DGTD methods using modal basis functions and symplectic local time-stepping: application to wave propagation problems

Serge Piperno

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

DGTD methods using modal basis functions and symplectic local time-stepping: application to wave propagation problems.

Serge Piperno

N° 5749

Novembre 2005

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--5749--FR+ENG

INRIA
SOPHIA ANTIPOLIS

DGTD methods using modal basis functions and symplectic local time-stepping: application to wave propagation problems.

Serge Piperno

Thème NUM — Systèmes numériques
Projet Caiman

Rapport de recherche n° 5749 — Novembre 2005 — 31 pages

Abstract: The Discontinuous Galerkin Time Domain (DGTD) methods are now widely used for the solution of wave propagation problems. Able to deal with unstructured meshes past complex geometries, they remain fully explicit with easy parallelization and extension to high orders of accuracy. Still, modal or nodal local basis functions have to be chosen carefully to obtain actual numerical accuracy. Concerning time discretization, explicit non-dissipative energy-preserving time-schemes exist, but their stability limit remains linked to the smallest element size in the mesh. Symplectic algorithms, based on local-time stepping or local implicit scheme formulations, can lead to dramatic reductions of computational time, which is shown here on two-dimensional acoustics problems.

Key-words: waves, acoustics, Maxwell's system, Discontinuous Galerkin methods, mass matrix condition number, symplectic schemes, energy conservation, local time-stepping.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Méthodes DGTD avec fonctions de base modales et schémas en temps symplectiques : applications en propagation d'ondes.

Résumé : Les méthodes de Galerkine discontinu sont maintenant largement utilisées pour la résolution numérique de problèmes de propagation d'ondes. S'appuyant sur des maillages non-structurés autour des géométries les plus générales, elles restent quasiment complètement explicites, facilement parallélisables et d'ordre élevé. Il convient néanmoins d'optimiser le choix des fonctions de base discontinues (modales ou nodales). Pour ce qui est de la discrétisation en temps, des schémas explicites non-dissipatifs existent, mais leur limite de stabilité reste liée aux plus petits éléments du maillage. Des algorithmes symplectiques, avec pas de temps local ou schéma localement implicite, conduisent à des diminutions considérables du temps de calcul.

Mots-clés : ondes, acoustique, système de Maxwell, Galerkine discontinu, conditionnement, schémas symplectiques, conservation de l'énergie, pas de temps local.

1 Introduction

The accurate modeling of systems involving electromagnetic, acoustic, or elastic waves, in particular through the time-domain numerical solution of wave equations on space grids, remains of strategic interest for many technologies. The still prominent, explicit, energy-conserving Finite Difference Time-Domain (FDTD) method [29] lacks two important features to be fully applied in industrial design chains: 1) the huge restriction to structured or block-structured grids, and 2) the efficiency of FDTD methods is limited when fully curvilinear coordinates are used.

Many different types of methods have been proposed in order to handle complex geometries and heterogeneous configurations by dealing with unstructured tetrahedral meshes, including, for example, mass lumped Finite Element Time-Domain (FETD) methods [5, 17], mimetic methods [16], or Finite Volume Time-Domain (FVTD) methods [26, 3, 23], which all fail in being at the same time as efficient as explicit methods, easily extendible to high orders of accuracy, and provably stable. The global conservation of the electromagnetic energy and the preservation of divergence, which are two desirable properties of Yee's original method, have been also obtained for FETD methods or for FVTD methods based on totally centered numerical fluxes [23], coupled with a centered implicit time-scheme or an explicit leap-frog time-scheme.

The Discontinuous Galerkin methods enjoy an impressive favor nowadays and are now used in many and various applications [4], taking advantage of their ability to achieve a high order of accuracy by simply choosing suitable basis functions (spectral elements [18], Lagrange high-order polynomials on tetrahedra [9, 10, 11]) or to handle complicated geometries and meshes (including locally-refined [2] and non-conformal grids [28]). The existing software, based on Discontinuous Galerkin Time-Domain (DGTD) methods implement in most cases upwind fluxes and multi-step low-storage Runge-Kutta time-schemes [3, 27, 18, 11], which make them robust, stable, all-purpose, but slightly dissipative when applied to wave propagation problems. However, centered fluxes coupled with an explicit leap-frog time-scheme lead to a convergent, stable, and energy-conserving DGTD method [6], for which the time-integration remains a concern for locally refined grids like those obtained by automatic mesh generators round configurations involving small devices or details in the geometry.

It has been shown [20] that symplectic time-schemes, originally developed for the numerical time integration of dynamical Hamiltonian systems – astronomy, molecular dynamics, etc – [25] and currently being used for the time-integration of spatially-discretized wave propagation problems [12, 13, 24] can overcome this problem, via locally implicit time-integration or explicit local time-stepping.

We consider in this paper the application of some particular symplectic schemes to the finite-dimensional system obtained after space-discretization using a Discontinuous Galerkin method based on totally centered fluxes, with a particular attention to configurations where different scales in the grid are present. In Section 2, we recall the basic features of Discontinuous Galerkin space-discretizations of first-order 3D Maxwell's equations or first-order 2D acoustics equations in the time domain, based on totally centered numerical fluxes. In Section 4, we quickly discuss the advantages and drawbacks of several choices of local basis

functions in the context of Discontinuous Galerkin methods. We propose an original choice for local basis functions in simplices in any space dimension, which have a whole set of numerical properties. In Section 4.2, we quickly recall two symplectic approaches in the particular context of DGTD methods for Maxwell's equations or acoustics equations, one is explicit with recursive local time-stepping, the second one is locally implicit. Numerical results in two space dimensions are presented in Section 5 and conclusions and further research and development directions are proposed in Section 6.

2 DGTD methods for wave propagation problems

2.1 Discontinuous Galerkin discretization of Maxwell's system

We first consider in this section the Maxwell's equations in three space dimensions for heterogeneous anisotropic linear media with no source. The electric permittivity tensor $\bar{\bar{\varepsilon}}(x)$ and the magnetic permeability tensor $\bar{\bar{\mu}}(x)$ are varying in space and both symmetric positive definite (with uniform strictly positive lower and upper bounds). The electric field $\vec{E}$ and the magnetic field $\vec{H}$ verify

$$\bar{\bar{\varepsilon}}\partial_t \vec{E} = \vec{\mathrm{curl}}\, \vec{H}, \quad \bar{\bar{\mu}}\partial_t \vec{H} = -\vec{\mathrm{curl}}\, \vec{E}, \tag{1}$$

where the symbol ∂_t denotes a time derivative. These equations are set and solved on a bounded polyhedral domain Ω of $\mathbb{R}^3$. These equations have a particular form: the time derivative of the electric field $\vec{E}$ (resp. the magnetic field $\vec{H}$) only depends on the other field, i.e. $\vec{H}$ (resp. $\vec{E}$). This feature is also present in two-dimensional and three-dimensional acoustics equations, which are introduced in the next section.

For the sake of simplicity, a metallic boundary condition is set everywhere on the domain boundary $\partial\Omega$, i.e. $\vec{n}\times\vec{E} = \vec{0}$ (where $\vec{n}$ is the unitary outwards normal). We assume we dispose of a partition of a polyhedral domain Ω_h (approximating the regular or Lipschitz-continuous domain of interest Ω) into a finite number of polyhedra. For each polyhedral element $\mathcal{T}_i$, V_i denotes its volume, and $\bar{\bar{\varepsilon}}_i$ and $\bar{\bar{\mu}}_i$ are respectively the local electric permittivity and magnetic permeability tensors of the medium, which could be varying inside the element $\mathcal{T}_i$. We call face between two finite elements their intersection, whenever it is a polyhedral surface. We denote by $\mathcal{F}_h$ the union of faces and by $\mathcal{F}_h^{\mathrm{int}} = \mathcal{F}_h/\partial\Omega_h$ the union of internal faces (common to two finite elements). For each internal face $a_{ik} = \mathcal{T}_i \bigcap \mathcal{T}_k$, we denote by S_{ik} the measure of a_{ik} and by $\vec{n}_{ik}$ the unitary normal, oriented from $\mathcal{T}_i$ towards $\mathcal{T}_k$. The same definitions are extended to metallic boundary faces (in the intersection of the domain boundary $\partial\Omega_h$ with a finite element), the index k corresponding to a fictitious element outside the domain. Finally, we denote by $\mathcal{V}_i$ the set of indices of the neighboring elements of the $\mathcal{T}_i$ (having a face in common). We also define the perimeter P_i of $\mathcal{T}_i$ by $P_i = \sum_{k\in\mathcal{V}_i} S_{ik}$. We recall the following geometrical property for all elements: $\sum_{k\in\mathcal{V}_i} S_{ik}\vec{n}_{ik} = 0$.

Following the Discontinuous Galerkin approach, the electric and magnetic fields inside each finite element are seeked for as linear combinations $(\vec{\mathbf{E}}_i, \vec{\mathbf{H}}_i)$ of linearly independent basis vector fields $\vec{\varphi}_{ij}$, $1 \le j \le d_i$, where d_i denotes the local number of scalar degrees

of freedom inside $\mathcal{T}_i$. We denote by $\mathcal{P}_i = Span(\vec{\varphi}_{ij},\ 1 \leq j \leq d_i)$. The approximate fields $(\vec{\mathbf{E}}_h, \vec{\mathbf{H}}_h)$, defined by $(\forall i,\ \vec{\mathbf{E}}_{h|\mathcal{T}_i} = \vec{\mathbf{E}}_i,\ \vec{\mathbf{H}}_{h|\mathcal{T}_i} = \vec{\mathbf{H}}_i)$ are allowed to be completely discontinuous across element boundaries. The formulation is mathematically invariant if another local basis vector fields is chosen, the set $\mathcal{P}_i$ remaining constant. However, numerical accuracy and efficiency of the methods are related to the choice of basis functions, which will be discussed in Section 4.

Because of this complete discontinuity, a global variational formulation cannot be obtained. However, dot-multiplying (1) by any given vector field $\vec{\varphi} \in \mathcal{P}_i$, integrating over each single element $\mathcal{T}_i$ and integrating by parts, yields

$$\left\{ \begin{array}{l} \displaystyle\int_{\mathcal{T}_i} \vec{\varphi} \cdot \bar{\bar{\varepsilon}}_i \partial_t \vec{\mathbf{E}} = -\int_{\partial\mathcal{T}_i} \vec{\varphi} \cdot (\vec{\mathbf{H}} \times \vec{n}) + \int_{\mathcal{T}_i} \vec{\mathrm{curl}}\, \vec{\varphi} \cdot \vec{\mathbf{H}}, \\ \displaystyle\int_{\mathcal{T}_i} \vec{\varphi} \cdot \bar{\mu}_i \partial_t \vec{\mathbf{H}} = \int_{\partial\mathcal{T}_i} \vec{\varphi} \cdot (\vec{\mathbf{E}} \times \vec{n}) - \int_{\mathcal{T}_i} \vec{\mathrm{curl}}\, \vec{\varphi} \cdot \vec{\mathbf{E}}. \end{array} \right. \tag{2}$$

In equations (2), we now replace the exact fields $\vec{\mathbf{E}}$ and $\vec{\mathbf{H}}$ by the approximate fields $\vec{\mathbf{E}}_h$ and $\vec{\mathbf{H}}_h$ in order to evaluate volume integrals. For integrals over $\partial\mathcal{T}_i$, some additional approximations have to be done since the approximate fields are discontinuous through element faces. We choose to use completely centered fluxes, i.e. $\forall i, \forall k \in \mathcal{V}_i$, $\vec{\mathbf{E}}_{|a_{ik}} \simeq (\vec{\mathbf{E}}_i + \vec{\mathbf{E}}_k)/2$, $\vec{\mathbf{H}}_{|a_{ik}} \simeq (\vec{\mathbf{H}}_i + \vec{\mathbf{H}}_k)/2$. The metallic boundary condition on a boundary face a_{ik} (k in the element index of the fictitious neighboring element) is dealt with *weakly*, in the sense that traces of fictitious fields $\vec{\mathbf{E}}_k$ and $\vec{\mathbf{H}}_k$ are used for the computation of numerical fluxes for the boundary element $\mathcal{T}_i$. In the present case, where all boundaries are metallic, we simply take $\vec{\mathbf{E}}_{k|a_{ik}} = -\vec{\mathbf{E}}_{i|a_{ik}}$ and $\vec{\mathbf{H}}_{k|a_{ik}} = \vec{\mathbf{H}}_{i|a_{ik}}$. Replacing surface integrals using centered fluxes in (2) and re-integrating by parts yields the final Discontinuous Galerkin discretization of the Maxwell's system (see [6] for more details). In terms of scalar unknowns inside each element, the fields being recomposed according to $\vec{\mathbf{E}}_i = \sum_{1 \leq j \leq d_i} E_{ij}\ \vec{\varphi}_{ij}$, $\vec{\mathbf{H}}_i = \sum_{1 \leq j \leq d_i} H_{ij}\ \vec{\varphi}_{ij}$ and denoting by $\mathbf{E}_i$ and $\mathbf{H}_i$ respectively the columns $(E_{il})_{1 \leq l \leq d_i}$ and $(H_{il})_{1 \leq l \leq d_i}$, we get the general form:

$$\left\{ \begin{array}{l} M_i^\epsilon \partial_t \mathbf{E}_i = \quad K_i \mathbf{H}_i - \displaystyle\sum_{k \in \mathcal{V}_i} S_{ik} \mathbf{H}_k, \\ M_i^\mu \partial_t \mathbf{H}_i = -K_i \mathbf{E}_i + \displaystyle\sum_{k \in \mathcal{V}_i} S_{ik} \mathbf{E}_k, \end{array} \right. \tag{3}$$

where the positive definite symmetric mass matrices M_i^ϵ, M_i^μ, and the symmetric stiffness matrices K_i (all of size d_i) are given by: $(M_i^\epsilon)_{jl} = \int_{\mathcal{T}_i} {}^t\vec{\varphi}_{ij} \bar{\bar{\varepsilon}}_i \vec{\varphi}_{il}$, $(M_i^\mu)_{jl} = \int_{\mathcal{T}_i} {}^t\vec{\varphi}_{ij} \bar{\bar{\mu}}_i \vec{\varphi}_{il}$, $(K_i)_{jl} = \frac{1}{2} \int_{\mathcal{T}_i} \left({}^t\vec{\varphi}_{ij}\, \vec{\mathrm{curl}} \vec{\varphi}_{il} + {}^t\vec{\varphi}_{il}\, \vec{\mathrm{curl}} \vec{\varphi}_{ij} \right)$, and for any interface a_{ik}, the $d_i \times d_k$ rectangular matrix S_{ik} is given by $(S_{ik})_{jl} = \frac{1}{2} \int_{a_{ik}} \vec{\varphi}_{ij} \cdot (\vec{\varphi}_{kl} \times \vec{n}_{ik})$.

