Explicit implementation strategy of high order edge finite elements and Schwarz preconditioning for the time-harmonic Maxwell's equations

Marcella Bonazzoli, Victorita Dolean, Frédéric Hecht, Francesca Rapetti

▶ **To cite this version:**

Marcella Bonazzoli, Victorita Dolean, Frédéric Hecht, Francesca Rapetti. Explicit implementation strategy of high order edge finite elements and Schwarz preconditioning for the time-harmonic Maxwell's equations. 2017. hal-01298938v2

HAL Id: hal-01298938

https://hal.science/hal-01298938v2

Preprint submitted on 14 May 2017 (v2), last revised 12 Nov 2017 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Explicit implementation strategy of high order edge finite elements and Schwarz preconditioning for the time-harmonic Maxwell's equations

Marcella Bonazzoli[1], Victorita Dolean[1,2], Frédéric Hecht[3], and Francesca Rapetti[1]

[1] *Université Côte d'Azur, CNRS, LJAD, France. E-mail: marcella.bonazzoli@unice.fr, victorita.dolean@unice.fr, francesca.rapetti@unice.fr*
[2] *University of Strathclyde, Glasgow, UK. E-mail: victorita.dolean@strath.ac.uk*
[3] *UPMC Univ Paris 6, LJLL, Paris, France. E-mail: frederic.hecht@upmc.fr*

Abstract

In this paper we focus on high order finite element approximations of the electric field combined with suitable preconditioners, to solve the time-harmonic Maxwell's equations in waveguide configurations. The implementation of high order curl-conforming finite elements is quite delicate, especially in the three-dimensional case. Here, we explicitly describe an implementation strategy, which has been embedded in the open source finite element software FreeFem++ (`http://www.freefem.org/ff++/`). In particular, we use the inverse of a generalized Vandermonde matrix to build a basis of generators in duality with the degrees of freedom, resulting in an easy-to-use but powerful interpolation operator. We carefully address the problem of applying the same Vandermonde matrix to possibly differently oriented tetrahedra of the mesh over the computational domain. We investigate the preconditioning for Maxwell's equations in the time-harmonic regime, which is an underdeveloped issue in the literature, particularly for high order discretizations. In the numerical experiments, we study the effect of varying several parameters on the spectrum of the matrix preconditioned with overlapping Schwarz methods, both for 2d and 3d waveguide configurations.

Keywords: High order finite elements; edge elements; Schwarz preconditioners; time-harmonic Maxwell's equations; FreeFem++.

1 Introduction

Developing high-speed microwave field measurement systems for wireless, medical or engineering industries is a challenging task. These systems often rely on high frequency (from 1 to 60 GHz) electromagnetic wave propagation in waveguides, and the underlying mathematical model is given by Maxwell's equations. High order finite elements methods make it possible, for a given precision, to reduce significantly the number of unknowns, and they are particularly well suited to discretize wave propagation problems since they can provide a solution with very low dispersion and dissipation errors. However, the resulting algebraic linear systems can be ill conditioned, so that preconditioning becomes mandatory when using iterative solvers.

In the case of finite elements (FE) for Maxwell's equations, degrees of freedom (dofs) are associated with geometrical mesh elements other than nodes, such as edges or faces. Indeed, one needs to recognize that different physical quantities have different properties and must be treated accordingly. Whitney finite elements are thus generally adopted [1, 2]. The high order version of Whitney finite elements we consider here is the one developed in [3, 4] (for other possible high order finite element bases see for example [5, 6, 7, 8, 9]). In particular, within the family of Whitney finite elements, we address the edge elements case, which is the standard choice to describe the electric field solution of a waveguide propagation

problem. We thus added high order edge finite elements to FreeFem++ [10]. FreeFem++ is an open source domain specific language (DSL) specialized in solving boundary value problems (BVP) by using variational methods, and it is based on a natural transcription of the weak formulation of the considered BVP. Moreover, the user can add new finite elements to it by defining certain ingredients including an interpolation operator. For the definition of the latter, in the high order edge elements case we need the generalized Vandermonde matrix introduced in [11] to build a basis of generators in duality with the degrees of freedom. In this work, we carefully address various implementation issues, as the problem of applying the same Vandermonde matrix to possibly differently oriented simplices (triangles, tetrahedra) of the whole mesh, in order to be able to use in numerical experiments the concepts presented for just one simplex in [11]. Note that in FreeFem++ the basis functions are constructed locally, i.e. in each simplex, without the need of a transformation from the reference simplex; the chosen definition of high order generators fits perfectly this local construction feature of FreeFem++ since it involves only the barycentric coordinates of the simplex.

For Maxwell's equations in the *time domain*, for which an implicit time discretization yields at each step a positive definite problem, there are many good solvers and preconditioners in the literature: multigrid or auxiliary space methods, see e.g. [12, 13, 14, 15] for low order finite elements, [16] for high order ones, and Schwarz domain decomposition methods, see e.g. [17, 18]. In this paper, we are interested in solving Maxwell's equations in the *frequency domain*, also called the *time-harmonic* Maxwell's equations: these involve the inherent difficulties of the *indefinite* Helmholtz equation, which is difficult to solve for high frequencies with classical iterative methods [19]. It is widely recognized that domain decomposition methods or preconditioners are key in solving efficiently Maxwell's equations in the time-harmonic regime.

The first domain decomposition method for the time-harmonic Maxwell's equations was proposed by Després in [20]. Further improvements can be found in [21] where modified, more efficient Robin transmission conditions are used at the interfaces between subdomains. Over the last decade, optimized Schwarz methods were developed: for the first order formulation of the equations complete optimized results are known, also in the case of conductive medium [18, 22], while for the second order (or curl-curl) formulation partial optimization results were obtained in various works. Recently it has been shown that the convergence factors and the optimization process for the two formulations are the same [23].

Nevertheless, the development of Schwarz algorithms and preconditioners for high order discretizations is still an open issue. A recent work for the non overlapping case is reported in [24]. In the present work, we use overlapping Schwarz preconditioners based on impedance transmission conditions for high order discretizations of the curl-curl formulation of time-harmonic Maxwell's equations. Note that domain decomposition preconditioners are suited by construction to parallel computing, which is necessary for large scale simulations. The coupling of high order edge finite elements with domain decomposition preconditioners studied in this paper has been applied in [25] to a large scale problem, coming from a practical application in microwave brain imaging: there, it is shown that the high order approximation of degree 2 makes it possible to attain a given accuracy with much fewer unknowns and much less computing time than the lowest order approximation.

The paper is organized as follows. In Section 2 we introduce the waveguide time-harmonic problem and its variational formulation. In Section 3 we recall the definition of generators and degrees of freedom that we adopted here as high order edge FEs. Then, in Section 4 we describe in detail the implementation issues of these FEs, the strategy developed to overcome those difficulties and the ingredients to add them as a new FE in FreeFem++. The overlapping Schwarz preconditioners we used are described in Section 5, followed in Section 6 by the numerical experiments, both in two and three dimensions.

2 The waveguide problem

Waveguides are used to transfer electromagnetic power efficiently from one point in space, where an antenna is located, to another, where electronic components treat the in/out

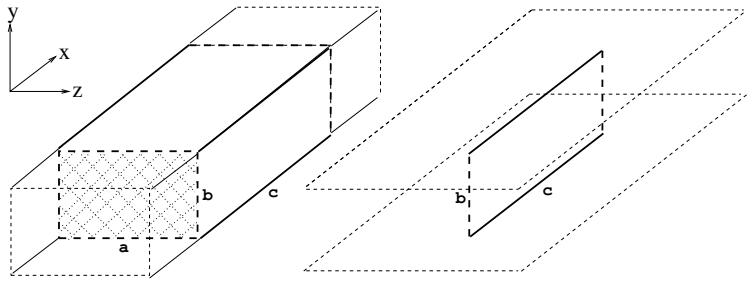


Figure 1: Rectangular waveguide configuration for 3d (left) and 2d (right) problems with wave propagation in the x-direction. The physical domain $\mathcal{D}$ is in thin line, with dashed style for those boundaries that should be extended to infinity. The computational domain Ω is in thick line, with dashed style for those boundaries where suitable absorbing conditions are imposed.

information. Rectangular waveguides, which are considered here, are often used to transfer large amounts of microwave power at frequencies greater than 2 GHz. In this section, we describe in detail the derivation of the simple but physically meaningful boundary value problem which simulates the electromagnetic wave propagation in such waveguide structures. To work in the frequency domain, we restrict the analysis to a time-harmonic electromagnetic field varying with an angular frequency $\omega > 0$. For all times $t \in \mathbb{R}$, we consider the representation of the electric field $\boldsymbol{\mathcal{E}}$ and the magnetic field $\boldsymbol{\mathcal{H}}$ as

$$\boldsymbol{\mathcal{E}}(\mathbf{x},t) = \Re(\mathbf{E}(\mathbf{x})e^{\mathtt{i}\omega t}), \qquad \boldsymbol{\mathcal{H}}(\mathbf{x},t) = \Re(\mathbf{H}(\mathbf{x})e^{\mathtt{i}\omega t}),$$

where $\mathbf{E}(\mathbf{x})$, $\mathbf{H}(\mathbf{x})$ are the complex amplitudes, for all $\mathbf{x} \in \mathcal{D}$, $\mathcal{D} \subset \mathbb{R}^3$ being the considered physical domain. The mathematical model is thus given by the (fist order) *time-harmonic Maxwell's equations*:

$$\nabla \times \mathbf{H} = \mathtt{i}\omega\varepsilon_\sigma \mathbf{E}, \qquad \nabla \times \mathbf{E} = -\mathtt{i}\omega\mu\mathbf{H},$$

where μ is the magnetic permeability and ε_σ the electric permittivity of the considered medium in $\mathcal{D}$. To include dissipative effects, we work with a complex valued ε_σ, related to the dissipation-free electric permittivity ε and the electrical conductivity σ by the relation $\varepsilon_\sigma = \varepsilon - \mathtt{i}\frac{\sigma}{\omega}$. This assumption holds in the regions of $\mathcal{D}$ where the current density $\mathbf{J}$ is of conductive type, that is, $\mathbf{J}$ and $\mathbf{E}$ are related by Ohm's law $\mathbf{J} = \sigma\mathbf{E}$. Both ε and μ are assumed to be positive, bounded functions. Expressing Maxwell's equations in terms of the electric field, and supposing that μ is constant, we obtain the *second order (or curl-curl) formulation*

$$\nabla \times (\nabla \times \mathbf{E}) - \gamma^2 \mathbf{E} = \mathbf{0}, \tag{1}$$

where the (complex-valued) coefficient γ is related to the physical parameters as follows

$$\gamma = \sqrt{\omega^2\mu\varepsilon - \mathtt{i}\omega\mu\sigma} = \omega\sqrt{\mu\varepsilon_\sigma}, \qquad \varepsilon_\sigma = \varepsilon - \mathtt{i}\frac{\sigma}{\omega}.$$

Note that if $\sigma = 0$, we have $\gamma = \tilde{\omega}$, $\tilde{\omega} = \omega\sqrt{\mu\varepsilon}$ being the wavenumber.

