, 89–96 (2009).
[92] T. Warburton, URSI International Symposium on Electromagnetic Theory, Berlin, Germany (IEEE, 2010), 996–999.
[93] J. Niegemann, M. König, C. Prohm, R. Diehl, and K. Busch, AIP Conf. Proc. **1291**, 76–78 (2010).
[94] J. W. Cooley and J. W. Tukey, Math. Comp. **19**, 297–301 (1965).
[95] M. Frigo and S. G. Johnson, Proc. IEEE **93**, 216–231 (2005).
[96] A. Dutt and V. Rokhlin, SIAM J. Sci. Comp. **14**, 1368–1393 (1993).
[97] V. A. Mandelshtam and H. S. Taylor, J. Chem. Phys. **107**, 6756 (1997).
[98] S. G. Johnson, Harminv, http://ab-initio.mit.edu/wiki/index.php/Harminv.
[99] M. Husnik, M. W. Klein, N. Feth, M. König, J. Niegemann, K. Busch, S. Linden, and M. Wegener, Nature Photon. **2**, 614 (2008).