Finally, if all electric (resp. magnetic) unknowns are regrouped inside column vectors $\mathbb{E}$ (resp. $\mathbb{H}$) of size $d = \sum_i d_i$, then the space discretized system (3) can be rewritten as

$$\left\{ \begin{array}{l} \mathbb{M}^\epsilon \partial_t \mathbb{E} = \quad \mathbb{K}\mathbb{H} - \mathbb{A}\mathbb{H} - \mathbb{B}\mathbb{H}, \\ \mathbb{M}^\mu \partial_t \mathbb{H} = -\mathbb{K}\mathbb{E} + \mathbb{A}\mathbb{E} - \mathbb{B}\mathbb{E}, \end{array} \right.$$

where we have the following definitions and properties:

- $\mathbb{M}^\epsilon$, $\mathbb{M}^\mu$ and $\mathbb{K}$ are $d \times d$ block diagonal matrices with diagonal blocks equal to M_i^ϵ, M_i^μ, and K_i respectively. Therefore $\mathbb{M}^\epsilon$ and $\mathbb{M}^\mu$ are symmetric positive definite, and $\mathbb{K}$ is symmetric; one can recall that the matrices M_i^ϵ and M_i^μ being block diagonal, time integration with an explicit time-scheme leads to an almost completely explicit algorithm;
- $\mathbb{A}$ is also a $d \times d$ block sparse matrix, whose non-zero blocks are equal to S_{ik} when $k \in \mathcal{V}_i$ is not fictitious (a_{ik} then is an internal face of the grid). Since $\vec{n}_{ki} = -\vec{n}_{ik}$, it can be checked that $(S_{ik})_{jl} = (S_{ki})_{lj}$, and then $S_{ki} = {}^tS_{ik}$; then $\mathbb{A}$ is symmetric;
- $\mathbb{B}$ is a $d \times d$ block diagonal matrix, whose non-zero diagonal blocks are equal to S_{ik} when a_{ik} is a metallic boundary face of the grid. In that case, $(S_{ik})_{jl} = -(S_{ik})_{lj}$, and $S_{ik} = -{}^tS_{ik}$; then $\mathbb{B}$ is skew-symmetric (${}^t\mathbb{B} = -\mathbb{B}$).

One finally obtains that the Maxwell's equations, discretized using discontinuous Galerkin finite-elements with centered fluxes and arbitrary local accuracy and basis functions can be written, in function of the matrix $\mathbb{S} = \mathbb{K} - \mathbb{A} - \mathbb{B}$, in the form:

$$\begin{cases} \mathbb{M}^\epsilon \partial_t \mathbb{E} = \mathbb{S}\mathbb{H}, \\ \mathbb{M}^\mu \partial_t \mathbb{H} = -{}^t\mathbb{S}\mathbb{E}, \end{cases} \quad (\mathbb{M}^\epsilon,\ \mathbb{M}^\mu \text{ symmetric positive definite}). \tag{4}$$

After spatial discretization, we obtain a system of ordinary differential equations for which the quantity $\mathcal{E} \equiv \frac{1}{2}\left({}^t\mathbb{E}\mathbb{M}^\epsilon\mathbb{E} + {}^t\mathbb{H}\mathbb{M}^\mu\mathbb{H}\right)$ is exactly conserved for any solution of (4). This property will be preserved after symplectic time-discretization.

2.2 Discontinuous Galerkin discretization of acoustics

We now consider the equation of classical acoustics in two or three space dimensions with no source. Assuming isentropic perturbations of still air with uniform pressure p_0 and density ρ_0 (with $c_0^2 = \gamma p_0/\rho_0$), the velocity and pressure perturbations $\vec{u}$ and p verify $\rho_0\partial_t\vec{u} + \nabla p = 0$ and $\partial_t p + \rho_0 c_0^2 \nabla\cdot\vec{u} = 0$, which take the following symmetric form in terms of non-dimensional velocity $\vec{v} = \vec{u}/c_0$ and pressure $q = p/(\rho_0 c_0^2)$:

$$\partial_t\vec{v} + c_0\nabla q = 0, \quad \partial_t q + c_0\nabla\cdot\vec{v} = 0. \tag{5}$$

If s is the space dimension ($s \in \{1, 2, 3\}$), these equations are set and solved on a bounded polyhedral domain Ω of $\mathbb{R}^s$ and have the same "cross-over" form as Maxwell's system. For the sake of simplicity, a slip boundary condition is set on the whole domain boundary $\partial\Omega$, i.e. $\vec{n}.\vec{v} = 0$. All other geometrical settings are unchanged.

Following a simplified Discontinuous Galerkin approach, the velocity $\vec{v}$ and the pressure q inside elements are seeked for as combinations of the same scalar fields with respectively vectorial and scalar coefficients, i.e. d_i now denotes the local number of scalar basis fields $\varphi_{ij},\ 1 \le j \le d_i$ inside $\mathcal{T}_i$, and $\vec{v}$ and q are expressed as $\vec{v} = \sum_j \varphi_{ij}\vec{v}_{ij}$ and $q = \sum_j \varphi_{ij} q_{ij}$.

Again, the approximate fields are allowed to be completely discontinuous across element boundaries. The same process as previously is used: local variational formulation, centered numerical fluxes for interface integrals, and weak treatment of the slip boundary condition (i.e. on a boundary face a_{ik}, k being the index of the fictitious neighbour of element $\mathcal{T}_i$, we simply take $q_{k|a_{ik}} = q_{i|a_{ik}}$ for the pressure and $\vec{v}_{k|a_{ik}} = -\vec{v}_{i|a_{ik}}$ for the velocity, which seems to enforce a no slip condition $\vec{v} = 0$, but one must remember than only the normal component of $\vec{v}_k$ is used in the computation of the wall flux).

Denoting by $\mathbf{Q}_i$ and $\mathbf{V}_i$ respectively the column $(q_{il})_{1\leq l\leq d_i}$ and the matrix $(\vec{v}_{il})_{1\leq l\leq d_i}$, we get finally the general form:

$$\left\{\begin{array}{l} M_i\partial_t\mathbf{Q}_i + K_i\mathbf{V}_i + \displaystyle\sum_{k\in\mathcal{V}_i} S_{ik}\mathbf{V}_k = 0, \\ M_i\partial_t\mathbf{V}_i + K_i\mathbf{Q}_i + \displaystyle\sum_{k\in\mathcal{V}_i} S_{ik}\mathbf{Q}_k = 0. \end{array}\right. \tag{6}$$

where, by convention, 1) $\mathbf{V}_i$ is understood as a column vector with vectorial entries in $\mathbb{R}^s$; 2) K_i and S_{ik} are understood as matrices with transposed vector entries in $\mathbb{R}^s$; 3) the product $K_i\mathbf{V}_i$ is understood as a sum of matricial products of entries; and 4) the product $M_i\mathbf{V}_i$ is understood as a column of vectorial entries obtained by simple linear combination. Finally, we have the following definitions: the positive definite symmetric mass matrix M_i is given by $(M_i)_{jl} = \int_{\mathcal{T}_i}\varphi_{ij}\varphi_{il}$; the "skew-symmetric" stiffness matrix K_i is given by $(K_i)_{jl} = \frac{c_0}{2}\int_{\mathcal{T}_i}\left(\varphi_{ij}\,{}^t\!\nabla\varphi_{il} - \varphi_{il}\,{}^t\!\nabla\varphi_{ij}\right)$; for any interface a_{ik}, the $d_i\times d_k$ rectangular matrix S_{ik} (with transposed vector entries) is given by $(S_{ik})_{jl} = \frac{c_0}{2}\,{}^t\vec{n}_{ik}\int_{a_{ik}}\varphi_{ij}\varphi_{kl}$.

Finally, if all pressure (resp. velocity) unknowns are regrouped inside column vectors $\mathbb{Q}$ (resp. $\mathbb{V}$ with vectorial entries) of size $d = \sum_i d_i$, then the space discretized system (6) can be summed up, like for Maxwell's system, as

$$\left\{\begin{array}{l} \mathbb{M}\partial_t\mathbb{Q} + \mathbb{K}\mathbb{V} + \mathbb{A}\mathbb{V} - \mathbb{B}\mathbb{V} = 0, \\ \mathbb{M}\partial_t\mathbb{V} - {}^t\!\mathbb{K}\mathbb{Q} - {}^t\!\mathbb{A}\mathbb{Q} - {}^t\!\mathbb{B}\mathbb{Q} = 0, \end{array}\right.$$

where we have the following definitions and properties:

- $\mathbb{M}$ and $\mathbb{K}$ are $d\times d$ block diagonal matrices with diagonal blocks equal to M_i and K_i respectively (then $\mathbb{K}$ has vectorial entries). Therefore $\mathbb{M}$ is symmetric positive definite.
- $\mathbb{A}$ also is a $d\times d$ block sparse matrix with transposed vector entries, whose non-zero blocks are equal to S_{ik} when $k\in\mathcal{V}_i$ is not fictitious (a_{ik} then is an internal interface of the grid). Since $\vec{n}_{ki} = -\vec{n}_{ik}$, it can be checked that $S_{ki} = -{}^tS_{ik}$.
- $\mathbb{B}$ is a $d\times d$ block diagonal matrix with vectorial entries, whose non-zero diagonal blocks are equal to S_{ik} when a_{ik} is a metallic boundary face of the grid. In that case, $(S_{ik})_{jl} = (S_{ik})_{lj}$, and $S_{ik} = {}^tS_{ik}$.

One finally obtains that the equations of acoustics, discretized using discontinuous Galerkin finite-elements with centered fluxes and arbitrary local accuracy and basis functions can be

written, in function of the matrix $\mathbb{S} = -(\mathbb{K} + \mathbb{A} - \mathbb{B})$, in the form:

$$\left\{ \begin{array}{l} \mathbb{M}\partial_t \mathbb{Q} = \mathbb{S}\mathbb{V}, \\ \mathbb{M}\partial_t \mathbb{V} = -{}^t\mathbb{S}\mathbb{Q}. \end{array} \right. \tag{7}$$

We again obtain a system of ordinary differential equations (7) for which an energy, here $\mathcal{E} \equiv \frac{1}{2}\left({}^t\mathbb{Q}\mathbb{M}\mathbb{Q} + {}^t\mathbb{V}\mathbb{M}\mathbb{V}\right)$, is exactly conserved for any solution.

3 An original set of modal basis functions

In the previous section, we have proposed two Discontinuous Galerkin discretizations of PDEs over simplices. The context of this paper is quite particular, since the PDEs considered (Maxwell's system, homogeneous acoustics) are linear and with at least elementwise constant coefficients. If one is considering the construction of a more general DG-based software, the scope of application should also include non-linear PDEs (possibly other than hyperbolic systems of conservation laws), with elementwise varying coefficients, curvilinear elements, etc.

As recalled in the previous section, the mathematical properties of a Discontinuous Galerkin method does not depend on the particular choice of basis functions or basis vector fields. Indeed, the method only depends on the local vector spaces $\mathcal{P}_i = Span(\vec{\varphi}_{ij},\ 1 \leq j \leq d_i)$.