Equation (1) is to be solved in a suitable bounded section Ω of the physical domain $\mathcal{D}$, as shown in Fig. 1. In the 3d case, the physical domain $\mathcal{D} \subset \mathbb{R}^3$ is an infinite 'parallelepiped' parallel to the x-direction and the computational domain is a bounded section, say $\Omega = (0,\mathtt{X}) \times (0,\mathtt{Y}) \times (0,\mathtt{Z}) = (0,\mathtt{c}) \times (0,\mathtt{b}) \times (0,\mathtt{a})$ of $\mathcal{D}$. In the 2d case, the physical domain $\mathcal{D} \subset \mathbb{R}^3$ is the space contained between two infinite parallel metallic plates, say $y = 0$, $y = \mathtt{b}$, and all physical parameters μ, σ, ε have to be assumed invariant in the z-direction. The computational domain in 2d is a bounded section, say $\Omega = (0,\mathtt{X}) \times (0,\mathtt{Y}) = (0,\mathtt{c}) \times (0,\mathtt{b})$, of $\mathcal{D}$. In both 2d and 3d cases, the wave propagates in the x-direction within the domain.

Let $\mathbf{n}$ be the unit outward normal to $\partial\Omega$. We solve the boundary value problem given by equation (1), with metallic boundary conditions

$$\mathbf{E} \times \mathbf{n} = \mathbf{0}, \text{ on } \Gamma_{\mathrm{w}}, \tag{2}$$

on the waveguide perfectly conducting walls $\Gamma_{\mathrm{w}} = \{\mathbf{x} \in \partial\Omega,\ \mathbf{n}(\mathbf{x}) \cdot \mathbf{e}_x = 0\}$, with $\mathbf{e}_x = (1,0,0)^t$, and impedance boundary conditions

$$\begin{aligned} &(\nabla \times \mathbf{E}) \times \mathbf{n} + \mathtt{i}\eta \mathbf{n} \times (\mathbf{E} \times \mathbf{n}) = \mathbf{g}^{\mathrm{in}}, \text{ on } \Gamma_{\mathrm{in}}, \qquad \eta \in \mathbb{R}^+, \\ &(\nabla \times \mathbf{E}) \times \mathbf{n} + \mathtt{i}\eta \mathbf{n} \times (\mathbf{E} \times \mathbf{n}) = \mathbf{g}^{\mathrm{out}}, \text{ on } \Gamma_{\mathrm{out}}, \end{aligned} \tag{3}$$

at the waveguide entrance $\Gamma_{\mathrm{in}} = \{\mathbf{x} \in \partial\Omega,\ \mathbf{n}(\mathbf{x}) \cdot \mathbf{e}_x < 0\}$, and exit $\Gamma_{\mathrm{out}} = \{\mathbf{x} \in \partial\Omega,\ \mathbf{n}(\mathbf{x}) \cdot \mathbf{e}_x > 0\}$. The vectors $\mathbf{g}^{\mathrm{in}}$, $\mathbf{g}^{\mathrm{out}}$ depend on the incident wave. On one hand, the impedance conditions on the artificial boundaries Γ_{in}, Γ_{out} are absorbing boundary conditions, first order approximations of transparent boundary conditions defined to let outgoing waves pass through Ω unaffected; they mathematically translate the fact that Ω is a truncated part of an infinite domain $\mathcal{D}$. On the other hand, they simply model the fact that the waveguide is connected to electronic components such as co-axial cables or antennas.

To cast in the weak form the continuous problem (1) with boundary conditions (2) and (3), one has to multiply (1) by the complex conjugate of a test function $\mathbf{v}$ of a suitable functional space V and integrate by parts over the computational domain Ω. More precisely, the weak problem reads: find $\mathbf{E} \in V$ such that

$$\begin{aligned} \int_\Omega \Big[(\nabla \times \mathbf{E}) \cdot (\nabla \times \overline{\mathbf{v}}) - \gamma^2 \mathbf{E} \cdot \overline{\mathbf{v}}\Big] + \int_{\Gamma_{\mathrm{in}} \cup \Gamma_{\mathrm{out}}} \mathtt{i}\eta (\mathbf{E} \times \mathbf{n}) \cdot (\overline{\mathbf{v}} \times \mathbf{n}) \\ = \int_{\Gamma_{\mathrm{in}}} \mathbf{g}^{\mathrm{in}} \cdot \overline{\mathbf{v}} + \int_{\Gamma_{\mathrm{out}}} \mathbf{g}^{\mathrm{out}} \cdot \overline{\mathbf{v}} \quad \forall \mathbf{v} \in V, \end{aligned} \tag{4}$$

with $V = \{\mathbf{v} \in H(\mathrm{curl}, \Omega), \mathbf{v} \times \mathbf{n} = 0 \text{ on } \Gamma_{\mathrm{w}}\}$, where $H(\mathrm{curl}, \Omega)$ is the space of square integrable functions whose curl is also square integrable. For a detailed discussion about existence and uniqueness of solutions we refer to [26]. Note that in this paper we chose the *sign convention* with $e^{+\mathtt{i}\omega t}$ in the time-harmonic assumption, and therefore negative imaginary part in the complex valued electric permittivity $\varepsilon_\sigma = \varepsilon - \mathtt{i}\sigma/\omega$ and positive parameter η in the impedance boundary condition (3).

3 High order edge finite elements

Consider a simplicial (triangular in 2d, tetrahedral in 3d) mesh $\mathcal{T}_h$ over $\bar{\Omega}$, where h denotes the maximal diameter of simplices in $\mathcal{T}_h$. The unknown $\mathbf{E}$ and the functional operators on it have meaningful discrete equivalents if we work in the curl-conforming finite dimensional subspace $V_h \subset H(\mathrm{curl}, \Omega)$ of *Nédélec edge finite elements* [27]. For a simplex $T \in \mathcal{T}_h$, the local *lowest order basis functions* for the Nédélec curl-conforming space are associated with the oriented edges $e = \{n_i, n_j\}$ of T as follows

$$\mathbf{w}^e = \lambda_{n_i} \nabla \lambda_{n_j} - \lambda_{n_j} \nabla \lambda_{n_i}, \tag{5}$$

where the λ_{n_ℓ} are the barycentric coordinates of a point $\mathbf{x} \in T$ with respect to the node n_ℓ of T of Cartesian coordinates $\mathbf{x}_\ell$. The *degrees of freedom* (*dofs*) over T are defined as the functionals

$$\xi_e \colon \mathbf{w} \mapsto \frac{1}{|e|} \int_e \mathbf{w} \cdot \mathbf{t}_e, \quad \forall\, e \in \mathcal{E}(T),$$

where $\mathbf{t}_e = \mathbf{x}_j - \mathbf{x}_i$ is the tangent vector to the edge e, $|e| = |\mathbf{t}_e|$ the length of e and $\mathcal{E}(T)$ the set of edges of T. At the lowest order, the basis functions are in *duality* with the dofs, that is $\xi_e(\mathbf{w}^{e'}) = 1$, resp. 0, if $e = e'$, resp. if $e \neq e'$. As a consequence, the coefficients that define the Galerkin projection $\mathbf{E}_h$ of the field $\mathbf{E}$ onto V_h are the *circulations* of $\mathbf{E}_h$ along the oriented edges e of the simplicial mesh $\mathcal{T}_h$: locally, in each $T \in \mathcal{T}_h$, we have

$$\mathbf{E}(\mathbf{x}) \approx \mathbf{E}_h(\mathbf{x}) = \sum_{e \in \mathcal{E}(T)} c_e \mathbf{w}^e(\mathbf{x}), \quad \forall\, \mathbf{x} \in T, \qquad c_e = \frac{1}{|e|} \int_e \mathbf{E}_h \cdot \mathbf{t}_e.$$

There are several reasons to rely on edge elements rather than on other FE discretizations of $H(\mathrm{curl}, \Omega)$ [28]. By construction, edge elements guarantee the continuity of the tangential

components across inter-element interfaces, they thus fit the continuity properties of the electric field. In addition, for propagation problems, edge elements are known to avoid the pollution of the numerical solution by spurious modes [29, 30].

High order curl-conforming finite elements of Nédélec type have become established techniques in computational electromagnetism. Their popularity for wave propagation problems is due to the fact that they are characterized by low numerical dispersion and dissipation errors. Moreover, at a fixed number of dofs, their numerical accuracy is higher.

We adopt the high order generators of Nédélec elements presented in [3, 4]: the definition of these generators is rather simple since it only involves the barycentric coordinates of the simplex (see also [8] for previous work in this direction). Moreover, we consider the friendly definition of the degrees of freedom and the 'dualizing' Vandermonde matrix which were introduced in [11] for all the spaces of the complex $H_{\rm grad} \xrightarrow{\nabla} H_{\rm curl} \xrightarrow{\nabla\times} H_{\rm div} \xrightarrow{\nabla\cdot} L^2$,. Here, to make the presentation more accessible, we recall the definitions for the case of $H_{\rm curl}$ in which we are interested, and highlight the relevant properties, also with illustrative examples. Then, in Section 4 we describe how to deal with the delicate implementation issues of these finite elements.

To state the definitions and further properties, we need to introduce multi-index notations. A multi-index is an array $\mathbf{k} = (k_1, \ldots, k_\nu)$ of ν integers $k_i \geq 0$, and its weight k is $\sum_{i=1}^{\nu} k_i$. The set of multi-indices $\mathbf{k}$ with ν components and of weight k is denoted $\mathcal{I}(\nu, k)$. If $d = 2, 3$ is the ambient space dimension, we consider $\nu \leq d+1$ and, given $\mathbf{k} \in \mathcal{I}(\nu, k)$, we set $\lambda^{\mathbf{k}} = \prod_{i=1}^{\nu} (\lambda_{n_i})^{k_i}$, where the n_i are ν nodes of the $d+1$ nodes of T. Now, in the generators definition we take $\nu = d+1$ and $k = r-1$, with r the polynomial degree of the generators.

Definition 3.1 (Generators)**.** The generators for Nédélec edge element spaces $W^1_{h,r}(T)$ of *degree* $r \geq 1$ in a simplex $T \in \mathcal{T}_h$ are the $\lambda^{\mathbf{k}}\mathbf{w}^e$, with $\mathbf{k} \in \mathcal{I}(d+1, k)$, $k = r-1$ and $e \in \mathcal{E}(T)$. The $\mathbf{w}^e$ are the low order edge basis functions (5) (note that the polynomial degree of the $\mathbf{w}^e$ is $r = 1$ and they are obtained with $k = 0$).

In Section 1.2 of [27] $W^1_{h,r}(T)$-unisolvent dofs are presented, for any $r \geq 1$ (the space $W^1_{h,r}(T)$ is indeed a discrete counterpart of $H(\mathrm{curl}, T) = \{\mathbf{v} \in L_2(T)^3, \nabla \times \mathbf{v} \in L_2(T)^3\}$). By relying on the generators introduced in Definition 3.1, the functionals in [27] can be recast in a new more friendly form as follows (see details in [11], which are inspired by [26]).