3.1 Criteria for the choice of local basis functions

When a real implementation is considered, all choices for basis fields are not equivalent, because the numerical computation of integrals present in matrix terms, and then the numerical solution of linear systems (or the inversion/factorization of local mass matrix) cannot be exact in general. Indeed, the condition number of the mass matrices as well as the quadrature formulae used for volume or interface integrals play an important role in the actual numerical accuracy and stability of the software developed [9]. The use of simplicial elements makes the construction of an optimal set of basis functions more difficult.

In the formulations seen above, several numerical operations must be performed which can be numerically difficult:

- the computation of volume integrals for the mass matrices, typically $\int_{\mathcal{T}_i} \varphi_{ij}\varphi_{il}$;
- the computation of volume integrals for the stiffness matrices, typically of the form $\int_{\mathcal{T}_i} \left({}^t\vec{\varphi}_{ij}\, \vec{\mathrm{curl}}\vec{\varphi}_{il} + {}^t\vec{\varphi}_{il}\, \vec{\mathrm{curl}}\vec{\varphi}_{ij} \right)$ or more simply $\int_{\mathcal{T}_i} (\varphi_{ij}\, \partial_x\varphi_{il} - \varphi_{il}\, \partial_x\varphi_{ij})$ involving derivatives (it should be noted that these formulae are quite different if a system of nonlinear PDEs is considered);
- the numerical solution of linear system requiring the inversion or the factorization of the mass matrices.

We now limit our context to linear PDEs with at least elementwise constant coefficients. Among many possible approaches, one can disitinguish nodal approaches from modal approaches.

Nodal approaches rely on Lagrange polynomials based on a set of nodes. The set of nodes is optimized such that 1) the subset of nodes on element faces yields efficient and accurate computation of surface integrals; 2) the nodes provide an efficient and accurate way to compute volume integrals; 3) the condition number of the mass matrix is controlled and the inverse of the mass matrix can be computed or factorized. The simultaneous optimization of all these criteria is very difficult and has produced an important literature, for example on spectral finite element methods (see [9] for a review).

Modal approaches do not rely on particular sets of nodes. The idea is only to chose an optimal family of functions generating $\mathcal{P}_i$. Criteria for optimizing the family is the efficiency to compute surface integrals (for $\mathbb{P}_k$ elements, only $k+1$ in 2D and $(k+1)(k+2)/2$ in 3D functions should be nonzero on an element face), the volume and interface integrals should be exactly known if possible, the condition number of the mass matrix should be minimal or the inverse of the mass matrix should be known.

We consider $\mathbb{P}_k$ functions in affine simplicial elements (polynomials of total degree at most k in coordinates). We propose an original set of scalar basis functions for a modal approaches with many interesting properties which seems to be quite optimal with respect to the criteria listed above. We first describe these modes in one space dimension and then derive expressions and two and three space dimensions.

3.2 A family of modal basis functions in 1D

We consider here scalar basis functions of $\mathbb{P}_k$ ($k \geq 0$) in the reference interval $(0;1)$ of $\mathbb{R}$. We propose the following polynomials π_i^k (for $0 \leq i \leq k$):

$$\pi_i^k(x) = c_{ki} x^i (1-x)^{k-i},$$

where the scalar coefficients c_{ki} are assumed positive and still to be defined. It is clear that the $k+1$ polynomials π_i^k are linearly independent and then that $\mathbb{P}_k = Span(\pi_i^k,\ 0 \leq i \leq k)$. Some elementary remarks can be done:

- the polynomials π_i^k are of degree k exactly;
- assuming the c_{ki} are positive, they are positive over $(0;1)$;
- since $\partial_x(\pi_i^k) = c_{ki} x^{i-1}(1-x)^{k-i-1}(i-kx)$, each polynomial π_i^k is increasing, has a maximum at $x = i/k$ and then is decreasing. The maxima of the π_i^k are then uniformly placed in $(0;1)$;
- for all $i \neq 0$, $\pi_i^k(0) = 0$ and for all $i \neq k$, $\pi_i^k(1) = 0$: then only one degree of freedom is necessary to compute the value of any polynomial at $x = 0$ or $x = 1$.

The coefficients c_{ki} are chosen such that the condition number of the mass matrix $\mathbb{M}^k$ is minimum. The entries in the mass matrix are derived from the generic expression $\int_0^1 x^\alpha (1-x)^\beta = \alpha!\beta!/(\alpha+\beta+1)!$. Then we have:

$$\mathbb{M}^k_{ij} \equiv \int_0^1 \pi^k_i \pi^k_j = c_{ki} c_{kj} (i+j)!(2k-i-j)!/(2k+1)!.$$

We then propose a conjecture, which is still to be proved, but has been numerically verified (up to order $k=5$).

Proposition 3.1 *The condition number of the mass matrix deriving from the basis functions π^k_i is minimal when $c_{ki} = \binom{k}{i}$ (where the $\binom{k}{i}$ are the binomial coefficients $\binom{k}{i} \equiv \frac{k!}{i!(k-i)!}$).*

In the remainder of this section, we then take

$$\pi^k_i(x) = \binom{k}{i} x^i (1-x)^{k-i}. \tag{8}$$

One can "understand" why this choice for the c_{ki} coefficients might be optimal. Indeed, they play a normalization role and reset the basis functions to "comparable levels". This is quite nicely done for this set of coefficients, since we have two renormalization properties at the same time. The positive functions add up to equal 1, and all basis functions π^k_i have the same L^1 norm over $(0;1)$, which is summed up as

$$a) \quad \sum_{i=0}^{k} \pi^k_i(x) = 1, \quad b) \quad \forall i, \int_0^1 \pi^k_i(x)\, dx = \frac{1}{k+1}. \tag{9}$$

Concerning the condition number, eigenvalues and eigenmodes of the mass matrix $\mathbb{M}^k$ (being symmetric, it is diagonalizable), we have the following properties (all of which have been numerically verified, many being still unproved):

Proposition 3.2 *The condition number $\#\mathbb{M}^k$ of the mass matrix deriving from the basis functions π^k_i is equal to $\binom{2k+1}{k}$. More precisely,*

- *(i) the eigenvalues μ^k_l $(0 \le l \le k)$ of $\mathbb{M}^k$ are simple and $\mu^k_l = \frac{k!k!}{l!(2k+1-l)!} = \frac{1}{k+1}\binom{2k+1}{l}/\binom{2k+1}{k}$. The eigenvalues are ordered: $\mu^k_k = \frac{1}{k+1} > \mu^k_{k-1} > \ldots > \mu^k_0$.*
- *(ii) for $0 \le l \le k$, an eigenvector $z = (z_i)_{0\le i\le k}$ corresponding to μ^k_{k-l} is given by $z_i = \int_0^1 \pi^k_i \mathbb{L}_l$, where $\mathbb{L}_l$ is the l^{th} Legendre polynomial (defined over $(0;1)$, i.e. $\mathbb{L}_0 = 1$, $\mathbb{L}_1 = 2x-1$, $\mathbb{L}_2 = 6x^2-6x+1$, $(l+1)\mathbb{L}_{l+1} = (2l+1)(2x-1)\mathbb{L}_l - l\mathbb{L}_{l-1}$).*
- *(iii) This implies the following property:*

$$\forall l,\ 0 \leq l \leq k,\ \frac{k!k!}{(k-l)!(k+1+l)!}\ \mathbb{L}_l(X) = \sum_{i=0}^{k} \left[\left(\int_0^1 \pi_i^k \mathbb{L}_l \right) \right] \pi_i^k(X). \tag{10}$$

Remark: Some parts of these results are easily shown. For example, the equation (9-b) yields the result for the first eigenvalue μ_0^k. The result for μ_1^k derives from the identity $\int_0^1 \pi_i^k \mathbb{L}_1 = \frac{2i-k}{(k+1)(k+2)}$.
As stated by the previous proposition, the condition number of the mass matrix grows quickly, but reasonably since $\#\mathbb{M}^7 = 6435$. It is also remarkable that the eigenvalues have such a simple expression. Indeed, the polynomials being with integer coefficients, the mass matrix entries are rational and the eigenvalues and the inverse of the mass matrix as well. Numerically, the condition number $\#\mathbb{M}^k$ is not so important anymore, since the inverse of the mass matrix can be computed and stored. More, it seems that the this inverse must be multiplied by $k(k+1)/2$, which is a "slowly growing" integer, to become integer itself, as shown in the examples in Annex A.

Finally, the entries in the stiffness matrix are exactly known, since

$$\int_0^1 \left(\pi_i^k\, \partial_x \pi_j^k - \pi_j^k\, \partial_x \pi_i^k \right)\ dx = (2k+1)\ \mathbb{M}_{ij}^k s_{ij} \text{ with } \left\{ \begin{array}{l} s_{ij} = 0 \text{ if } i = j, \text{else} \\ s_{ij} = \frac{2k(j-i)}{(j+i)(2k-i-j)}. \end{array} \right.$$

3.3 A family of modal basis functions in more space dimensions

The family of modal basis functions seen in one space dimension can be naturally extended to higher dimensions on simplices. Consider scalar functions of $\mathbb{P}_k$ ($k \geq 0$) in the reference simplex $\mathcal{T}_s = \{x = (x_1, \ldots, x_s) \in (\mathbb{R}^+)^s,\ \sum_{i=1}^s x_s \leq 1\}$ of $\mathbb{R}^s$. Each point x in the simplex can be located through its $s+1$ barycentric coordinates with respect to the $s+1$ vertices of the simplex. There is an affine bijective map between the simplex and the set of $s+1$-barycentric coordinates of points in the simplex, equal to $\{\lambda = (\lambda_1, \ldots, \lambda_s, \lambda_{s+1}) \in (\mathbb{R}^+)^{s+1};\ \sum_{i=1}^{s+1} \lambda_i = 1\}$. Let us introduce the notion of a $s+1$-multiindex: $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{s+1})$ is a $s+1$-multiindex (let us say $\boldsymbol{\alpha} \in I_{s+1}$) if the α_i are non-negative integers. We define $|\boldsymbol{\alpha}|_{s+1} = \sum_{i=1}^{s+1} \alpha_i$ and we denote by I_{s+1}^k the multi-indices $\boldsymbol{\alpha}$ of I_{s+1} such that $|\boldsymbol{\alpha}|_{s+1} = k$. We propose the following polynomials $\pi_{\boldsymbol{\alpha}}^k$, $\boldsymbol{\alpha} \in I_{s+1}^k$ by:

$$\pi_{\boldsymbol{\alpha}}^k(x) = c_{k\boldsymbol{\alpha}} \prod_{i=1}^{s+1} \lambda_i^{\alpha_i},$$

where the scalar coefficients $c_{k\boldsymbol{\alpha}}$ are assumed positive and still to be defined. The $\binom{k+s}{s}$ polynomials $\pi_{\boldsymbol{\alpha}}^k$ are linearly independent and $\mathbb{P}_k = Span(\pi_{\boldsymbol{\alpha}}^k,\ \boldsymbol{\alpha} \in I_{s+1},\ |\boldsymbol{\alpha}|_{s+1} = k)$. Some elementary remarks can be done:

- the polynomials $\pi_{\boldsymbol{\alpha}}^k$ are of total degree k exactly in the barycentric coordinates and in the geometric coordinates;
- if $c_{k\boldsymbol{\alpha}} > 0$, then $\pi_{\boldsymbol{\alpha}}^k \geq 0$ over the reference simplex $\mathcal{T}_s$;

- the extrema of $\pi^k_{\boldsymbol{\alpha}}$ are reached when $\partial_{\lambda_j}(\pi^k_{\boldsymbol{\alpha}}) = 0$ for all $j > s$ (with λ_{s+1} expressed has $\lambda_{s+1} = 1 - \sum_{i=1}^{s} \lambda_i$). Simple calculus shows that the extrema are obtained at points where $\lambda_j = \frac{\alpha_j}{k}$, which are uniformly placed in the simplex.
- considering a given face of the simplex, let us say defined by $\lambda_j = 0$, then all $\pi^k_{\boldsymbol{\alpha}}$ but $\binom{k+s-1}{s-1}$ vanish on that face. The $\binom{k+s-1}{s-1}$ polynomials are (up to renumbering) the same polynomials as those defined for $\mathbb{P}_k$ polynomials on the simplex $\mathcal{T}_{s-1}$. Then the minimal number of degrees of freedom is necessary to compute the value of any polynomial or integral on a given face of the simplex $\mathcal{T}_s$.

The coefficients $c_{k\boldsymbol{\alpha}}$ are chosen such that the condition number of the mass matrix $\mathbb{M}^k$ is minimum. The entries in the mass matrix are derived from the generic expression $\int_{\mathcal{T}_s} \prod_{i=1}^{s+1} \lambda_i^{\alpha_i} = \frac{\prod_{i=1}^{s+1} \alpha_i!}{(s+\sum_{i=1}^{s+1} \alpha_i)!}$. Then we have:

$$\mathbb{M}^k_{\boldsymbol{\alpha\beta}} \equiv \int_{\mathcal{T}_s} \pi^k_{\boldsymbol{\alpha}} \pi^k_{\boldsymbol{\beta}} = c_{k\boldsymbol{\alpha}} c_{k\boldsymbol{\beta}} \frac{\prod_{i=1}^{s+1} (\alpha_i + \beta_i)!}{(2k+s)!}.$$

We have a conjecture in s space dimensions corresponding to the one in one space dimension (it is still to be proved and has been numerically verified up to order $k = 5$ in 2D).

Proposition 3.3 *The condition number of the mass matrix deriving from the basis functions $\pi^k_{\boldsymbol{\alpha}}$ is minimal when $c_{k\boldsymbol{\alpha}} = \binom{k}{\boldsymbol{\alpha}}$ (where the $\binom{k}{\boldsymbol{\alpha}}$ are the generalized "binomial" coefficients $\binom{k}{\boldsymbol{\alpha}} \equiv \frac{k!}{\prod_{i=1}^{s+1} \alpha_i!}$).*

In the remainder of this section, we then take

$$\pi^k_{\boldsymbol{\alpha}}(x) = \binom{k}{\boldsymbol{\alpha}} \prod_{i=1}^{s+1} \lambda_i^{\alpha_i}. \tag{11}$$

As in one space dimension, this choice for the $c_{k\boldsymbol{\alpha}}$ coefficients leads to two renormalization properties:

$$a) \quad \sum_{\boldsymbol{\alpha} \in I^k_{s+1}} \pi^k_{\boldsymbol{\alpha}} = 1, \quad b) \quad \forall \boldsymbol{\alpha} \in I^k_{s+1}, \int_{\mathcal{T}_s} \pi^k_{\boldsymbol{\alpha}}(x)\, dx = \frac{k!}{(k+s)!}. \tag{12}$$

Concerning the condition number, eigenvalues and eigenmodes of the mass matrix $\mathbb{M}^k$ (being symmetric, it is diagonalizable), we have the following properties (all of which have been numerically verified, many being still unproved):

Proposition 3.4 *The condition number $\#\mathbb{M}^k$ of the mass matrix deriving from the basis functions $\pi^k_{\boldsymbol{\alpha}}$ is equal to $\binom{2k+s}{k}$. More precisely, there are $k+1$ different eigenvalues for $\mathbb{M}^k$, denoted by μ^k_l ($0 \le l \le k$). μ^k_l is multiple (unless $l = 0$ or $s = 1$) of multiplicity $\binom{l+s-1}{l}$ and $\mu^k_l = \frac{k!k!}{l!(2k+s-l)!} = \frac{k!}{(k+s)!}\binom{2k+s}{l}/\binom{2k+s}{k}$. The eigenvalues are ordered: $\mu^k_k = \frac{k!}{(k+s)!} > \mu^k_{k-1} > \ldots$*

It seems more difficult to find the eigenvectors in more than one space dimensions. However, one easily finds that 1 is the function reconstructed for the eigenvector μ_k^k, that $\mathbb{L}_1(\lambda_j)$ are $s+1$ eigenvectors for the μ_{k-1}^k, but only s are independent...

For practical applications, it is valuable to give the actual condition numbers $\#\mathbb{M}^k$ for low orders k, in one, two, and three space dimensions. They are given in Table 1 and seem to grow in any space dimension like in one space dimension (not far from $2^s 4^k$ instead of a more common 4^{ks}). Anyway, it is remarkable that the eigenvalues and the condition number

Order k	0	1	2	3	4	5	6	7
$\#\mathbb{M}^k$ 1D	1	3	10	35	126	462	1716	6435
$\#\mathbb{M}^k$ 2D	1	4	15	56	210	792	3003	11440
$\#\mathbb{M}^k$ 3D	1	5	21	84	330	1287	5005	19448

Table 1: Condition numbers of the mass matrix for polynomials $\pi_{\boldsymbol{\alpha}}^k$ (1D, 2D, and 3D).

of the mass matrix have such a simple expression in any space dimension. Also, as in one space dimension, the polynomials being with integer coefficients, the mass matrix entries are rational and its inverse as well. Numerically, the condition number $\#\mathbb{M}^k$ is not so important anymore, since the inverse of the mass matrix can be computed and stored. More, like in one space dimension, it seems that the this inverse must be multiplied by a "slowly growing" integer to become integer (k is enough in two and three space dimensions up to order $k=5$), as shown in the examples 2D in Annex B. Finally, the entries in the stiffness matrix can again be exactly computed.

3.4 An actual implementation in 2D

One particular delicate task to do, when actually implementing these basis functions, is to choose a numbering of the set I_{s+1}^k (the multi-indices $\boldsymbol{\alpha}$ of I_{s+1} are such that $|\boldsymbol{\alpha}|_{s+1} = k$). Do we need to recall that this numbering has to be automatic? For the sake of simplicity, we shall consider here the two-dimensional case only.

At the same time, when computing interface integrals (here edge integrals), a problem appears: in the two neighbouring simplices (here triangles), local vertex indices (1, 2, or 3) are not necessarily the same! Then vertices do not correspond and basis functions must be identified. A solution consists in storing, for each edge the permutation which leads to a coinciding numbering of vertices in the two neighbouring triangles. This can be done by storing only two integers per edge. Consequently, some deriving permutation on basis functions $\pi_{\boldsymbol{\alpha}}^k$ must also be recovered. This permutation can be precomputed once and for all. It is global and not local. It gives, gor a given permutation of vertices in a triangle, what is the induced permutation on functions $\pi_{\boldsymbol{\alpha}}^k$. This indeed assumes that the 6 vertex permutations are also numbered (it is easy), and we recall that the basis functions have already been numbered.

For instance, we have chosen to number the basis functions in a reverse lexicographic order (in function of the decreasing vertex numbers in barycentric coordinates). This yields

for example for $\mathbb{P}_3$ on a triangle:

$$\begin{array}{ll}
\boldsymbol{\alpha}[1] = (3,0,0) & \pi^k_{[1]} = \lambda_1^3 \\
\boldsymbol{\alpha}[2] = (2,1,0) & \pi^k_{[2]} = 3\lambda_1^2\lambda_2 \\
\boldsymbol{\alpha}[3] = (1,2,0) & \pi^k_{[3]} = 3\lambda_1\lambda_2^2 \\
\boldsymbol{\alpha}[4] = (0,3,0) & \pi^k_{[4]} = \lambda_2^3 \\
\boldsymbol{\alpha}[5] = (2,0,1) & \pi^k_{[5]} = 3\lambda_1^2\lambda_3 \\
\boldsymbol{\alpha}[6] = (1,1,1) & \pi^k_{[6]} = 6\lambda_1\lambda_2\lambda_3 \\
\boldsymbol{\alpha}[7] = (0,2,1) & \pi^k_{[7]} = 3\lambda_2^2\lambda_3 \\
\boldsymbol{\alpha}[8] = (1,0,2) & \pi^k_{[8]} = 3\lambda_1\lambda_3^2 \\
\boldsymbol{\alpha}[9] = (0,1,2) & \pi^k_{[9]} = 3\lambda_2\lambda_3^2 \\
\boldsymbol{\alpha}[10] = (0,0,3) & \pi^k_{[10]} = \lambda_3^3.
\end{array}$$

This ordering yields a little simplification: in order to compute edge integrals, once the vertices are locally renumbered, the degrees of freeom involved are the first ones, i.e. after permutations, the "third" vertex is not on the edge considered, then $\lambda_3 = 0$ on the edge and the first four degrees of freedom only are needed. More examples are given in Annex B.

4 Symplectic schemes applied to wave propagation problems

4.1 Symplectic schemes for Hamiltonian systems

Symplectic integrators include a variety of different time-discretization schemes designed to preserve the global symplectic structure of the phase space for a Hamiltonian system. These integrators are well established for finite-dimensional Hamiltonian systems (see [19] for several references), most applications being devoted to N-body mechanical systems. However, the number of applications of symplectic schemes in the context of computational electromagnetics is currently growing [13, 24]. The electromagnetics (or acoustics) equations are first discretized, then the finite-dimensional system of ODEs obtained is considered as an input for symplectic methods. However, in some cases only, the discretization of Maxwell's equations actually leads to a Hamiltonian system of ODEs: it is indeed the case for some FDTD methods [13], more generally for FETD methods [24], and, also for the case considered here: DGTD methods with totally centered numerical fluxes.

One particular feature of symplectic schemes is their ability to reach high accuracy and to deal with local time-stepping. This is particularly needed for N-body mechanical systems for instance, where fixed stepsize numerical integration leads to difficulties when particles are very close. In this context, the leapfrog scheme is often replaced by the equivalent Verlet method which serves as a basis for further enhancements. The time-integration of (4) for

instance would then take the form:

$$\left\{ \begin{array}{l} \mathbb{M}^{\mu}\mathbb{H}^{n+\frac{1}{2}} = \mathbb{M}^{\mu}\mathbb{H}^{n} - \Delta t/2\, {}^{t}\mathbb{S}\mathbb{E}^{n}, \\ \mathbb{M}^{\epsilon}\mathbb{E}^{n+1} = \mathbb{M}^{\epsilon}\mathbb{E}^{n} + \Delta t \mathbb{S}\mathbb{H}^{n+\frac{1}{2}}, \\ \mathbb{M}^{\mu}\mathbb{H}^{n+1} = \mathbb{M}^{\mu}\mathbb{H}^{n+\frac{1}{2}} - \Delta t/2\, {}^{t}\mathbb{S}\mathbb{E}^{n+1}. \end{array} \right. \tag{13}$$

The classical leapfrog writing leads to an equivalent, cheaper two-step algorithm. The Verlet writing allows for the computations of fields at the same time stations. Moreover, the reversible writing leads to many quite easy enhancements in the scheme (varying time-step [15], high-order accurate extensions [14], etc). Last but not least, two strategies have been proposed in the context of waves propagation [20] where the local refinement of the unstructured grid requires the use of locally-implicit schemes or of local time-stepping. To our knowledge, the first one is still to be found. The second one, i.e. the construction of a totally explicit, stable, energy-conserving, second-order accurate algorithm with local time-stepping, has been seeked for without total success [7, 1, 21]. These two strategies are quickly presented in the next section.

4.2 DGTD methods based on symplectic schemes

4.2.1 A locally-implicit symplectic scheme

We consider a case where the set of elements has been partitioned (once and for all) into two classes with no particular assumption on the connectivity of the classes: one made of particularly small elements and the other one gathering all other elements. The "small" elements will be handled using an implicit midpoint rule, while all other elements will be time-advanced using a Verlet method.

As an illustration, we assume we solve Maxwell's equations. Using notations inspired from domain decomposition algorithms, we denote with an "e" (resp. "i") subscript unknowns and matrices related to the explicit (resp. implicit) subdomain. Unknowns are reordered such that explicit elements and unknowns are numbered first:

$$\mathbb{E} = \left(\begin{array}{c} \mathbb{E}_e \\ \mathbb{E}_i \end{array} \right), \ \mathbb{H} = \left(\begin{array}{c} \mathbb{H}_e \\ \mathbb{H}_i \end{array} \right).$$

The system of ordinary differential equations (4) can be rewritten [22] as

$$\left\{ \begin{array}{l} \mathbb{M}^{\epsilon}_{e}\partial_t \mathbb{E}_e = \mathbb{S}_e \mathbb{H}_e - \mathbb{A}_{ei}\mathbb{H}_i, \\ \mathbb{M}^{\mu}_{e}\partial_t \mathbb{H}_e = -{}^{t}\mathbb{S}_e \mathbb{E}_e + \mathbb{A}_{ei}\mathbb{E}_i, \end{array} \right.$$

$$\left\{ \begin{array}{l} \mathbb{M}^{\epsilon}_{i}\partial_t \mathbb{E}_i = \mathbb{S}_i \mathbb{H}_i - \mathbb{A}_{ie}\mathbb{H}_e, \\ \mathbb{M}^{\mu}_{i}\partial_t \mathbb{H}_i = -{}^{t}\mathbb{S}_i \mathbb{E}_i + \mathbb{A}_{ie}\mathbb{E}_e, \end{array} \right.$$

where $\mathbb{A}_{ie} = {}^{t}\mathbb{A}_{ei}$ and $\mathbb{M}^{\mu \text{ or } \epsilon}_{e \text{ or } i}$ are symmetric positive definite matrices. We propose the following implicit-explicit algorithm: starting from unknowns at time $t^n = n\Delta t$: 1) advance of $\Delta t/2$ the explicit domain with a pseudo-forward-Euler scheme; 2) advance of Δt the

implicit domain with the implicit midpoint rule; 3) advance of $\Delta t/2$ the explicit domain again with the time-reversed pseudo-forward-Euler scheme. The whole algorithm reads:

$$\left\{\begin{array}{l} \mathbb{M}_e^{\mu}\mathbb{H}_e^{n+\frac{1}{2}} = \mathbb{M}_e^{\mu}\mathbb{H}_e^{n} + \Delta t/2\left(-{}^t\mathbb{S}_e\mathbb{E}_e^{n} + \mathbb{A}_{ei}\mathbb{E}_i^{n}\right), \\ \mathbb{M}_e^{\epsilon}\mathbb{E}_e^{n+\frac{1}{2}} = \mathbb{M}_e^{\epsilon}\mathbb{E}_e^{n} + \Delta t/2\left(\mathbb{S}_e\mathbb{H}_e^{n+\frac{1}{2}} - \mathbb{A}_{ei}\mathbb{H}_i^{n}\right), \end{array}\right.$$
$$\left\{\begin{array}{l} \mathbb{M}_i^{\epsilon}\mathbb{E}_i^{n+1} = \mathbb{M}_i^{\epsilon}\mathbb{E}_i^{n} + \Delta t\left(\mathbb{S}_i\frac{\mathbb{H}_i^{n}+\mathbb{H}_i^{n+1}}{2} - \mathbb{A}_{ie}\mathbb{H}_e^{n+\frac{1}{2}}\right), \\ \mathbb{M}_i^{\mu}\mathbb{H}_i^{n+1} = \mathbb{M}_i^{\mu}\mathbb{H}_i^{n} + \Delta t\left(-{}^t\mathbb{S}_i\frac{\mathbb{E}_i^{n}+\mathbb{E}_i^{n+1}}{2} + \mathbb{A}_{ie}\mathbb{E}_e^{n+\frac{1}{2}}\right), \end{array}\right. \quad (14)$$
$$\left\{\begin{array}{l} \mathbb{M}_e^{\epsilon}\mathbb{E}_e^{n+1} = \mathbb{M}_e^{\epsilon}\mathbb{E}_e^{n+\frac{1}{2}} + \Delta t/2\left(\mathbb{S}_e\mathbb{H}_e^{n+\frac{1}{2}} - \mathbb{A}_{ei}\mathbb{H}_i^{n+1}\right), \\ \mathbb{M}_e^{\mu}\mathbb{H}_e^{n+1} = \mathbb{M}_e^{\mu}\mathbb{H}_e^{n+\frac{1}{2}} + \Delta t/2\left(-{}^t\mathbb{S}_e\mathbb{E}_e^{n+1} + \mathbb{A}_{ei}\mathbb{E}_i^{n+1}\right). \end{array}\right.$$

This algorithm is obviously reversible. One can verify that, if the two subdomains are disconnected (i.e. $\mathbb{A}_{ei} = \mathbb{O}_d$), this algorithm reduces to the juxtaposition of the Verlet-method for the "explicit" subdomain and the midpoint-rule for the "implicit" subdomain. The stability (at least for small time steps) can be shown [22] using an energy approach : the scheme exactly preserves an energy (a quadratic form of numerical unknowns $\mathbb{E}_e^n$, $\mathbb{E}_i^n$, $\mathbb{H}_e^n$, and $\mathbb{H}_i^n$ is exactly conserved) and this quadratic form is positive definite at least for small time steps.

4.2.2 A multi-scale fully-explicit symplectic scheme

The fully explicit algorithm recalled in this section is directly inspired from the one introduced by Hardy et al. [8] for N-body problems with multiple time stepping, i.e. the atoms or bodies are time-advanced simultaneously with different time steps. We present here a less general version, with time steps given as $\Delta t/2^k$ where Δt is the global time step of the algorithm. We assume that the set of elements has been partitioned into N classes and that, for $1 \leq k \leq N$, elements in the class k will be time-advanced using the Verlet method with the local time step $\Delta t/2^{N-k}$: thus the larger elements lie in class N (and are time-advanced with a local time step Δt) and the smallest lie in class 1.

The algorithm can be built recursively. Let us denote by $R^N(\tau)$ the algorithm for advancing in time N classes over the time interval $\tau > 0$. We decide that the algorithm $R^1(\tau)$ with only one class is exactly the Verlet method (13) with $\Delta t = \tau$. For any $N \geq 1$, if $R^N(\tau)$ is well defined, we define $R^{N+1}(\tau)$ by:

1. start with all unknowns at time $t^n = n\Delta t$;
2. advance all elements with class $k \leq N$ with $R^N(\Delta t/2)$; if required, use values at time t^n for unknowns in elements of class $N+1$;
3. advance all elements with class $k = N+1$ with the Verlet method (i.e. $R^1(\Delta t)$); if required, use values at time $t^n + \Delta t/2$ for unknowns in elements of class $k \leq N$;
4. advance all elements with class $k \leq N$ with $R^N(\Delta t/2)$; if required, use values at time t^{n+1} for unknowns in elements of class $N+1$;

5. all unknowns at time $t^{n+1} = t^n + \Delta t$ have been computed.

The reader can check that this algorithm does not required any additional storage and remains completely explicit. It is reversible, symplectic, second-order accurate, stable for small time-steps and conserves an energy [22]. The algorithm $R^2(\Delta t)$ and $R^3(\Delta t)$ are respectively sketched in Figure 1 and Figure 2.

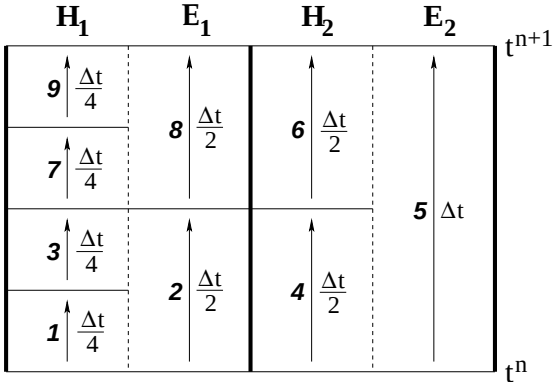

Figure 1: Algorithm $R^2(\Delta t)$: the nine sub-steps are detailed from **1** to **9**.

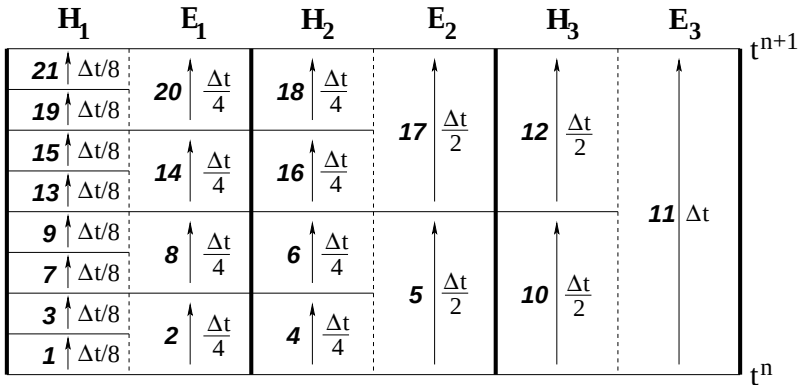

Figure 2: Algorithm $R^3(\Delta t)$: the twenty-one sub-steps are detailed from **1** to **21**.

5 Numerical results

We consider here the acoustics equations in two space dimensions (with $c_0 = 1$). We give illustrations of the explicit local time-stepping algorithm of Section 4.2.2 and of the locally implicit algorithm of Section 4.2.1 in the following two sections.

5.1 An illustration of the explicit local time-stepping algorithm

We have imagined a toy problem where the propagation of acoustic waves is confined in a completely reflecting cavity. In order to have different scales in the geometry, the cavity has be designed the following way:

- the cavity is an ellipse ($2m \times 1.6m$);
- inside the cavity, a small perfectly reflecting inclusion is located on the right focus; the device is a circular array of 0.2mm square, set at a distance equal to 1mm from the focus;
- the initial condition is a p/H_z pulse (for the acoustic equations) located at the other focus, such that the solution should refocus exactly on the other focus and scatter on the detail.

The unstructured mesh produced by a commercial mesh generator contains 1176 vertices and 2254 elements. The mesh partitioning leads to eleven classes of elements, i.e. the smallest elements are time-advanced 1024 times more often than the largest elements. A zoom of the mesh near the square is shown on Figure 3. Contours of the fields obtained with the algorithm $R^{11}(2.6ms)$ are shown on Figure 4. We have used in this section the $\mathbb{P}_5$-DGTD (the fields are described with polynomials of degree at most 5 inside elements). The CPU times obtained with the different time schemes considered are given on Table 2. For

Algorithm	$R^{11}(2.6ms)$	$R^{1}(3.54\mu s)$	leapfrog ($3.54\mu s$)
CPU time	7958	58212	38808
Gain (vs. leap-frog)	4.88	0.67	1

Table 2: Comparison of CPU times and gain between algorithms R^{11}, R^{1}, and a classical leapfrog implementation.

this particular case, the computational time is reduced by a factor near 5. This reduction factor is due to the fact that 80% of elements are time-advanced at most 4 times per global time-step, but 11% of elements are time-advanced at most 512 times per global time-step (the refined zone of the mesh is quite large). In some other more adhoc cases, (with a very limited, highly refined zone), CPU accelerations up to 40 have been observed.

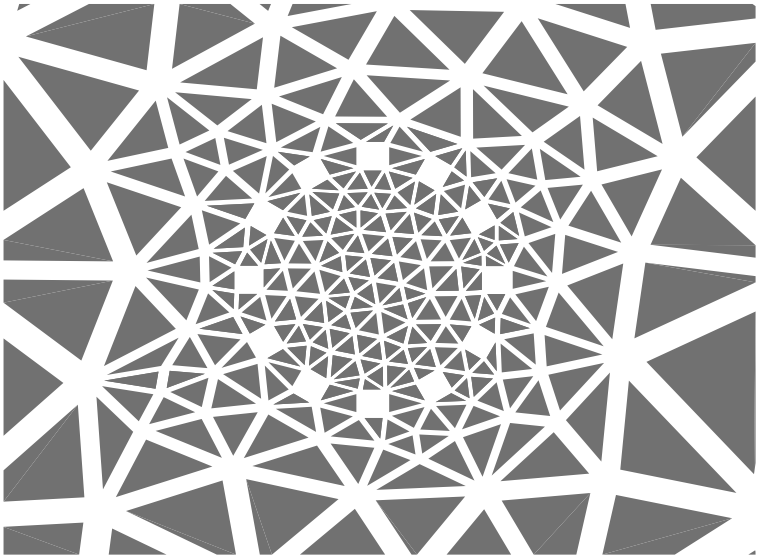

Figure 3: Unstructured triangular mesh near the circular array.

5.2 An illustration of the locally implicit algorithm

We give here an illustration for the locally implicit algorithm used jointly with a local time-stepping algorithm. The global procedure is the following. We assume we dispose of 1) a table giving the minimum admissible number n_k of points per wavelength for each $\mathbb{P}_k$-DGTD method, 2) a maximum Courant number ν_k leading to the stability of the method obtained with the simple Verlet method and $\mathbb{P}_k$-DGTD elements.

We know assume the user is able to set, for his computation:

- a global "wavelength" λ_{TC} of his problem (or a minimal characteristic scale);
- extremal ((k_{min}, k_{max}) values for the degree k of polynomials inside elements;
- a maximal number a different time-classes used in the recursive Verlet method;
- a maximal storage size for implicit matrix LU factorizations.

Then we propose the following process:

1. for all elements, define the local order k_i as the smallest integer k ($k_{min} \leq k \leq k_{max}$) such that $\lambda_{TC}/h_i > n_k$ where h_i is the diameter of the element.
2. for all elements, define a maximal local admissible time step by $\Delta t_i = h_i \nu_{k_i}/c$;

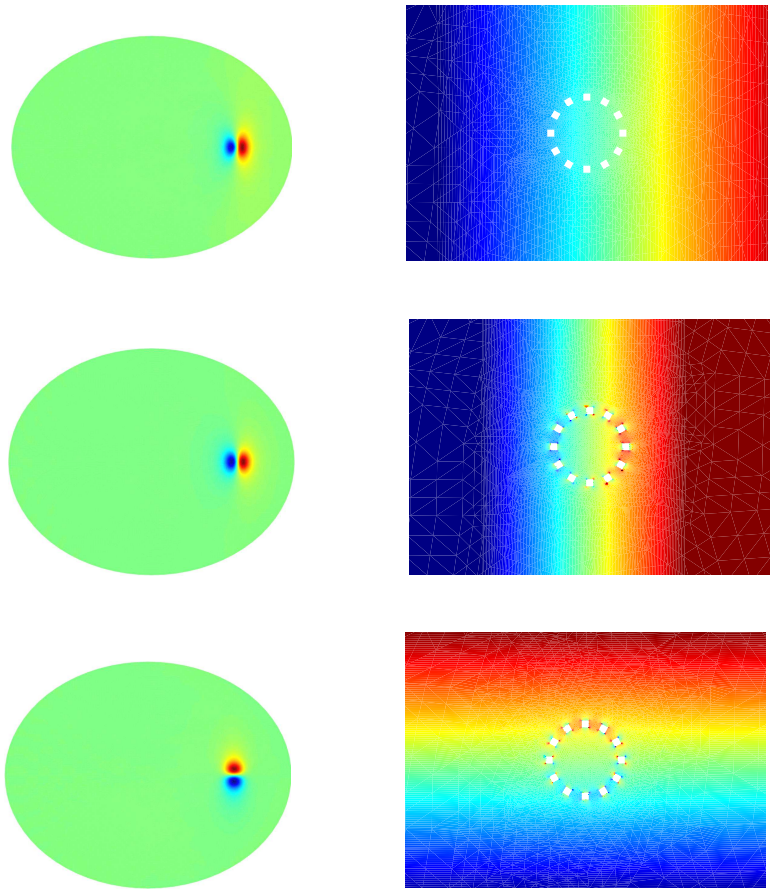

Figure 4: Square inclusion: p/H_z (top), u/E_y (middle), and $v/-E_x$ (bottom) near the inclusion, obtained with algorithm $R^{11}(2.6ms)$ at $t = 4s$ (extremal values for contours on the zooms have been adapted).