Definition 3.2 (Degrees of freedom)**.** For $r \geq 1$, $d = 3$, the functionals

$$\xi_e \colon \mathbf{w} \mapsto \frac{1}{|e|} \int_e (\mathbf{w} \cdot \mathbf{t}_e)\, q, \qquad \forall\, q \in \mathbb{P}_{r-1}(e),\ \forall\, e \in \mathcal{E}(T), \tag{6}$$

$$\xi_f \colon \mathbf{w} \mapsto \frac{1}{|f|} \int_f (\mathbf{w} \cdot \mathbf{t}_{f,i})\, q, \qquad \forall\, q \in \mathbb{P}_{r-2}(f),\ \forall\, f \in \mathcal{F}(T), \tag{7}$$

$$\mathbf{t}_{f,i} \text{ two independent sides of } f,\ i = 1, 2,$$

$$\xi_T \colon \mathbf{w} \mapsto \frac{1}{|T|} \int_T (\mathbf{w} \cdot \mathbf{t}_{T,i})\, q, \qquad \forall\, q \in \mathbb{P}_{r-3}(T), \tag{8}$$

$$\mathbf{t}_{T,i} \text{ three independent sides of } T,\ i = 1, 2, 3,$$

with $\mathcal{F}(T)$ the set of faces of T, are the dofs for a function $\mathbf{w} \in W^1_{h,r}(T)$. The norm of the vectors $\mathbf{t}_e, \mathbf{t}_{f,i}, \mathbf{t}_{T,i}$ is the length of the associated edge. We say that e, f, T are the *supports* of the dofs ξ_e, ξ_f, ξ_T.

Note that for $d = 2$, the dofs are given only by (6) and (7) substituting f with the triangle T; similarly, in the following, when $d = 2$, what concerns volumes should not be taken into account and what concerns faces f actually concerns the triangle T.

Remark 3.3. To make the computation of dofs easier , a *convenient choice for the polynomials* q spanning the polynomial spaces over (sub)simplices e, f, T that appear in Definition 3.2 is given by suitable products of the barycentric coordinates associated with the nodes of the considered (sub)simplex. The space $\mathbb{P}_\rho(S)$ of polynomials of degree $\leq \rho$ over a p-simplex S

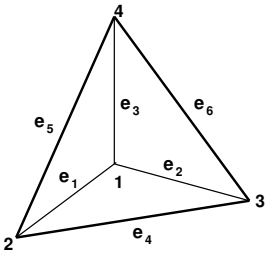


Figure 2: For the tetrahedron in the figure, the edges are $e_1 = \{1, 2\}$, $e_2 = \{1, 3\}$, $e_3 = \{1, 4\}$, $e_4 = \{2, 3\}$, $e_5 = \{2, 4\}$, $e_6 = \{3, 4\}$, the faces are $f_1 = \{2, 3, 4\}$, $f_2 = \{1, 3, 4\}$, $f_3 = \{1, 2, 4\}$, $f_4 = \{1, 2, 3\}$ (note that the face f_i is the one opposite the node i).

(i.e. a simplex of dimension $1 \leq p \leq d$) can be generated by the products $\lambda^{\mathbf{k}} = \prod_{i=1}^{p+1} (\lambda_{n_i})^{k_i}$, with $\mathbf{k} \in \mathcal{I}(p + 1, \rho)$ and n_i being the nodes of S.

The *classification* of dofs into edge-type, face-type, volume-type dofs can be done also for generators: volume-type generators contain (inside $\lambda^{\mathbf{k}}$ or $\mathbf{w}^e$) the barycentric coordinates w.r.t. all the nodes of a tetrahedron T, face-type generators contain the ones w.r.t. all and only the nodes of a face f, edge-type generators contain the ones w.r.t. only the nodes of an edge e. Note that face-type (resp. volume-type) generators appear for $r > 1$ (resp. $r > 2$) (and the same happens for face-type and volume-type dofs). See the explicit list of generators and dofs for the case $d = 3, r = 2$ in Example 1. It turns out that dofs ξ_e are 0 on face-type and volume-type generators, and dofs ξ_f are 0 on volume-type generators.

For the high order case ($r > 1$), the fields $\lambda^{\mathbf{k}}\mathbf{w}^e$ in Definition 3.1 are generators for $W^1_{h,r}(T)$, but some of the face-type or volume-type generators are *linearly dependent*. The selection of generators that constitute an actual basis of $W^1_{h,r}(T)$ can be guided by the dofs in Definition 3.2. More precisely, as face-type (resp. volume-type) generators keep the ones associated with the two (resp. three) edges e *chosen as* the two sides $\mathbf{t}_{f,1}, \mathbf{t}_{f,2}$ (resp. three sides $\mathbf{t}_{T,1}, \mathbf{t}_{T,2}, \mathbf{t}_{T,3}$) of face-type dofs (7) (resp. volume-type dofs (8)). A convenient choice of sides is described in Subsection 4.1 and is the one adopted in Example 1. One can check that the total number of dofs ξ_e, ξ_f, ξ_T in a simplex T is equal to $\dim(W^1_{h,r}(T)) = (r + d)(r + d - 1) \cdots (r + 2)r/(d - 1)!$.

The considered basis functions are not in *duality* with the dofs in Definition 3.2 when $r > 1$, namely, the matrix V with entries the weights $V_{ij} = \xi_i(\mathbf{w}_j)$, $1 \leq i, j \leq n_{\text{dofs}} = \dim(W^1_{h,r}(T))$ after a suitable renumbering of dofs, is not the identity matrix for $r > 1$. Duality can be re-established, if necessary, by considering new basis functions $\tilde{\mathbf{w}}_j$ built as *linear combinations* of the previous basis functions with coefficients given by the entries of V^{-1} [11]. The matrix V is a sort of generalized Vandermonde matrix. Note that V (and then V^{-1}) does not depend on the metric of the simplex T for which its entries are calculated. Indeed, first of all note that dofs in Definition 3.2 are conveniently normalized. Moreover, the $\xi_i(\mathbf{w}_j)$ are integrals of two addends of the type $\lambda^{\mathbf{k}'}\nabla\lambda_{n_i} \cdot \mathbf{t}_e$ (here $\lambda^{\mathbf{k}'}$ gathers the products of barycentric coordinates appearing in the basis functions and in q, and $\mathbf{t}_e$ stands also for $\mathbf{t}_{f,i}, \mathbf{t}_{T,i}$). Now, we have $\nabla\lambda_{n_i} \cdot \mathbf{t}_e = -1$ if n_i is the first node of e, $+1$ if it is its second node, 0 if it isn't a node of e; so, in the end, only terms of the type $\lambda^{\mathbf{k}'}$ survive in the integral and the value of $\xi_i(\mathbf{w}_j)$ can be calculated using the 'magic formula' (it is a classical result, see for instance [3]): if S is a p-simplex, $\frac{1}{|S|}\int_S \prod_{i=1}^{p+1}(\lambda_{n_i})^{k_i} = p!(\prod_{i=1}^{p+1} k_i!)/(p + \sum_{i=1}^{p+1} k_i)!$. This value is clearly independent of the metric of T. Moreover, the entries of V^{-1} turn out to be *integer* numbers. See Example 1 for the case $d = 3, r = 2$.

Example 1 (Generators, dofs, dualizing matrix for $d = 3$, $r = 2$)**.** If the edges and the faces of a tetrahedron are numbered as in Fig. 2, the basis functions are $\mathbf{w}_j = \lambda_{n_{r_j}}\mathbf{w}^{e_{s_j}}$, $1 \leq j \leq 20$, where the 12 edge-type basis functions have $(r_j)_{j=1}^{12} = (1, 2,\ 1, 3,\ 1, 4,\ 2, 3,\ 2, 4,\ 3, 4)$ and $(s_j)_{j=1}^{12} = (1, 1,\ 2, 2,\ 3, 3,\ 4, 4,\ 5, 5,\ 6, 6)$, and the 8 face-type basis functions have $(r_j)_{j=13}^{20} = (4, 3,\ 4, 3,\ 4, 2,\ 3, 2)$ and $(s_j)_{j=13}^{20} = (4, 5,\ 2, 3,\ 1, 3,\ 1, 2)$. Note that in order to get a basis, i.e. a set of linearly independent generators, we have chosen to eliminate the (face-type)

generators $\mathbf{w}_{21} = \lambda_{n_2}\mathbf{w}^{e_6}$, $\mathbf{w}_{22} = \lambda_{n_1}\mathbf{w}^{e_6}$, $\mathbf{w}_{23} = \lambda_{n_1}\mathbf{w}^{e_5}$, $\mathbf{w}_{24} = \lambda_{n_1}\mathbf{w}^{e_4}$. The corresponding edge-type dofs are:

$$\xi_1 \colon \mathbf{w} \mapsto \frac{1}{|e_1|}\int_{e_1}(\mathbf{w}\cdot\mathbf{t}_{e_1})\,\lambda_{n_1}, \quad \xi_2 \colon \mathbf{w} \mapsto \frac{1}{|e_1|}\int_{e_1}(\mathbf{w}\cdot\mathbf{t}_{e_1})\,\lambda_{n_2}, \ \ldots$$
$$\xi_{11} \colon \mathbf{w} \mapsto \frac{1}{|e_6|}\int_{e_6}(\mathbf{w}\cdot\mathbf{t}_{e_6})\,\lambda_{n_3}, \quad \xi_{12} \colon \mathbf{w} \mapsto \frac{1}{|e_6|}\int_{e_6}(\mathbf{w}\cdot\mathbf{t}_{e_6})\,\lambda_{n_4},$$

and the face-type dofs are:

$$\xi_{13} \colon \mathbf{w} \mapsto \frac{1}{|f_1|}\int_{f_1}(\mathbf{w}\cdot\mathbf{t}_{e_4}), \quad \xi_{14} \colon \mathbf{w} \mapsto \frac{1}{|f_1|}\int_{f_1}(\mathbf{w}\cdot\mathbf{t}_{e_5}), \ \ldots$$
$$\xi_{19} \colon \mathbf{w} \mapsto \frac{1}{|f_4|}\int_{f_4}(\mathbf{w}\cdot\mathbf{t}_{e_1}), \quad \xi_{20} \colon \mathbf{w} \mapsto \frac{1}{|f_4|}\int_{f_4}(\mathbf{w}\cdot\mathbf{t}_{e_2}).$$

For this ordering and choice of generators and dofs, the 'dualizing' matrix V^{-1} is

$$V^{-1} = \begin{bmatrix}
4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -4 & -2 & 2 & -2 & 2 & 4 & 8 & -4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & -4 & -2 & -4 & -2 & -4 & 8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -4 & -2 & 2 & -2 & 0 & 0 & 0 & 0 & 2 & 4 & 0 & 0 & 8 & -4 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & -2 & -4 & -2 & 0 & 0 & 0 & 0 & -4 & -2 & 0 & 0 & -4 & 8 & 0 & 0 & 0 & 0 \\
-4 & -2 & 0 & 0 & 2 & -2 & 0 & 0 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & -4 & 0 & 0 \\
2 & -2 & 0 & 0 & -4 & -2 & 0 & 0 & -4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 8 & 0 & 0 \\
-4 & -2 & 2 & -2 & 0 & 0 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & -4 \\
2 & -2 & -4 & -2 & 0 & 0 & -4 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 8
\end{bmatrix}.$$

4 Implementation of high order edge finite elements in FreeFem++

In general, to *add* a new finite element to FreeFem++, the user can write a C++ plugin that defines in a simplex the basis functions (and their derivatives), and an interpolation operator (which requires dofs and basis functions in *duality*). Indeed, in FreeFem++ the basis functions (and in some cases the coefficients of the interpolation operator) are constructed *locally*, i.e. in each simplex of $\mathcal{T}_h$, without the need of a transformation from the reference simplex. Note that the chosen definition of high order generators, which involves only the barycentric coordinates of the simplex, fits perfectly this local construction feature of FreeFem++. Nevertheless, the local construction should be done in such a way that the contributions coming from simplices sharing edges or faces can be then assembled properly inside the *global* matrix of the FE discretization. The strategy developed to deal with this issue for the high order edge elements is described in Subsection 4.1.The definition and the implementation of the interpolation operator are detailed in Subsection 4.2.

We added in this way the edge elements in 3d of degree $2, 3$ presented before. The code of the C++ plugin `Element_Mixte3d.cpp`, in which they are defined, is visible if FreeFem++ sources are downloaded (from `http://www.freefem.org/ff++/`) and is thus found in the folder `examples++-load`.

4.1 Local implementation strategy for the global assembling

The implementation of edge finite elements is quite delicate. Indeed, basis functions and dofs are associated with the *oriented* edges of mesh simplices: note that the low order $\mathbf{w}^e$

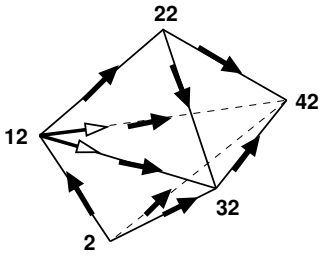


Figure 3: Orientation of edges ('filled' arrows) and choice of 2 edges ('empty' arrows) of the face shared by two adjacent tetrahedra using the numbering of mesh nodes.

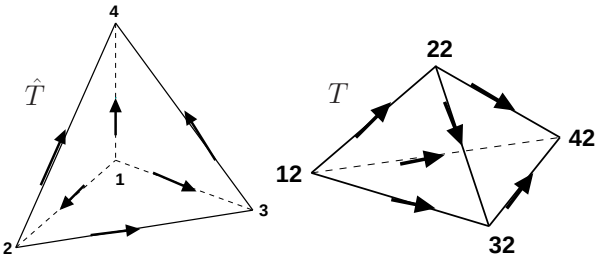


Figure 4: Using global numbers to examine edges and faces, the 'structure of orientation' of $T = \{12, 32, 42, 22\}$ is the one of $\hat{T} = \{1, 2, 3, 4\}$ up to a rotation.

and the high order $\lambda^{\mathbf{k}}\mathbf{w}^e$ generators change sign if the orientation of the edge e is reversed. Moreover, recall that for $r > 1$, in order to get a set of linearly independent generators, we also have to *choose* 2 edges for each face f. Here we wish to construct basis functions *locally*, i.e. in each simplex of $\mathcal{T}_h$, in such a way that the contributions coming from simplices sharing edges or faces could be assembled properly inside the *global* matrix of the FE discretization. For this purpose, it is essential to orient in the *same* way edges shared by simplices and to choose the *same* 2 edges for faces shared by adjacent tetrahedra. We have this need also to construct dofs giving the coefficients for the interpolation operator.

This need is satisfied using the *global numbers* of the mesh nodes (see Fig. 3). More precisely, to orient the edges e of the basis functions and the vectors $\mathbf{t}_e, \mathbf{t}_{f,i}, i = 1, 2, \mathbf{t}_{T,i}, i = 1, 2, 3$ of the dofs, we go from the node with the smallest global number to the node with the biggest global number. Similarly, to choose 2 edges per face for the face-type basis functions and dofs, we take the 2 edges going out from the node with the smallest global number in the face (and the 1st edge goes to the node with the 2nd smallest global number, the 2nd edge goes to the node with the biggest global number in the face).

Moreover, when we want basis functions $\tilde{\mathbf{w}}_j$ in duality with the dofs, a *second need* should be satisfied: we wish to use *for all mesh simplices* T the 'dualizing' coefficients of the matrix $\hat{V}^{-1}$ calculated, once for all, for the reference simplex $\hat{T}$ *with* a certain choice of orientation and choice of edges (recall that V^{-1} already does not depend on the metric of the simplex for which it is calculated). To be allowed to do this, it is sufficient to use the nodes *global numbers* to decide the order in which the non dual $\mathbf{w}_j$ (from which we start to then get the $\tilde{\mathbf{w}}_j$) are constructed locally on T. More precisely, for the edge-type (resp. face-type) basis functions the edges (resp. faces) are examined in the order written in the caption of Fig. 2, but replacing the nodes numbers $1, 2, 3, 4$ with the increasing global numbers of the nodes of T: the 1st examined edge is from the node with the 1st smallest global number to the one with the 2nd smallest global number, the 2nd examined edge is from the node with the 1st smallest global number to the one with the 3rd smallest global number, and so on, then the 1st examined face is the one opposite the node with the smallest global number, and so on. Indeed, in this way the first need is respected *and* the 'structure of orientation' of T is the one of $\hat{T}$ up to a rotation (see Fig. 4): then we are allowed to use the coefficients of $\hat{V}^{-1}$ for the linear combinations giving the $\tilde{\mathbf{w}}_j$.

Note that in 3d (resp. in 2d), to assemble the global linear system matrix, it is not essential which volume-type (resp. face-type) generators are chosen since they are not shared between tetrahedra (resp. triangles). On the contrary, also this choice is important when we want to use for all mesh simplices the coefficients of $\hat{V}^{-1}$ calculated for a simplex with a certain choice of orientation and choice of edges.

4.1.1 Implementation of the basis functions

To implement the strategy introduced to construct locally the basis functions $\tilde{\mathbf{w}}_j$ while respecting the two requirements just described, two *permutations* can be used; note that in this paragraph the numberings start from 0, and no more from 1, in order to comply with the C++ plugin written for the insertion in FreeFem++ of the new FE space. First, to construct the non dual $\mathbf{w}_j$, we define a permutation p_{d+1} of $d+1$ elements as follows: $p_{d+1}[i]$ is the local number (it takes values among $0, \ldots, d$) of the node with the i-th smallest global number in the simplex T, so we can say that p_{d+1} is the permutation for which the nodes of T are listed with increasing global number. For instance, for the tetrahedron $T = \{12, 32, 42, 22\}$ in Fig. 4, we have $p_4 = \{0, 3, 1, 2\}$. So, in the first step of construction of the $\mathbf{w}_j$, we replace each λ_i appearing in their expression with $\lambda_{p_{d+1}[i]}$. In the code of the FreeFem++ plugin, the permutation p_4 is called `perm`.

Then, in the second step of construction of the $\tilde{\mathbf{w}}_j$ as linear combinations of the $\mathbf{w}_j$, we use a permutation $P_{n_{\text{dofs}}}$ of $n_{\text{dofs}} = \dim(W^1_{h,r}(T))$ elements to go back to the local order of edges and faces. For instance for the tetrahedron $T = \{12, 32, 42, 22\}$, the order in which edges are examined in the first step is $\{\{12, 22\}, \{12, 32\}, \{12, 42\}, \{22, 32\}, \{22, 42\}, \{32, 42\}\}$, while the local order of edges would be $\{\{12, 32\}, \{12, 42\}, \{12, 22\}, \{32, 42\}, \{22, 32\}, \{22, 42\}\}$ (the local order is given by how the nodes of T are listed); similarly, the order in which faces are examined in the first step is $\{\{22, 32, 42\}, \{12, 32, 42\}, \{12, 22, 42\}, \{12, 22, 32\}\}$, while the local order of faces would be $\{\{22, 32, 42\}, \{12, 22, 42\}, \{12, 22, 32\}, \{12, 32, 42\}\}$. So for this tetrahedron, if $r = 2$ (for which there are 2 basis functions for each edge and 2 basis functions for each face, 20 basis functions in total listed in Example 1), we have

$$P_{20} = \{4, 5,\ 0, 1,\ 2, 3,\ 8, 9,\ 10, 11,\ 6, 7;\ 12, 13,\ 18, 19,\ 14, 15,\ 16, 17\},$$

(note that inside each edge or face the 2 related dofs remain ordered according to the global numbers). This permutation ($r = 2$) is built with the following code. There, `edgesMap` corresponds to a map that associates the pair $\{a, b\}$ of nodes of an edge e_i with its number $0 \le i \le 5$; this map is rather implemented with an array defined as `edgesMap`$[(a+1)(b+1)] = i$, where $(a+1)(b+1)$ results to be unique and symmetric for a pair (a, b), $0 \le a, b \le 3$, representing a tetrahedron edge.

```
int edgesMap[13] = {-1,-1,0,1,2,-1,3,-1,4,-1,-1,-1,5};
// static const int nvedge[6][2] = {{0,1},{0,2},{0,3},{1,2},{1,3},{2,3}};
int p20[20];
for(int i=0; i<6; ++i) // edge dofs
{
    int ii0 = Element::nvedge[i][0], ii1 = Element::nvedge[i][1];
    int i0 = perm[ii0]; int i1 = perm[ii1];
    int iEdge = edgesMap[(i0+1)*(i1+1)]; // i of the edge [i0,i1]
    p20[i*2] = iEdge*2;
    p20[i*2+1] = iEdge*2+1;
}
for(int j=0; j<4; ++j) // face dofs
{
    int jFace = perm[j];
    p20[12+j*2] = 12+jFace*2;
    p20[12+j*2+1] = 12+jFace*2+1;
}
```

Then, we will save the linear combinations of the $\mathbf{w}_\ell$, with coefficients given by the j-th column of $\hat{V}^{-1}$ (see Example 1), in the final basis functions $\tilde{\mathbf{w}}_{P_{20}[j]}$, thus in duality with the chosen dofs:

```
wtilde[p20[0]] = +4*w[0]-2*w[1]-4*w[16]+2*w[17]-4*w[18]+2*w[19];
wtilde[p20[1]] = -2*w[0]+4*w[1]-2*w[16]-2*w[17]-2*w[18]-2*w[19];
wtilde[p20[2]] = +4*w[2]-2*w[3]-4*w[14]+2*w[15]+2*w[18]-4*w[19];
wtilde[p20[3]] = -2*w[2]+4*w[3]-2*w[14]-2*w[15]-2*w[18]-2*w[19];
wtilde[p20[4]] = +4*w[4]-2*w[5]+2*w[14]-4*w[15]+2*w[16]-4*w[17];
wtilde[p20[5]] = -2*w[4]+4*w[5]-2*w[14]-2*w[15]-2*w[16]-2*w[17];
wtilde[p20[6]] = +4*w[6]-2*w[7]-4*w[12]+2*w[13]+2*w[18]-4*w[19];
wtilde[p20[7]] = -2*w[6]+4*w[7]-2*w[12]-2*w[13]+4*w[18]-2*w[19];
wtilde[p20[8]] = +4*w[8]-2*w[9]+2*w[12]-4*w[13]+2*w[16]-4*w[17];
wtilde[p20[9]] = -2*w[8]+4*w[9]-2*w[12]-2*w[13]+4*w[16]-2*w[17];
wtilde[p20[10]] = +4*w[10]-2*w[11]+2*w[12]-4*w[13]+2*w[14]-4*w[15];
wtilde[p20[11]] = -2*w[10]+4*w[11]+4*w[12]-2*w[13]+4*w[14]-2*w[15];
wtilde[p20[12]] = +8*w[12]-4*w[13];
wtilde[p20[13]] = -4*w[12]+8*w[13];
wtilde[p20[14]] = +8*w[14]-4*w[15];
wtilde[p20[15]] = -4*w[14]+8*w[15];
wtilde[p20[16]] = +8*w[16]-4*w[17];
wtilde[p20[17]] = -4*w[16]+8*w[17];
wtilde[p20[18]] = +8*w[18]-4*w[19];
wtilde[p20[19]] = -4*w[18]+8*w[19];
```

4.2 The interpolation operator

Duality of the basis functions with the dofs is needed in FreeFem++ to provide an *interpolation operator* onto a desired FE space of a function given by its analytical expression (or of a function belonging to another FE space). Indeed, if we define for a (vector) function $\mathbf{u}$ its finite element approximation $\mathbf{u}_h = \Pi_h(\mathbf{u})$ using the interpolation operator

$$\Pi_h \colon H(\mathrm{curl}, T) \to W^1_{h,r}(T), \quad \mathbf{u} \mapsto \mathbf{u}_h = \sum_{i=1}^{n_{\mathrm{dofs}}} c_i \tilde{\mathbf{w}}_i, \quad \text{with } c_i := \xi_i(\mathbf{u}), \tag{9}$$

we have that, if the duality property $\xi_j(\tilde{\mathbf{w}}_i) = \delta_{ij}$ holds, then $\xi_j(\mathbf{u}_h) = \sum_{i=1}^{n_{\mathrm{dofs}}} c_i \xi_j(\tilde{\mathbf{w}}_i) = c_j$. The interpolant coefficients $c_i = \xi_i(\mathbf{u})$ are computed in FreeFem++ with suitable quadrature formulas (on edges, faces or volumes) to approximate the values of the dofs in Definition 3.2 applied to $\mathbf{u}$.