3. define an implicit time-step Δt_{imp} and an implicit set of elements such that implicit elements are such that $\Delta t_i \leq \Delta t_{imp}$ and the storage required for the implicit computations is less than the maximal admissible storage (if not, reduce Δt_{imp}).

4. compose classes of explicit elements for the recursive Verlet method.

A toy example has been composed by deforming the mesh shown on Figure 3. It is presented on Figure 5, where we have shown the implicit elements (each connected component must be isolated and we have contoured the number of this connected component for implici elements) and more generally the class number of all elements (we have taken here $k_{min} = 1$, $k_{max} = 4$). One can see that isolated implicit elements are present, which is not a concern at

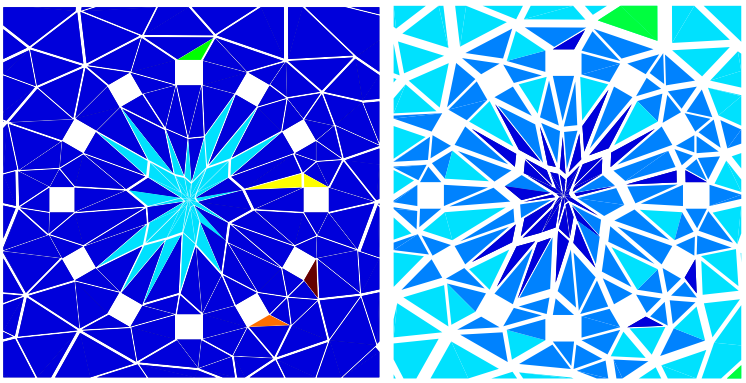

Figure 5: Deformed triangular mesh near the circular array: (number of the implicit connected zone (left) and number of class (right).

all. Implicit elements correspond to the class #1, and the explicit local time step increases with the class number for explicit elements. This sample mesh was not sufficient to produce a significative difference in CPU time. We then proposed a specially designed case: in the mesh of Figure 3, we have moved two neighbouring vertices towards each other, such that their mutual distance has been divided by 100. We then have two very thin elements. The mesh obtained could be seen as a poor result of an automatic mesh generator. We then compared the behaviours of the fully explicit multiscale algorithm and the implicit-explicit mutliscale algorithm. In both cases, the global time step, for an overall $\mathbb{P}_4$-DGTD discretization, is $\Delta t = 1.19ms$. However:

- in the fully explicit multiscale algorithm, the two thin elements are advanced in time with $\Delta t_i = 36.3ns = 1.19/2^{-15}ms$ (there are 16 classes in the computation);

- in the implicit/explicit multiscale algorithm (where only these two-elements are dealt with implicitly), only ten classes are necessary; this computation required a very small storage and a CPU time with a reduction of 36%.

In the present, the CPU time reduction is related to one defect in the mesh. This shows the implicit/explicit multiscale algorithm can lead to important computational time reduction, especially in cases where defects in the mesh are present (small number of ridiculously small elements). Such small elements are not necessarily easy to get rid of, in particular slivers in unstructured tetrahedral meshes.

6 Conclusion

In this paper, we have presented two symplectic algorithms which are able to perform a reversible, energy-conserving, second-order accurate, stable, and adaptive time-integration of the Maxwell's equations after discretization on unstructured meshes using the Discontinuous Galerkin method. The main conclusion is that, if totally centered numerical fluxes are to be used, in order to have no numerical dissipation at all, local time-stepping can overcome the stability limit set by the leapfrog time-scheme.

This kind of algorithm can be particularly valuable if the mesh is distorted or locally refined, i.e. the mesh is refined in a very limited area, for example around a geometrical detail. Two ways have been proposed in this paper. The first one relies on an simple implicit/explicit coupled algorithm. It has been implemented in two space dimensions and is very promising for configurations where the unstructured mesh at hand has very small elements and is difficult to restore. Another totally explicit algorithm, with no additional storage, has been proposed, and leads to very efficient implementations, at least in two space dimensions.

Further works will deal with the implementations in three space dimensions, the local time-stepping algorithm being quite straightforward because the algorithms can be seen has time-step reorganizations only. The main difficult task will certainly consist in obtaining an efficient parallel implementation of these local time-stepping algorithm. In particular, mesh partitioning and message passing have to be optimized.

Annex A. Inverse of the mass matrix (1D).

$$\left(\mathbb{M}^0\right)^{-1} = \begin{pmatrix} 1 \end{pmatrix}$$

$$\left(\mathbb{M}^1\right)^{-1} = 2\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

$$\left(\mathbb{M}^2\right)^{-1} = 3\begin{pmatrix} 3 & -3 & 1 \\ -3 & 7 & -3 \\ 1 & -3 & 3 \end{pmatrix}$$

$$\left(\mathbb{M}^3\right)^{-1} = \frac{4}{3}\begin{pmatrix} 12 & -18 & 12 & -3 \\ -18 & 52 & -43 & 12 \\ 12 & -43 & 52 & -18 \\ -3 & 12 & -18 & 12 \end{pmatrix}$$

$$(\mathbb{M}^4)^{-1} = \frac{5}{2}\begin{pmatrix} 10 & -20 & 20 & -10 & 2 \\ -20 & 70 & -85 & 47 & -10 \\ 20 & -85 & 132 & -85 & 20 \\ -10 & 47 & -85 & 70 & -20 \\ 2 & -10 & 20 & -20 & 10 \end{pmatrix}$$

$$(\mathbb{M}^5)^{-1} = \frac{6}{5}\begin{pmatrix} 30 & -75 & 100 & -75 & 30 & -5 \\ -75 & 310 & -495 & 408 & -173 & 30 \\ 100 & -495 & 954 & -887 & 408 & -75 \\ -75 & 408 & -887 & 954 & -495 & 100 \\ 30 & -173 & 408 & -495 & 310 & -75 \\ -5 & 30 & -75 & 100 & -75 & 30 \end{pmatrix}$$

$$(\mathbb{M}^6)^{-1} = \frac{7}{15}\begin{pmatrix} 105 & -315 & 525 & -525 & 315 & -105 & 15 \\ -315 & 1505 & -2975 & 3255 & -2065 & 715 & -105 \\ 525 & -2975 & 6881 & -8337 & 5671 & -2065 & 315 \\ -525 & 3255 & -8337 & 11229 & -8337 & 3255 & -525 \\ 315 & -2065 & 5671 & -8337 & 6881 & -2975 & 525 \\ -105 & 715 & -2065 & 3255 & -2975 & 1505 & -315 \\ 15 & -105 & 315 & -525 & 525 & -315 & 105 \end{pmatrix}$$

$$(\mathbb{M}^7)^{-1} = \frac{8}{7}\begin{pmatrix} 56 & -196 & 392 & -490 & 392 & -196 & 56 & -7 \\ -196 & 1064 & -2506 & 3416 & -2884 & 1496 & -439 & 56 \\ 392 & -2506 & 6776 & -10108 & 9080 & -4927 & 1496 & -196 \\ -490 & 3416 & -10108 & 16424 & -15823 & 9080 & -2884 & 392 \\ 392 & -2884 & 9080 & -15823 & 16424 & -10108 & 3416 & -490 \\ -196 & 1496 & -4927 & 9080 & -10108 & 6776 & -2506 & 392 \\ 56 & -439 & 1496 & -2884 & 3416 & -2506 & 1064 & -196 \\ -7 & 56 & -196 & 392 & -490 & 392 & -196 & 56 \end{pmatrix}$$

Annex B. Example of actual implementations.

$\mathbb{P}_2$-DGTD implementation in 2D.

In two space dimensions, for a $\mathbb{P}_2$ DGTD method, there are 6 degrees of freedom per triangle and 3 on each edge. The 6 basis functions are numbered as:

$$\begin{aligned} \boldsymbol{\alpha}[1] &= (2,0,0) \quad \pi^k_{[1]} = \lambda_1^2 \\ \boldsymbol{\alpha}[2] &= (1,1,0) \quad \pi^k_{[2]} = 2\lambda_1\lambda_2 \\ \boldsymbol{\alpha}[3] &= (0,2,0) \quad \pi^k_{[3]} = \lambda_2^2 \\ \boldsymbol{\alpha}[4] &= (1,0,1) \quad \pi^k_{[4]} = 2\lambda_1\lambda_3 \\ \boldsymbol{\alpha}[5] &= (0,1,1) \quad \pi^k_{[5]} = 2\lambda_2\lambda_3 \\ \boldsymbol{\alpha}[6] &= (0,0,2) \quad \pi^k_{[6]} = \lambda_3^2. \end{aligned}$$

The mass matrix and its inverse are given by

$$\mathbb{M}^2 = \frac{V_i}{90}\begin{pmatrix} 6&3&1&3&1&1\\ 3&4&3&2&2&1\\ 1&3&6&1&3&1\\ 3&2&1&4&2&3\\ 1&2&3&2&4&3\\ 1&1&1&3&3&6 \end{pmatrix},\ (\mathbb{M}^2)^{-1} = \frac{1}{V_i}\begin{pmatrix} 36&-24&6&-24&6&6\\ -24&66&-24&-9&-9&6\\ 6&-24&36&6&-24&6\\ -24&-9&6&66&-9&-24\\ 6&-9&-24&-9&66&-24\\ 6&6&6&-24&-24&36 \end{pmatrix}.$$

The three following stiffness matrices $\mathbb{K}^q$ $(q = 1, 2, 3)$ are computed:

$$\mathbb{K}^q_{\boldsymbol{\alpha\beta}} = \frac{c}{V_i}\int_{\mathcal{T}_s} \left(\pi^k_{\boldsymbol{\beta}}\,\partial_{\lambda_q}\pi^k_{\alpha} - \pi^k_{\boldsymbol{\alpha}}\,\partial_{\lambda_q}\pi^k_{\boldsymbol{\beta}}\right)\,dx.$$

They are given (here $c = 15$) by

$$\mathbb{K}^1 = \begin{pmatrix} 0&1&1&1&1&1\\ -1&0&3&0&2&1\\ -1&-3&0&-1&0&0\\ -1&0&1&0&2&3\\ -1&-2&0&-2&0&0\\ -1&-1&0&-3&0&0 \end{pmatrix}, \mathbb{K}^2 = \begin{pmatrix} 0&-3&-1&0&-1&0\\ 3&0&-1&2&0&1\\ 1&1&0&1&1&1\\ 0&-2&-1&0&-2&0\\ 1&0&-1&2&0&3\\ 0&-1&-1&0&-3&0 \end{pmatrix}, \mathbb{K}^3 = \begin{pmatrix} 0&0&0&-3&-1&-1\\ 0&0&0&-2&-2&-1\\ 0&0&0&-1&-3&-1\\ 3&2&1&0&0&-1\\ 1&2&3&0&0&-1\\ 1&1&1&1&1&0 \end{pmatrix}.$$

Finally, the 6 permutations (σ_1 to σ_6) of the vertices of a triangle are numbered as

$$\begin{array}{ll} \sigma_1:\ (1,2,3)\to(1,2,3), & \sigma_2:\ (1,2,3)\to(1,3,2),\\ \sigma_3:\ (1,2,3)\to(2,1,3), & \sigma_4:\ (1,2,3)\to(2,3,1),\\ \sigma_5:\ (1,2,3)\to(3,1,2), & \sigma_6:\ (1,2,3)\to(3,2,1), \end{array}$$

For each edge in the mesh, we need to identify the right degrees of freedom in order to compute edge integrals. If the edge was joining the vertex "1" to the vertex "2" of a neighbouring triangle, then the 3 basis functions which do not vanish on this edge would be $\pi^k_{[1]} = \lambda_1^2$, $\pi^k_{[2]} = 2\lambda_1\lambda_2$, and $pi^k_{[3]} = \lambda_2^2$, i.e. the three basis functions in one space dimension. In general, the vertices in the triangle must be permuted such that a given edge joins the vertex "1" to the vertex "2" after the permutation. Thus, the basis functions which do not vanish are not in general $\pi^k_{[1]}, \pi^k_{[2]}, \pi^k_{[3]}$. We have to compute the three indices. They are given in the array below:

$$\Theta = \begin{pmatrix} 1&1&3&3&6&6\\ 2&4&2&5&4&5\\ 3&6&1&6&1&3 \end{pmatrix}.$$

The array Θ is used as follows: assume a given edge joins the vertex "3" to the vertex "1" of a neighbouring triangle. Then the permutation σ_5 is required such that $3 = \sigma_5(1)$ and $1 = \sigma_5(2)$. Then $\Theta(.,5) = {}^t(6,4,1)$ yields the basis functions which play the role of λ_1^2, $2\lambda_1\lambda_2$, and λ_2^2 (here $\pi^k_{[6]} = \lambda_3^2$, $\pi^k_{[5]} = 2\lambda_3\lambda_1$, $\pi^k_{[1]} = \lambda_1^2$).