Now, denote by g the whole integrand inside the dof expression, by n_{Qpts_i} the number of quadrature points of the suitable quadrature formula (on a segment, triangle or tetrahedron) to compute the integral (of precision high enough so that the integral is computed exactly when the dof is applied to a basis function), and by $\mathbf{x}_p$, a_p, $1 \le p \le n_{\mathrm{QF}_i}$ the quadrature points and their weights. Then we have

$$c_i = \xi_i(\mathbf{u}) = \sum_{p=1}^{n_{\mathrm{QF}_i}} a_p \, g(\mathbf{x}_p) = \sum_{p=1}^{n_{\mathrm{QF}_i}} a_p \sum_{j=1}^{d} \beta_j(\mathbf{x}_p) \, u_j(\mathbf{x}_p), \tag{10}$$

where for the second equality we have factorized $g(\mathbf{x}_p)$ in order to put in evidence the d components of $\mathbf{u}$, denoted by u_j, $1 \le j \le d$ (see the paragraph below).

Therefore, by substituting the expression of the coefficients (10) in the interpolation operator definition (9), we have the following expression of the interpolation operator

$$\Pi_h(\mathbf{u}) = \sum_{i=1}^{n_{\mathrm{dofs}}} \sum_{p=1}^{n_{\mathrm{QF}_i}} \sum_{j=1}^{d} a_p \, \beta_j(\mathbf{x}_p) \, u_j(\mathbf{x}_p) \, \tilde{\mathbf{w}}_i = \sum_{\ell=1}^{n_{\mathrm{ind}}} \alpha_\ell \, u_{j_\ell}(\mathbf{x}_{p_\ell}) \, \tilde{\mathbf{w}}_{i_\ell}, \tag{11}$$

where we have set α_ℓ equals each $a_p \, \beta_j(\mathbf{x}_p)$ for the right triple $(i, p, j) = (i_\ell, p_\ell, j_\ell)$. Indeed, a FreeFem++ plugin to introduce a new finite element (represented with a C++ class) should implement (11) by specifying the quadrature points, the indices i_ℓ (dof indices), p_ℓ (quadrature point indices), j_ℓ (component indices), which do not depend on the simplex and are defined in the class constructor, and the coefficients α_ℓ, which can depend on the simplex (if so, which is in particular the edge elements case, the α_ℓ are defined with the class function `set`).

4.2.1 Interpolation operator for $d = 3$, $r = 2$

We report here the code (extracted from the plugin `Element_Mixte3d.cpp` mentioned before) defining first the indices of (11) for the `Edge13d` finite element, i.e. for $d = 3$, $r = 2$. There `QFe`, `QFf` are the edge, resp. face, quadrature formulas, and `ne`=6, `nf`=4, are the number of edges, resp. faces, of the simplex (tetrahedron); we have $n_{\text{ind}} = d{\cdot}$`QFe.n`$\cdot 2$`ne`$+d{\cdot}$`QFf.n`$\cdot 2$`nf`. Note that in the code the numberings start from 0, and no more from 1.

```
int i=0, p=0, e=0; // i is l
for(e=0; e<(Element::ne)*2; e++) // 12 edge dofs
{
  if (e%2==1) {p = p-QFe.n;}
  // if true, the quadrature pts are the ones of the previous dof (same edge)
  for(int q=0; q<QFe.n; ++q,++p) // 2 edge quadrature pts
    for (int c=0; c<3; c++,i++) // 3 components
    {
      this->pInterpolation[i]=p; // p_l
      this->cInterpolation[i]=c; // j_l
      this->dofInterpolation[i]=e; // i_l
      this->coefInterpolation[i]=0.; // alfa_l (filled with the function set)
    }
}
for(int f=0; f<(Element::nf)*2; f++) // 8 face dofs
{
  if (f%2==1) {p = p-QFf.n;}
  // if true, the quadrature pts are the ones of the previous dof (same face)
  for(int q=0; q<QFf.n; ++q,++p) // 3 face quadrature pts
    for (int c=0; c<3; c++,i++) // 3 components
    {
      this->pInterpolation[i]=p; // p_l
      this->cInterpolation[i]=c; // j_l
      this->dofInterpolation[i]=e+f; // i_l
      this->coefInterpolation[i]=0.; // alfa_l (filled with the function set)
    }
}
```

Then, the coefficients α_ℓ are defined as follows. We start by writing (10) for one edge-type dof, with $e = \{n_1, n_2\}$:

$$c_i = \xi_i(\mathbf{u}) = \frac{1}{|e|}\int_e (\mathbf{u}\cdot\mathbf{t}_e)\,\lambda_{n_1} = \sum_{p=1}^{\texttt{QFe.n}} a_p\,(\mathbf{u}(\mathbf{x}_p)\cdot\mathbf{t}_e)\,\lambda_{n_1}(\mathbf{x}_p)$$
$$= \sum_{p=1}^{\texttt{QFe.n}} a_p \sum_{j=1}^{d} u_j(\mathbf{x}_p)(x_{n_2 j} - x_{n_1 j})\lambda_{n_1}(\mathbf{x}_p)$$

so $\beta_j(\mathbf{x}_p) = (x_{n_2 j} - x_{n_1 j})\lambda_{n_1}(\mathbf{x}_p)$ and $\alpha_\ell = a_{p_\ell}\,\beta_{j_\ell}(\mathbf{x}_{p_\ell}) = (x_{n_2 j_\ell} - x_{n_1 j_\ell})\,a_{p_\ell}\,\lambda_{n_1}(\mathbf{x}_{p_\ell})$. Similarly for one face-type dof, with $f = \{n_1, n_2, n_3\}$, $e = \{n_1, n_2\}$:

$$c_i = \xi_i(\mathbf{u}) = \frac{1}{|f|}\int_f (\mathbf{u}\cdot\mathbf{t}_e) = \sum_{p=1}^{\texttt{QFf.n}} a_p\,(\mathbf{u}(\mathbf{x}_p)\cdot\mathbf{t}_e) = \sum_{p=1}^{\texttt{QFf.n}} a_p \sum_{j=1}^{d} u_j(\mathbf{x}_p)(x_{n_2 j} - x_{n_1 j})$$

so $\beta_j(x_p) = (x_{n_2 j} - x_{n_1 j})$ and $\alpha_\ell = a_{p_\ell}\,\beta_{j_\ell}(\mathbf{x}_{p_\ell}) = (x_{n_2 j_\ell} - x_{n_1 j_\ell})\,a_{p_\ell}$. The code that generalizes this calculations for all the dofs is the following, extracted from the function `set` of the plugin (note that also here we have to pay particular attention to the orientation and choice issues).

```
int i=0, p=0;
for(int ee=0; ee<Element::ne; ee++) // loop on the edges
{
  R3 E=K.Edge(ee);
  int eo = K.EdgeOrientation(ee);
  if(!eo) E=-E;
    for(int edof=0; edof<2; edof++) // 2 dofs for each edge
    {
```

```
      if (edof==1) {p = p-QFe.n;}
      for(int q=0; q<QFe.n; ++q,++p)
      {
        double ll=QFe[q].x; // value of lambda_0 or lambda_1
        if( (edof+eo) == 1 ) ll = 1-ll;
        for(int c=0; c<3; c++,i++)
        {
          M.coef[i] = E[c]*QFe[q].a*ll;
        }
      }
    }
}
for(int ff=0; ff<Element::nf; ff++) // loop on the faces
{
  const Element::Vertex * fV[3] = {& K.at(Element::nvface[ff][0]), ...
  //  (one unique line with the following)
  ... & K.at(Element::nvface[ff][1]), & K.at(Element::nvface[ff][2])};
  int i0=0, i1=1, i2=2;
  if(fV[i0]>fV[i1]) Exchange(i0,i1);
    if(fV[i1]>fV[i2]) { Exchange(i1,i2);
      if(fV[i0]>fV[i1]) Exchange(i0,i1); }
  // now local numbers in the tetrahedron:
  i0 = Element::nvface[ff][i0], i1 = Element::nvface[ff][i1], ...
  ... i2 = Element::nvface[ff][i2];
  for(int fdof=0; fdof<2; ++fdof) // 2 dofs for each face
  {
    int ie0=i0, ie1 = fdof==0? i1 : i2;
    // edge for the face dof (its endpoints local numbers)
    R3 E(K[ie0],K[ie1]);
    if (fdof==1) {p = p-QFf.n;}
    for(int q=0; q<QFf.n; ++q,++p) // loop on the 3 face quadrature pts
      for (int c=0; c<3; c++,i++) // loop on the 3 components
      {
        M.coef[i] = E[c]*QFf[q].a;
      }
  }
}
```

Example 2 (Using the new FEs in a FreeFem++ script)**.** The edge elements in 3d of degree $2, 3$ can be used (since FreeFem++ version 3.44) by loading in the `edp` script the plugin (`load "Element_Mixte3d"`), and using the keywords `Edge13d`, `Edge23d` respectively. The edge elements of the lowest degree 1 were already available and called `Edge03d`. After generating a tetrahedral mesh `Th`, complex vector functionx `E`, `v` in, e.g., the `Edge03d` space on `Th` are declared with the commands:

```
fespace Vh(Th,Edge03d);   Vh<complex> [Ex,Ey,Ez], [vx,vy,vz];
```

Then the weak formulation (4) of the problem is naturally transcribed as:

```
macro Curl(ux,uy,uz) [dy(uz)-dz(uy),dz(ux)-dx(uz),dx(uy)-dy(ux)] // EOM
macro Nvec(ux,uy,uz) [uy*N.z-uz*N.y,uz*N.x-ux*N.z,ux*N.y-uy*N.x] // EOM

problem waveguide([Ex,Ey,Ez], [vx,vy,vz], solver=sparsesolver) =
                  int3d(Th)(Curl(Ex,Ey,Ez)'*Curl(vx,vy,vz))
                - int3d(Th)(gamma^2*[Ex,Ey,Ez]'*[vx,vy,vz])
                + int2d(Th,in,out)(1i*eta*Nvec(Ex,Ey,Ez)'*Nvec(vx,vy,vz))
                - int2d(Th,in)([vx,vy,vz]'*[Gix,Giy,Giz])
                + on(guide,Ex=0,Ey=0,Ez=0);
```

See more details in the example `waveguide.edp` available in `examples++-load` folder of every FreeFem++ distribution. In FreeFem++ the interpolation operator is simply called with the `=` symbol: for example one can define analytical functions `func f1 = 1+x+2*y+3*z;` `func f2 = -1-x-2*y+2*z;` `func f3 = 2-2*x+y-2*z;` and call `[Ex,Ey,Ez]=[f1,f2,f3];`.

5 Overlapping Schwarz preconditioners

As shown numerically in [3], the matrix of the linear system resulting from the described high order discretization is ill conditioned. So, when using iterative solvers (GMRES in our case), preconditioning becomes necessary, and here we choose overlapping Schwarz domain decomposition preconditioners.

Consider a decomposition of the domain Ω into N_{sub} *overlapping* subdomains Ω_s that consist of a union of simplices of the mesh $\mathcal{T}_h$. In order to describe the matrices appearing in the algebraic expression of the preconditioners, let $\mathcal{N}$ be an ordered set of the degrees of freedom of the whole domain, and let $\mathcal{N} = \bigcup_{s=1}^{N_{\text{sub}}} \mathcal{N}_s$ be its decomposition into the (non disjoint) ordered subsets corresponding to the different (overlapping) subdomains Ω_s: a degree of freedom belongs to $\mathcal{N}_s$ if its support (edge, face or volume) is contained in Ω_s. For edge finite elements, it is important to ensure that the *orientation* of the degrees of freedom is the same in the domain and in the subdomains. Define the matrix R_s as the *restriction* matrix from Ω to the subdomain Ω_s: it is a $\#\mathcal{N}_s \times \#\mathcal{N}$ Boolean matrix, whose (i, j) entry equals 1 if the i-th degree of freedom in $\mathcal{N}_s$ is the j-th one in $\mathcal{N}$. Note that the *extension* matrix from the subdomain Ω_s to Ω is given by R_s^T. To deal with the unknowns that belong to the overlap between subdomains, define for each subdomain a $\#\mathcal{N}_s \times \#\mathcal{N}_s$ diagonal matrix D_s that gives a discrete *partition of unity*, i.e.

$$\sum_{s=1}^{N_{\text{sub}}} R_s^T D_s R_s = I.$$

Then the *Optimized Restricted Additive Schwarz* (ORAS) preconditioner can be expressed as

$$M_{\text{ORAS}}^{-1} = \sum_{s=1}^{N_{\text{sub}}} R_s^T D_s A_s^{-1} R_s, \tag{12}$$

where the matrices A_s are the local matrices of the *subproblems* with impedance boundary conditions $(\nabla \times \mathbf{E}) \times \mathbf{n} + \mathrm{i}\tilde{\omega}\mathbf{n} \times (\mathbf{E} \times \mathbf{n})$ as transmission conditions at the interfaces between subdomains (note that in this section the term 'local' refers to a subdomain and not to a mesh simplex). These local matrices stem from the discretization of the considered Maxwell's equation by high order finite elements introduced in the previous sections. While the term 'restricted' refers to the presence of the partition of unity matrices D_s, the term 'optimized' refers to the use of impedance boundary conditions (with parameter $\eta = \tilde{\omega}$ in (3)) as transmission conditions, proposed by Després in [20]. The algebraic formulation of optimized Schwarz methods, of the type of (12), was introduced in [31].

The implementation of the partition of unity in FreeFem++ is described in [32]: suitable piecewise linear functions χ_s giving a continuous partition of unity ($\sum_{s=1}^{N_{\text{sub}}} \chi_s = 1$) are interpolated at the barycenters of the support (edge, face, volume) of each dof of the (high order) edge finite elements. This interpolation is obtained thanks to an auxiliary FreeFem++ *scalar* FE space (`Edge03ds0`, `Edge13ds0`, `Edge23ds0`) that has only the interpolation operator and no basis functions, available in the plugin `Element_Mixte3d` mentioned before. When impedance conditions are chosen as transmission conditions at the interfaces, it is essential that not only the function χ_s but also its derivative are equal to zero on the border of the subdomain Ω_s. Indeed, if this property is satisfied, the continuous version of the ORAS algorithm is equivalent to P. L. Lions' algorithm (see [33] §2.3.2).

6 Numerical experiments

We validate the ORAS preconditioner (12) for different values of physical and numerical parameters, and compare it with a symmetric variant without the partition of unity (called Optimized Additive Schwarz):

$$M_{\text{OAS}}^{-1} = \sum_{s=1}^{N_{\text{sub}}} R_s^T A_s^{-1} R_s.$$

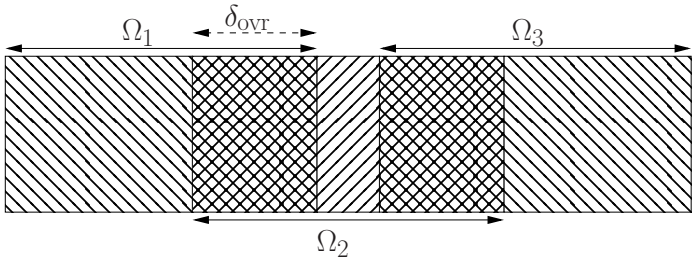


Figure 5: The stripwise decomposition of the two-dimensional domain.

k	N_{dofs}	N_{iterNp}	N_{iter}	$\max\lvert\lambda-(1,0)\rvert$	$\#\{\lambda\in\mathbb{C}\setminus\bar{\mathcal{D}}_1\}$	$\#\{\lambda\in\partial\mathcal{D}_1\}$
0	282	179	5(10)	1.04e−1(1.38e+1)	0(4)	0(12)
1	884	559	6(15)	1.05e−1(1.63e+1)	0(8)	0(40)
2	1806	1138	6(17)	1.05e−1(1.96e+1)	0(12)	0(84)
3	3048	1946	6(21)	1.05e−1(8.36e+2)	0(16)	0(144)
4	4610	2950	6(26)	1.05e−1(1.57e+3)	0(20)	0(220)

Table 1: Influence of the polynomial degree $r = k + 1$ on the convergence of ORAS(OAS) preconditioner for $\omega = \omega_2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 2h$.

The numerical experiments are performed for a waveguide configuration in $2d$ and then in $3d$.

6.1 Results for the two-dimensional problem

We present the results obtained for a two-dimensional waveguide with $\mathtt{c} = 0.0502\,\text{m}$, $\mathtt{b} = 0.00254\,\text{m}$, with the physical parameters: $\varepsilon = 8.85 \cdot 10^{-12}\,\text{F m}^{-1}$, $\mu = 1.26 \cdot 10^{-6}\,\text{H m}^{-1}$ and $\sigma = 0.15\,\text{S m}^{-1}$. We consider three angular frequencies $\omega_1 = 16\,\text{GHz}$, $\omega_2 = 32\,\text{GHz}$, and $\omega_3 = 64\,\text{GHz}$, varying the mesh size h according to the relation $h^2 \cdot \tilde{\omega}^3 = 2$ (in [34] it was proved that this relation avoids pollution effects for the one-dimensional Helmholtz equation).

Note that in 2d the function $\mathbf{E}_{\text{ex}} = (0, e^{-\mathtt{i}\gamma x})$ verifies the equation, the metallic boundary conditions on Γ_{w}, and the impedance boundary conditions on Γ_{in}, Γ_{out} with parameter $\eta = \tilde{\omega}$ and $\mathbf{g}^{\text{in}} = (\mathtt{i}\gamma + \mathtt{i}\tilde{\omega})\mathbf{E}_{\text{ex}}$ and $\mathbf{g}^{\text{out}} = (-\mathtt{i}\gamma + \mathtt{i}\tilde{\omega})\mathbf{E}_{\text{ex}}$; when $\sigma = 0$ we get $\mathbf{g}^{\text{in}} = 2\mathtt{i}\tilde{\omega}\mathbf{E}_{\text{ex}}$ and $\mathbf{g}^{\text{out}} = \mathbf{0}$. The real part of the propagation constant $-\mathtt{i}\gamma$ gives the rate at which the amplitude changes as the wave propagates, which corresponds to wave dissipation (note that if $\sigma > 0$, $\Re(-\mathtt{i}\gamma) < 0$, while if $\sigma = 0$, $\Re(-\mathtt{i}\gamma) = 0$). A numerical study about the order of (h- and r-) convergence with respect to the exact solution of the high order finite element method can be found in [35].

Here we solve the linear system resulting from the finite element discretization with GMRES (with a stopping criterion based on the relative residual and a tolerance of 10^{-6}), starting with a *random* initial guess, which ensures, unlike a zero initial guess, that all frequencies are present in the error. We compare the ORAS and OAS preconditioners, taking a stripwise subdomains decomposition, along the wave propagation, as shown in Fig. 5.

To study the convergence of GMRES preconditioned by ORAS or OAS we vary first the polynomial degree $r = k + 1$ (Table 1, Figs. 6–7), then the angular frequency ω (Table 2, Figs. 8–9), the number of subdomains N_{sub} (Table 3, Figs. 10–11) and finally the overlap size δ_{ovr} (Table 4, Figs. 12–13). Here, $\delta_{\text{ovr}} = 1h, 2h, 4h$ means that we consider a total overlap between two subdomains of $1, 2, 4$ mesh triangles along the horizontal direction (see Fig. 5).

In Tables 1–4, N_{dofs} is the total number of degrees of freedom, N_{iterNp} is the number of iterations necessary to attain the prescribed convergence for GMRES without any preconditioner, and N_{iter} is the number of iterations for GMRES preconditioned by ORAS (OAS).

ω	N_{dofs}	N_{iterNp}	N_{iter}	$\max\lvert\lambda-(1,0)\rvert$	$\#\{\lambda\in\mathbb{C}\setminus\bar{\mathcal{D}}_1\}$	$\#\{\lambda\in\partial\mathcal{D}_1\}$
ω_1	339	232	5(11)	2.46e−1(1.33e+1)	0(6)	0(45)
ω_2	1806	1138	6(17)	1.05e−1(1.96e+1)	0(12)	0(84)
ω_3	7335	4068	9(24)	3.03e−1(2.73e+1)	0(18)	0(123)

Table 2: Influence of the angular frequency ω on the convergence of ORAS(OAS) preconditioner for $k=2$, $N_{\text{sub}}=2$, $\delta_{\text{ovr}}=2h$.

N_{sub}	N_{iter}	$\max\lvert\lambda-(1,0)\rvert$	$\#\{\lambda\in\mathbb{C}\setminus\bar{\mathcal{D}}_1\}$	$\#\{\lambda\in\partial\mathcal{D}_1\}$
2	6(17)	1.05e−1(1.96e+1)	0(12)	0(84)
4	10(27)	5.33e−1(1.96e+1)	0(38)	0(252)
8	19(49)	7.73e−1(1.96e+1)	0(87)	0(588)

Table 3: Influence of the number of subdomains N_{sub} on the convergence of ORAS(OAS) preconditioner for $k=2$, $\omega=\omega_2$, $\delta_{\text{ovr}}=2h$.