$\mathbb{P}_3$-DGTD implementation in 2D.

We give here the same hints for the two-dimensional $\mathbb{P}_3$ DGTD method on tirangles. There are 10 degrees of freedom per triangle and 4 on each edge. The basis functions are given in

Section 3.4. The mass matrix and its inverse are given by

$$\mathbb{M}^3 = \frac{V_i}{560}\begin{pmatrix} 20 & 10 & 4 & 1 & 10 & 4 & 1 & 4 & 1 & 1 \\ 10 & 12 & 9 & 4 & 6 & 6 & 3 & 3 & 2 & 1 \\ 4 & 9 & 12 & 10 & 3 & 6 & 6 & 2 & 3 & 1 \\ 1 & 4 & 10 & 20 & 1 & 4 & 10 & 1 & 4 & 1 \\ 10 & 6 & 3 & 1 & 12 & 6 & 2 & 9 & 3 & 4 \\ 4 & 6 & 6 & 4 & 6 & 8 & 6 & 6 & 6 & 4 \\ 1 & 3 & 6 & 10 & 2 & 6 & 12 & 3 & 9 & 4 \\ 4 & 3 & 2 & 1 & 9 & 6 & 3 & 12 & 6 & 10 \\ 1 & 2 & 3 & 4 & 3 & 6 & 9 & 6 & 12 & 10 \\ 1 & 1 & 1 & 1 & 4 & 4 & 4 & 10 & 10 & 20 \end{pmatrix},$$

$$\left(\mathbb{M}^3\right)^{-1} = \frac{10}{V_i}\begin{pmatrix} 10 & -10 & 5 & -1 & -10 & 5 & -1 & 5 & -1 & -1 \\ -10 & 30 & -21 & 5 & 0 & -9 & 3 & 3 & 1 & -1 \\ 5 & -21 & 30 & -10 & 3 & -9 & 0 & 1 & 3 & -1 \\ -1 & 5 & -10 & 10 & -1 & 5 & -10 & -1 & 5 & -1 \\ -10 & 0 & 3 & -1 & 30 & -9 & 1 & -21 & 3 & 5 \\ 5 & -9 & -9 & 5 & -9 & 40 & -9 & -9 & -9 & 5 \\ -1 & 3 & 0 & -10 & 1 & -9 & 30 & 3 & -21 & 5 \\ 5 & 3 & 1 & -1 & -21 & -9 & 3 & 30 & 0 & -10 \\ -1 & 1 & 3 & 5 & 3 & -9 & -21 & 0 & 30 & -10 \\ -1 & -1 & -1 & -1 & 5 & 5 & 5 & -10 & -10 & 10 \end{pmatrix}.$$

The three stiffness matrices $\mathbb{K}^q$ $(q = 1, 2, 3)$ are not given here. The array Θ yielding right degrees of freedom on egdes is given, for our choices of permutations σ_j and of numbering of basis funcions $\pi^k_{[i]}$, by:

$$\Theta = \begin{pmatrix} 1 & 1 & 4 & 4 & 10 & 10 \\ 2 & 5 & 3 & 7 & 8 & 9 \\ 3 & 8 & 2 & 9 & 5 & 7 \\ 4 & 10 & 1 & 10 & 1 & 4 \end{pmatrix}.$$

$\mathbb{P}_4$-DGTD implementation in 2D.

The same elements are given for the two-dimensional $\mathbb{P}_4$ DGTD method on triangles. There are 15 degrees of freedom per triangle and 4 on each edge. The basis functions are numbered

as

$$
\begin{array}{ll}
\boldsymbol{\alpha}[1] = (4,0,0) & \pi^k_{[1]} = \lambda_1^4 \\
\boldsymbol{\alpha}[2] = (3,1,0) & \pi^k_{[2]} = 4\lambda_1^3\lambda_2 \\
\boldsymbol{\alpha}[3] = (2,2,0) & \pi^k_{[3]} = 6\lambda_1^2\lambda_2^2 \\
\boldsymbol{\alpha}[4] = (1,3,0) & \pi^k_{[4]} = 4\lambda_1\lambda_2^3 \\
\boldsymbol{\alpha}[5] = (0,4,0) & \pi^k_{[5]} = \lambda_2^4 \\
\boldsymbol{\alpha}[6] = (3,0,1) & \pi^k_{[6]} = 4\lambda_1^3\lambda_3 \\
\boldsymbol{\alpha}[7] = (2,1,1) & \pi^k_{[7]} = 12\lambda_1^2\lambda_2\lambda_3 \\
\boldsymbol{\alpha}[8] = (1,2,1) & \pi^k_{[8]} = 12\lambda_1\lambda_2^2\lambda_3 \\
\boldsymbol{\alpha}[9] = (0,3,1) & \pi^k_{[9]} = 4\lambda_2^3\lambda_3 \\
\boldsymbol{\alpha}[10] = (2,0,2) & \pi^k_{[10]} = 6\lambda_1^2\lambda_3^2 \\
\boldsymbol{\alpha}[11] = (1,1,2) & \pi^k_{[11]} = 12\lambda_1\lambda_2\lambda_3^2 \\
\boldsymbol{\alpha}[12] = (0,2,2) & \pi^k_{[12]} = 6\lambda_2^2\lambda_3^2 \\
\boldsymbol{\alpha}[13] = (1,0,3) & \pi^k_{[13]} = 4\lambda_1\lambda_3^3 \\
\boldsymbol{\alpha}[14] = (0,1,3) & \pi^k_{[14]} = 4\lambda_2\lambda_3^3 \\
\boldsymbol{\alpha}[15] = (0,0,4) & \pi^k_{[15]} = \lambda_3^4
\end{array}
$$

The mass matrix and its inverse are given by (with the notation $\bar{m} \equiv -m$):

$$
\mathbb{M}^4 = \frac{V_i}{3150}
\begin{pmatrix}
70 & 35 & 15 & 5 & 1 & 35 & 15 & 5 & 1 & 15 & 5 & 1 & 5 & 1 & 1 \\
35 & 40 & 30 & 16 & 5 & 20 & 20 & 12 & 4 & 10 & 8 & 3 & 4 & 2 & 1 \\
15 & 30 & 36 & 30 & 15 & 10 & 18 & 18 & 10 & 6 & 9 & 6 & 3 & 3 & 1 \\
5 & 16 & 30 & 40 & 35 & 4 & 12 & 20 & 20 & 3 & 8 & 10 & 2 & 4 & 1 \\
1 & 5 & 15 & 35 & 70 & 1 & 5 & 15 & 35 & 1 & 5 & 15 & 1 & 5 & 1 \\
35 & 20 & 10 & 4 & 1 & 40 & 20 & 8 & 2 & 30 & 12 & 3 & 16 & 4 & 5 \\
15 & 20 & 18 & 12 & 5 & 20 & 24 & 18 & 8 & 18 & 18 & 9 & 12 & 8 & 5 \\
5 & 12 & 18 & 20 & 15 & 8 & 18 & 24 & 20 & 9 & 18 & 18 & 8 & 12 & 5 \\
1 & 4 & 10 & 20 & 35 & 2 & 8 & 20 & 40 & 3 & 12 & 30 & 4 & 16 & 5 \\
15 & 10 & 6 & 3 & 1 & 30 & 18 & 9 & 3 & 36 & 18 & 6 & 30 & 10 & 15 \\
5 & 8 & 9 & 8 & 5 & 12 & 18 & 18 & 12 & 18 & 24 & 18 & 20 & 20 & 15 \\
1 & 3 & 6 & 10 & 15 & 3 & 9 & 18 & 30 & 6 & 18 & 36 & 10 & 30 & 15 \\
5 & 4 & 3 & 2 & 1 & 16 & 12 & 8 & 4 & 30 & 20 & 10 & 40 & 20 & 35 \\
1 & 2 & 3 & 4 & 5 & 4 & 8 & 12 & 16 & 10 & 20 & 30 & 20 & 40 & 35 \\
1 & 1 & 1 & 1 & 1 & 5 & 5 & 5 & 5 & 15 & 15 & 15 & 35 & 35 & 70
\end{pmatrix},
$$

$$
\left(\mathbb{M}^4\right)^{-1} = \frac{5}{4V_i}
\begin{pmatrix}
180 & 2\bar{4}0 & 180 & \bar{7}2 & 12 & 2\bar{4}0 & 180 & \bar{7}2 & 12 & 180 & \bar{7}2 & 12 & \bar{7}2 & 12 & 12 \\
2\bar{4}0 & 810 & 8\bar{2}8 & 390 & \bar{7}2 & 75 & 3\bar{8}7 & 243 & \bar{5}1 & 54 & 96 & \bar{3}0 & \bar{5}1 & \bar{9} & 12 \\
180 & 8\bar{2}8 & 1356 & 8\bar{2}8 & 180 & 54 & 1\bar{1}4 & 1\bar{1}4 & 54 & \bar{1}6 & 152 & \bar{1}6 & \bar{3}0 & \bar{3}0 & 12 \\
\bar{7}2 & 390 & 8\bar{2}8 & 810 & 2\bar{4}0 & \bar{5}1 & 243 & 3\bar{8}7 & 75 & \bar{3}0 & 96 & 54 & \bar{9} & \bar{5}1 & 12 \\
12 & \bar{7}2 & 180 & 2\bar{4}0 & 180 & 12 & \bar{7}2 & 180 & 2\bar{4}0 & 12 & \bar{7}2 & 180 & 12 & \bar{7}2 & 12 \\
2\bar{4}0 & 75 & 54 & \bar{5}1 & 12 & 810 & 3\bar{8}7 & 96 & \bar{9} & 8\bar{2}8 & 243 & \bar{3}0 & 390 & \bar{5}1 & \bar{7}2 \\
180 & 3\bar{8}7 & 1\bar{1}4 & 243 & \bar{7}2 & 3\bar{8}7 & 1454 & 6\bar{5}3 & 96 & 1\bar{1}4 & 6\bar{5}3 & 152 & 243 & 96 & \bar{7}2 \\
\bar{7}2 & 243 & 1\bar{1}4 & 3\bar{8}7 & 180 & 96 & 6\bar{5}3 & 1454 & 3\bar{8}7 & 152 & 6\bar{5}3 & 1\bar{1}4 & 96 & 243 & \bar{7}2 \\
12 & \bar{5}1 & 54 & 75 & 2\bar{4}0 & \bar{9} & 96 & 3\bar{8}7 & 810 & \bar{3}0 & 243 & 8\bar{2}8 & \bar{5}1 & 390 & \bar{7}2 \\
180 & 54 & \bar{1}6 & \bar{3}0 & 12 & 8\bar{2}8 & 1\bar{1}4 & 152 & \bar{3}0 & 1356 & 1\bar{1}4 & \bar{1}6 & 8\bar{2}8 & 54 & 180 \\
\bar{7}2 & 96 & 152 & 96 & \bar{7}2 & 243 & 6\bar{5}3 & 6\bar{5}3 & 243 & 1\bar{1}4 & 1454 & 1\bar{1}4 & 3\bar{8}7 & 3\bar{8}7 & 180 \\
12 & \bar{3}0 & \bar{1}6 & 54 & 180 & \bar{3}0 & 152 & 1\bar{1}4 & 8\bar{2}8 & \bar{1}6 & 1\bar{1}4 & 1356 & 54 & 8\bar{2}8 & 180 \\
\bar{7}2 & \bar{5}1 & \bar{3}0 & \bar{9} & 12 & 390 & 243 & 96 & \bar{5}1 & 8\bar{2}8 & 3\bar{8}7 & 54 & 810 & 75 & 2\bar{4}0 \\
12 & \bar{9} & \bar{3}0 & \bar{5}1 & \bar{7}2 & \bar{5}1 & 96 & 243 & 390 & 54 & 3\bar{8}7 & 8\bar{2}8 & 75 & 810 & 2\bar{4}0 \\
12 & 12 & 12 & 12 & 12 & \bar{7}2 & \bar{7}2 & \bar{7}2 & \bar{7}2 & 180 & 180 & 180 & 2\bar{4}0 & 2\bar{4}0 & 180
\end{pmatrix}
$$

The array Θ yielding right degrees of freedom on egdes is given by:

$$\Theta = \begin{pmatrix} 1 & 1 & 5 & 5 & 15 & 15 \\ 2 & 6 & 4 & 9 & 13 & 14 \\ 3 & 10 & 3 & 12 & 10 & 12 \\ 4 & 13 & 2 & 14 & 6 & 9 \\ 5 & 15 & 1 & 15 & 1 & 5 \end{pmatrix}.$$

$\mathbb{P}_2$-DGTD implementation in 3D.