Moreover, denoting by

$$\mathcal{D}_1=\{z\in\mathbb{C}:|z-z_0|<1\}$$

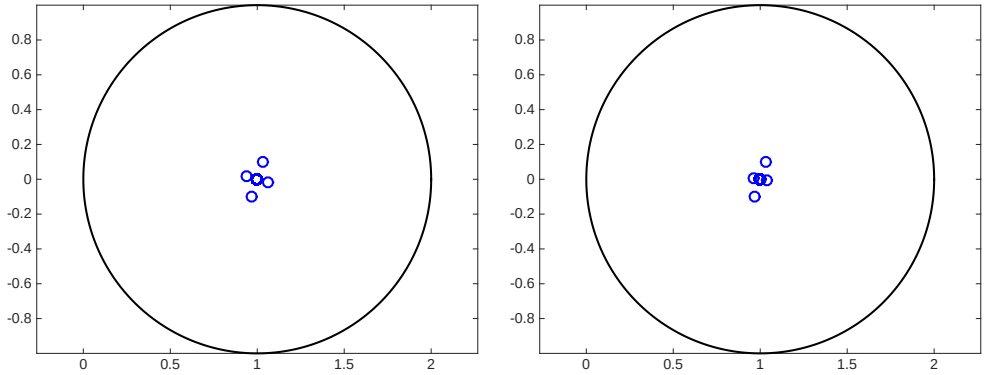
the unit disk centered at $z_0=(1,0)$ in the complex plane, we measure also the maximum distance to $(1,0)$ of the eigenvalues λ of the preconditioned matrix, the number of eigenvalues that have distance greater than 1, and the number of eigenvalues that have distance equal to 1 (up to a tolerance of 10^{-10}). This information is useful to characterize the convergence. Indeed, if A is the matrix of the system to solve and M^{-1} is the domain decomposition preconditioner, then $I-M^{-1}A$ is the iteration matrix of the domain decomposition method used as an iterative solver. So, a measure of the convergence of the domain decomposition solver would be to check whether the eigenvalues of the preconditioned matrix $M^{-1}A$ are contained in $\mathcal{D}_1$. When the domain decomposition method is used, like here, as a preconditioner, the distribution of the spectrum remains a good indicator of the convergence. Note that the matrix of the linear system doesn't change when N_{sub} or δ_{ovr} vary, therefore in Tables 3–4 (where $k=2$, $\omega=\omega_2$) we don't report $N_{\text{dofs}}=1806$ and $N_{\text{iterNp}}=1138$ again. In all Tables 1–4, we don't mention the condition number of the preconditioned matrix: indeed, no convergence rate estimates in terms of the condition number of the matrix, as those we are used to with the conjugate gradient method, are available for the GMRES method.

Figs. 6, 8, 10, 12, respectively Figs. 7, 9, 11, 13, show the whole spectrum in the complex plane of the matrix preconditioned by ORAS, respectively by OAS (note that many eigenvalues are multiple), together with $\partial\mathcal{D}_1$.

Looking at the tables and figures, we can see that the non preconditioned GMRES method is very slow, and the ORAS preconditioner gives much faster convergence than the OAS preconditioner. As expected, the convergence becomes slower when ω or N_{sub} increase, or when δ_{ovr} decreases. In these tests, when varying k (which gives the polynomial degree $r=k+1$ of the FE basis functions), the number of iterations for convergence using the ORAS preconditioner is equal to 5 for $k=0$ and then it stays equal to 6 for $k>0$; this is

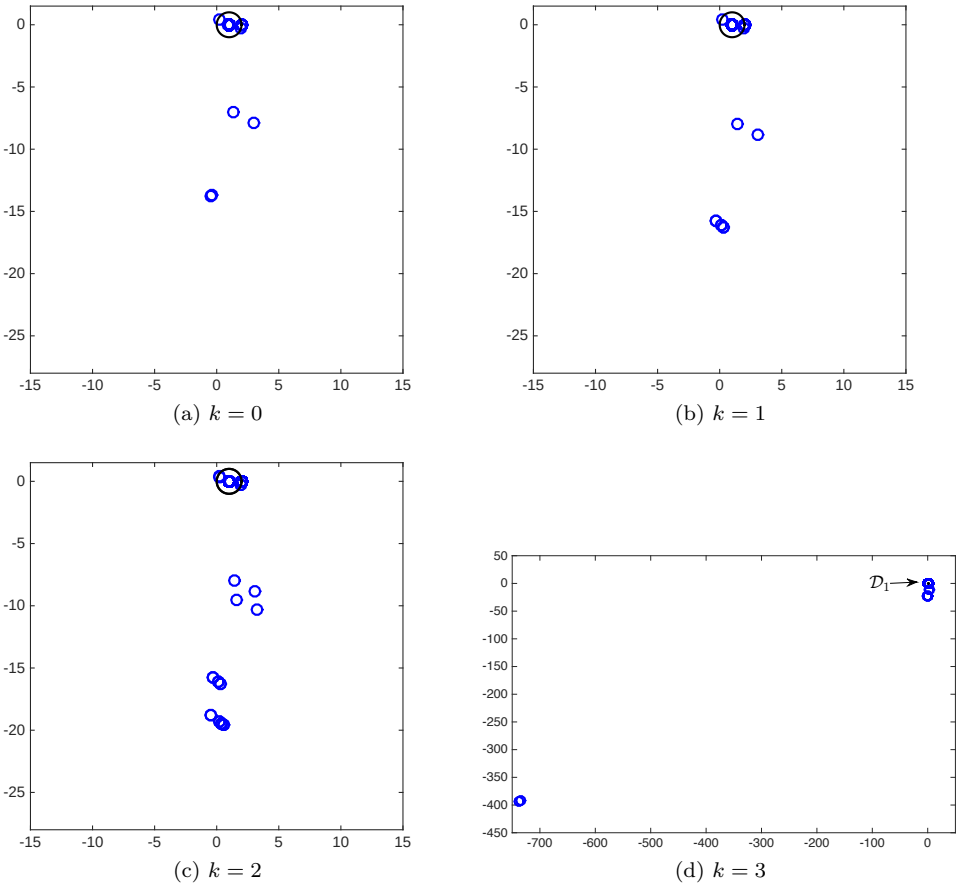
δ_{ovr}	N_{iter}	$\max\lvert\lambda-(1,0)\rvert$	$\#\{\lambda\in\mathbb{C}\setminus\bar{\mathcal{D}}_1\}$	$\#\{\lambda\in\partial\mathcal{D}_1\}$
$1h$	10(20)	1.95e+1(1.96e+1)	3(12)	0(39)
$2h$	6(17)	1.05e−1(1.96e+1)	0(12)	0(84)
$4h$	5(14)	1.06e−1(1.96e+1)	0(12)	0(174)

Table 4: Influence of the overlap size δ_{ovr} on the convergence of ORAS(OAS) preconditioner for $k=2$, $\omega=\omega_2$, $N_{\text{sub}}=2$.



(a) $k = 0$ (b) $k = 1, 2, 3$

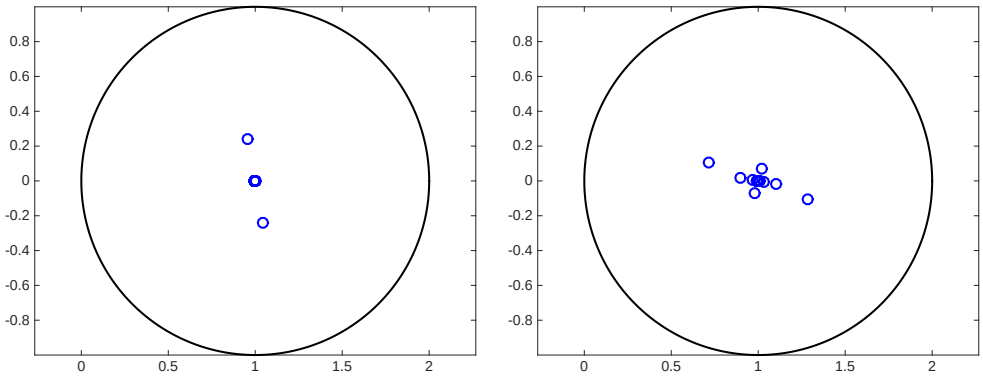
Figure 6: Influence of the polynomial degree $r = k + 1$ on the spectrum of the ORAS-preconditioned matrix for $\omega = \omega_2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 2h$.



(a) $k = 0$ (b) $k = 1$

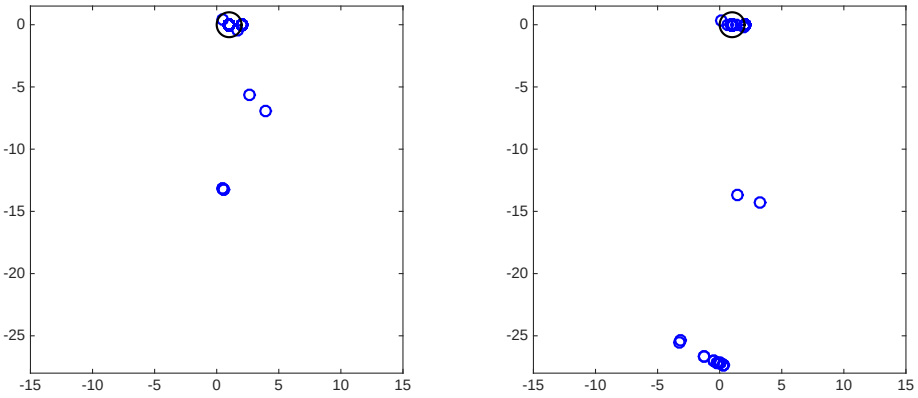
(c) $k = 2$ (d) $k = 3$

Figure 7: Influence of the polynomial degree $r = k + 1$ on the spectrum of the OAS-preconditioned matrix for $\omega = \omega_2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 2h$.



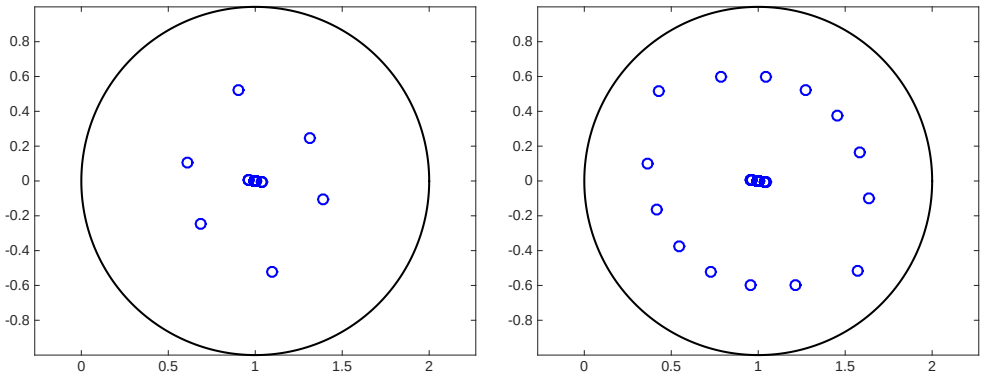
(a) $\omega = \omega_1$ (b) $\omega = \omega_3$

Figure 8: Influence of the angular frequency ω on the spectrum of the ORAS-preconditioned matrix for $k = 2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 2h$.



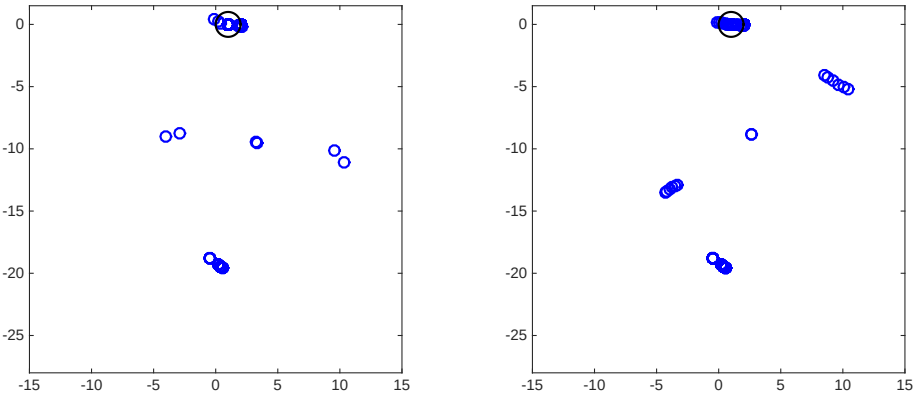
(a) $\omega = \omega_1$ (b) $\omega = \omega_3$

Figure 9: Influence of the angular frequency ω on the spectrum of the OAS-preconditioned matrix for $k = 2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 2h$.