The same elements are given for the three-dimensional $\mathbb{P}_2$ DGTD method on tetrahedra. There are 10 degrees of freedom per tetrahedron and 6 on each face. The basis functions are numbered as

$$\begin{array}{ll} \boldsymbol{\alpha}[1] = (2,0,0,0) & \pi^k_{[1]} = \lambda_1^2 \\ \boldsymbol{\alpha}[2] = (1,1,0,0) & \pi^k_{[2]} = 2\lambda_1\lambda_2 \\ \boldsymbol{\alpha}[3] = (0,2,0,0) & \pi^k_{[3]} = \lambda_2^2 \\ \boldsymbol{\alpha}[4] = (1,0,1,0) & \pi^k_{[4]} = 2\lambda_1\lambda_3 \\ \boldsymbol{\alpha}[5] = (0,1,1,0) & \pi^k_{[5]} = 2\lambda_2\lambda_3 \\ \boldsymbol{\alpha}[6] = (0,0,1,0) & \pi^k_{[6]} = \lambda_3^2 \\ \boldsymbol{\alpha}[7] = (1,0,0,1) & \pi^k_{[7]} = 2\lambda_1\lambda_4 \\ \boldsymbol{\alpha}[8] = (0,1,0,1) & \pi^k_{[8]} = 2\lambda_2\lambda_4 \\ \boldsymbol{\alpha}[9] = (0,0,1,1) & \pi^k_{[9]} = 2\lambda_3\lambda_4 \\ \boldsymbol{\alpha}[10] = (0,0,0,1) & \pi^k_{[10]} = \lambda_4^2 \end{array}$$

The mass matrix and its inverse are given by:

$$\mathbb{M}^2 = \frac{V_i}{210}\begin{pmatrix} 6 & 3 & 1 & 3 & 1 & 1 & 3 & 1 & 1 & 1 \\ 3 & 4 & 3 & 2 & 2 & 1 & 2 & 2 & 1 & 1 \\ 1 & 3 & 6 & 1 & 3 & 1 & 1 & 3 & 1 & 1 \\ 3 & 2 & 1 & 4 & 2 & 3 & 2 & 1 & 2 & 1 \\ 1 & 2 & 3 & 2 & 4 & 3 & 1 & 2 & 2 & 1 \\ 1 & 1 & 1 & 3 & 3 & 6 & 1 & 1 & 3 & 1 \\ 3 & 2 & 1 & 2 & 1 & 1 & 4 & 2 & 2 & 3 \\ 1 & 2 & 3 & 1 & 2 & 1 & 2 & 4 & 2 & 3 \\ 1 & 1 & 1 & 2 & 2 & 3 & 2 & 2 & 4 & 3 \\ 1 & 1 & 1 & 1 & 1 & 1 & 3 & 3 & 3 & 6 \end{pmatrix},$$

$$(\mathbb{M}^2)^{-1} = \frac{1}{V_i}\begin{pmatrix} 10 & -5 & 1 & -5 & 1 & 1 & -5 & 1 & 1 & 1 \\ -5 & 16 & -5 & -2 & -2 & 1 & -2 & -2 & 1 & 1 \\ 1 & -5 & 10 & 1 & -5 & 1 & 1 & -5 & 1 & 1 \\ -5 & -2 & 1 & 16 & -2 & -5 & -2 & 1 & -2 & 1 \\ 1 & -2 & -5 & -2 & 16 & -5 & 1 & -2 & -2 & 1 \\ 1 & 1 & 1 & -5 & -5 & 10 & 1 & 1 & -5 & 1 \\ -5 & -2 & 1 & -2 & 1 & 1 & 16 & -2 & -2 & -5 \\ 1 & -2 & -5 & 1 & -2 & 1 & -2 & 16 & -2 & -5 \\ 1 & 1 & 1 & -2 & -2 & -5 & -2 & -2 & 16 & -5 \\ 1 & 1 & 1 & 1 & 1 & 1 & -5 & -5 & -5 & 10 \end{pmatrix}$$

The 24 permutations (σ_1 to $\sigma_2 4$) of the vertices of a tetrahedron are numbered as

$$\begin{array}{llll}
\sigma_1 :\rightarrow (1,2,3,4), & \sigma_2 :\rightarrow (1,2,4,3), & \sigma_3 :\rightarrow (1,3,2,4), & \sigma_4 :\rightarrow (1,3,4,2),\\
\sigma_5 :\rightarrow (1,4,2,3), & \sigma_6 :\rightarrow (1,4,3,2), & \sigma_7 :\rightarrow (2,1,3,4), & \sigma_8 :\rightarrow (2,1,4,3),\\
\sigma_9 :\rightarrow (2,3,1,4), & \sigma_{10} :\rightarrow (2,3,4,1), & \sigma_{11} :\rightarrow (2,4,1,3), & \sigma_{12} :\rightarrow (2,4,3,1),\\
\sigma_{13} :\rightarrow (3,1,2,4), & \sigma_{14} :\rightarrow (3,1,4,2), & \sigma_{15} :\rightarrow (3,2,1,4), & \sigma_{16} :\rightarrow (3,2,4,1),\\
\sigma_{17} :\rightarrow (3,4,1,2), & \sigma_{18} :\rightarrow (3,4,2,1), & \sigma_{19} :\rightarrow (4,1,2,3), & \sigma_{20} :\rightarrow (4,1,3,2),\\
\sigma_{21} :\rightarrow (4,2,1,3), & \sigma_{22} :\rightarrow (4,2,3,1), & \sigma_{23} :\rightarrow (4,3,1,2), & \sigma_{24} :\rightarrow (4,3,2,1).
\end{array}$$

The array Θ yielding right degrees of freedom on egdes is given by:

$$\Theta = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 & 3 & 3 & 6 & 6 & 6 & 6 & 6 & 6 & 10 & 10 & 10 & 10 & 10 & 10\\
2 & 2 & 4 & 4 & 7 & 7 & 2 & 2 & 5 & 5 & 8 & 8 & 4 & 4 & 5 & 5 & 9 & 9 & 7 & 7 & 8 & 8 & 9 & 9\\
3 & 3 & 6 & 6 & 10 & 10 & 1 & 1 & 6 & 6 & 10 & 10 & 1 & 1 & 3 & 3 & 10 & 10 & 1 & 1 & 3 & 3 & 6 & 6\\
4 & 7 & 2 & 7 & 2 & 4 & 5 & 8 & 2 & 8 & 2 & 5 & 5 & 9 & 4 & 9 & 4 & 5 & 8 & 9 & 7 & 9 & 7 & 8\\
5 & 8 & 5 & 9 & 8 & 9 & 4 & 7 & 4 & 9 & 7 & 9 & 2 & 7 & 2 & 8 & 7 & 8 & 2 & 4 & 2 & 5 & 4 & 5\\
6 & 10 & 3 & 10 & 3 & 6 & 6 & 10 & 1 & 10 & 1 & 6 & 3 & 10 & 1 & 10 & 1 & 3 & 3 & 6 & 1 & 6 & 1 & 3
\end{pmatrix}.$$

References

[1] E. Bécache, P. Joly, and J. Rodríguez. Space-time mesh refinement for elastodynamics. Numerical results. *Comput. Methods Appl. Mech. Eng.*, 194(2-5):355–366, 2005.

[2] N. Canouet, L. Fezoui, and S. Piperno. A discontinuous galerkin method for 3d Maxwell's equation on non-conforming grids. In G. C. Cohen, E. Heikkola, P. Joly, and P. Neittaanmäki, editors, *WAVES 2003, The Sixth International Conference on Mathematical and Numerical Aspects of Wave Propagation*, pages 389–394, Jyväskylä, Finland, 2003. Springer-Verlag.

[3] Jean-Pierre Cioni, L. Fezoui, L. Anne, and F. Poupaud. A parallel FVTD Maxwell solver using 3d unstructured meshes. In *13th annual review of progress in applied computational electromagnetics*, Monterey, California, 1997.

[4] B. Cockburn, G. E Karniadakis, and C.-W. Shu, editors. *Discontinuous Galerkin methods. Theory, computation and applications.*, volume 11 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2000.

[5] A. Elmkies and P. Joly. Éléments finis d'arête et condensation de masse pour les équations de Maxwell: le cas de dimension 3. *C. R. Acad. Sci. Paris Sér. I Math.*, t. 325(11):1217–1222, 1997.

[6] L. Fezoui, S. Lanteri, S. Lohrengel, and S. Piperno. Convergence and stability of a discontinuous galerkin time-domain method for the 3d heterogeneous maxwell equations on unstructured meshes. *RAIRO Modél. Math. Anal. Numér.* to appear.

[7] T. Fouquet. *Raffinement de maillage spatio-temporel pour les équations de Maxwell.* PhD thesis, University Paris 9, France, 2000.

[8] D. J. Hardy, D. I. Okunbor, and R. D. Skeel. Symplectic variable step size integration for n-body problems. *Appl. Numer. Math.*, 29:19–30, 1999.

[9] J. Hesthaven and C. Teng. Stable spectral methods on tetrahedral elements. *J. Sci. Comput.*, 21:2352–2380, 2000.

[10] J. Hesthaven and T. Warburton. Nodal high-order methods on unstructured grids. i: Time-domain solution of Maxwell's equations. *J. Comput. Phys.*, 181:186–221, 2002.

[11] J.S. Hesthaven and T. Warburton. Nodal high-order methods on unstructured grids. i: Time-domain solution of maxwell's equations. *J. Comput. Phys.*, 181(1):186–221, 2002.

[12] T. Hirono, W. W. Lui, and K. Yokoyama. Time-domain simulation of electromagnetic field using a symplectic integrator. *IEEE Microwave Guided Wave Lett.*, 7:279–281, 1997.

[13] T. Hirono, W. W. Lui, K. Yokoyama, and S. Seki. Stability and numerical dispersion of symplectic fourth-order time-domain schemes for optical field simulation. *J. Lightwave Tech.*, 16:1915–1920, 1998.

[14] T. Holder, B. Leimkuhler, and S. Reich. Explicit variable step-size and time-reversible integration. *Appl. Numer. Math.*, 39:367–377, 2001.

[15] W. Huang and B. Leimkuhler. The adaptive verlet method. *J. Sci. Comput.*, 18:239–256, 1997.

[16] J. M. Hyman and M. Shashkov. Mimetic discretizations for Maxwell's equations. *J. Comput. Phys.*, 151:881–909, 1999.

[17] P. Joly and C. Poirier. A new second order 3d edge element on tetrahedra for time dependent Maxwell's equations. In A. Bermudez, D. Gomez, C. Hazard, P. Joly, and J.-E. Roberts, editors, *Fifth International Conference on Mathematical and Numerical Aspects of Wave Propagation*, pages 842–847, Santiago de Compostella, Spain, July 10-14 2000. SIAM.

[18] D. A. Kopriva, S. L. Woodruff, and M. Y. Hussaini. *Discontinuous spectral element approximation of Maxwell's equations*, volume 11 of *Lecture Notes in Computational Science and Engineering*, pages 355–362. Springer-Verlag, Berlin, 2000.

[19] X. Lu and R. Schmide. Symplectic discretization for Maxwell's equations. *J. of Math. and Computing*, 25, 2001.

[20] S. Piperno. Symplectic local time-stepping in non-dissipative dgtd methods applied to wave propagation problems. *RAIRO Modél. Math. Anal. Numér.* submitted for publicqtion.

[21] S. Piperno. Schémas en éléments finis discontinus localement raffinés en espace et en temps pour les équations de maxwell 1d. Technical Report 4986, INRIA, 2003.

[22] S. Piperno. Symplectic local time-stepping in non-dissipative dgtd methods applied to wave propagation problems. Technical Report 5643, INRIA, 2005.

[23] M. Remaki. A new finite volume scheme for solving Maxwell's system. *COMPEL*, 19(3):913–931, 2000.

[24] R. Rieben, D. White, and G. Rodrigue. High-order symplectic integration methods for finite element solutions to time dependent maxwell equations. *IEEE Trans. Antennas and Propagation*, 52:2190–2195, 2004.

[25] J. M. Sanz-Serna and M. P. Calvo. *Numerical Hamiltonian Problems*. Chapman and Hall, London, U.K., 1994.

[26] J.S. Shang and R.M. Fithen. A comparative study of characteristic-based algorithms for the Maxwell equations. *J. Comput. Phys.*, 125:378–394, 1996.

[27] T. Warburton. *Application of the discontinuous Galerkin method to Maxwell's equations using unstructured polymorphic hp-finite elements*, volume 11 of *Lecture Notes in Computational Science and Engineering*, pages 451–458. Springer-Verlag, Berlin, 2000.

[28] T. Warburton. Spurious solutions and the discontinuous galerkin method on non-conforming meshes. In *WAVES 2005, The Seventh International Conference on Mathematical and Numerical Aspects of Wave Propagation*, pages 405–407, Providence, RI., 2005.

[29] K. S. Yee. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas and Propagation*, AP-16:302–307, 1966.

Contents

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399