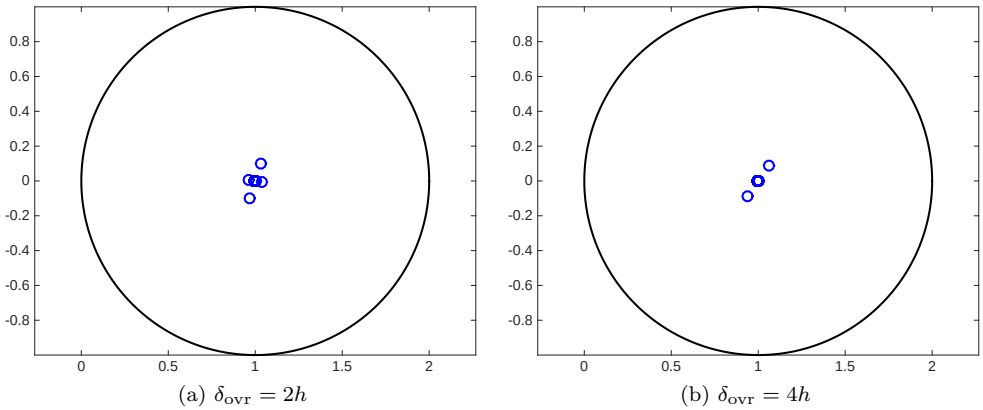
(a) $N_{\text{sub}} = 4$ (b) $N_{\text{sub}} = 8$

Figure 10: Influence of the number of subdomains N_{sub} on the spectrum of the ORAS-preconditioned matrix for $k = 2$, $\omega = \omega_2$, $\delta_{\text{ovr}} = 2h$.



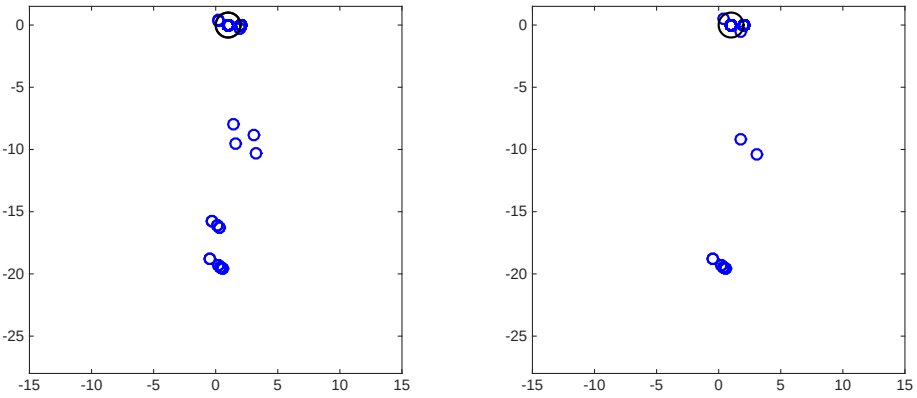
(a) $N_{\text{sub}} = 4$ (b) $N_{\text{sub}} = 8$

Figure 11: Influence of the number of subdomains N_{sub} on the spectrum of the OAS-preconditioned matrix for $k = 2$, $\omega = \omega_2$, $\delta_{\text{ovr}} = 2h$.



(a) $\delta_{\text{ovr}} = 2h$ (b) $\delta_{\text{ovr}} = 4h$

Figure 12: Influence of the overlap size δ_{ovr} on the spectrum of the ORAS-preconditioned matrix for $k = 2$, $\omega = \omega_2$, $N_{\text{sub}} = 2$.



(a) $\delta_{\text{ovr}} = 2h$ (b) $\delta_{\text{ovr}} = 4h$

Figure 13: Influence of the overlap size δ_{ovr} on the spectrum of the OAS-preconditioned matrix for $k = 2$, $\omega = \omega_2$, $N_{\text{sub}} = 2$.

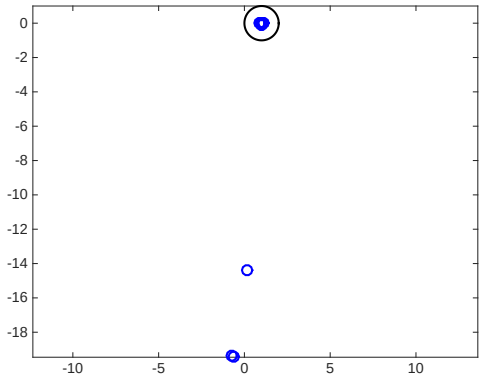


Figure 14: The spectrum of the ORAS-preconditioned matrix for $k = 2$, $\omega = \omega_2$, $N_{\text{sub}} = 2$, $\delta_{\text{ovr}} = 1h$.

reflected by the corresponding spectra in Fig. 6, which indeed remain quite similar when k varies.

Note also that, when using the ORAS preconditioner, for 2 subdomains the spectrum is always well clustered inside the unit disk, except for the case with $\delta_{\text{ovr}} = 1h$ (see Fig. 14), in which 3 eigenvalues are outside with distances from $(1, 0)$ equal to $19.5, 19.4, 14.4$. This case $\delta_{\text{ovr}} = 1h$ corresponds to adding a layer of simplices just to one of the two non overlapping subdomains to obtain the overlapping decomposition; hence it appears necessary to add at least one layer from both subdomains. Then we see that for 4 and 8 subdomains the spectrum becomes less well clustered. With the OAS preconditioner there are always eigenvalues outside the unit disk. For all the considered cases, we see that the less clustered the spectrum, the slower the convergence.

6.2 Results for the three-dimensional problem

We complete the presentation showing some results for the full 3d simulation, for a waveguide of dimensions $\mathtt{c} = 0.1004\,\text{m}$, $\mathtt{b} = 0.00508\,\text{m}$, and $\mathtt{a} = 0.01016\,\text{m}$. The physical parameters are: $\varepsilon = 8.85 \cdot 10^{-12}\,\text{F}\,\text{m}^{-1}$, $\mu = 1.26 \cdot 10^{-6}\,\text{H}\,\text{m}^{-1}$ and $\sigma = 0.15\,\text{S}\,\text{m}^{-1}$ or $\sigma = 0\,\text{S}\,\text{m}^{-1}$. We take a stripwise subdomains decomposition along the wave propagation, with $\delta_{\text{ovr}} = 2h$; however, note that in FreeFem++ very general subdomains decompositions can be considered.

In 3d, if $\sigma = 0$ there is an exact solution given by the Transverse Electric (TE) modes:

$$
\begin{aligned}
E_x^{TE} &= 0, \\
E_y^{TE} &= -\mathcal{C}\frac{m\pi}{\mathtt{a}} \sin\left(\frac{m\pi z}{\mathtt{a}}\right) \cos\left(\frac{n\pi y}{\mathtt{b}}\right) e^{-\mathtt{i}\beta x}, \\
E_z^{TE} &= \mathcal{C}\frac{n\pi}{\mathtt{b}} \cos\left(\frac{m\pi z}{\mathtt{a}}\right) \sin\left(\frac{n\pi y}{\mathtt{b}}\right) e^{-\mathtt{i}\beta x}, \qquad m, n \in \mathbb{N}.
\end{aligned}
$$

The real constant β is linked to the waveguide dimensions $\mathtt{a}, \mathtt{b}$ by the so called dispersion relation $\left(\frac{m\pi}{\mathtt{a}}\right)^2 + \left(\frac{n\pi}{\mathtt{b}}\right)^2 = \tilde{\omega}^2 - \beta^2$, and we choose $\mathcal{C} = \mathtt{i}\omega\mu/(\tilde{\omega}^2 - \beta^2)$. The field $\mathbf{E}^{TE}$ satisfies the metallic boundary conditions on Γ_{w} and the impedance boundary conditions on Γ_{in}, Γ_{out} with parameter $\eta = \beta$ and $\mathbf{g}^{\text{in}} = (\mathtt{i}\beta + \mathtt{i}\beta)\mathbf{E}^{TE} = 2\mathtt{i}\beta\mathbf{E}^{TE}$ and $\mathbf{g}^{\text{out}} = (-\mathtt{i}\beta + \mathtt{i}\beta)\mathbf{E}^{TE} = \mathbf{0}$.

Since the propagation constant in 3d is β and no more $\tilde{\omega}$, we compute the mesh size h using the relation $h^2 \cdot \beta^3 = 1$, taking $\beta = \omega_\beta\sqrt{\mu\varepsilon}$, with $\omega_\beta = 32\,\text{GHz}$. Then the dispersion relation gives $\tilde{\omega} = \sqrt{\beta^2 + (m\pi/\mathtt{a})^2 + (n\pi/\mathtt{b})^2}$ (where we choose $m = 1, n = 0$), and we get $\omega = \tilde{\omega}/\sqrt{\mu\varepsilon}$.

Again, the linear system is solved with preconditioned GMRES, with a stopping criterion based on the relative residual and a tolerance of 10^{-6}, starting with a random initial guess. To apply the preconditioner, the local problems in each subdomain of matrices A_s are solved with the direct solver MUMPS [36].

k	N_{dofs}	N_{iter}	N_{sub}	N_{dofs}	N_{iter}
0	62283	8(40)	2	324654	8(70)
1	324654	8(70)	4	324654	11(106)
2	930969	8(99)	8	324654	17(168)

Table 5: Results in 3d, $\sigma = 0.15\,\mathrm{S\,m^{-1}}$: influence of the polynomial degree $r = k + 1$ (for $N_{\text{sub}} = 2$), and of the number of subdomains N_{sub} (for $k = 1$), on the convergence of ORAS(OAS) preconditioner ($\beta = \omega_\beta \sqrt{\mu\varepsilon}$ with $\omega_\beta = 32\,\mathrm{GHz}$, $\delta_{\text{ovr}} = 2h$).

k	N_{dofs}	N_{iter}	N_{sub}	N_{dofs}	N_{iter}
0	62283	7(40)	2	324654	8(67)
1	324654	8(67)	4	324654	13(114)
2	930969	8(97)	8	324654	23(201)

Table 6: Results in 3d, $\sigma = 0\,\mathrm{S\,m^{-1}}$: influence of the polynomial degree $r = k + 1$ (for $N_{\text{sub}} = 2$), and of the number of subdomains N_{sub} (for $k = 1$), on the convergence of ORAS(OAS) preconditioner ($\beta = \omega_\beta \sqrt{\mu\varepsilon}$ with $\omega_\beta = 32\,\mathrm{GHz}$, $\delta_{\text{ovr}} = 2h$).

In Tables 5, 6 we show the number of iterations for convergence, for the problem with $\sigma = 0.15\,\mathrm{S\,m^{-1}}$ and $\sigma = 0\,\mathrm{S\,m^{-1}}$ respectively, varying first the polynomial degree $r = k + 1$ (for $N_{\text{sub}} = 2$), and then the number of subdomains N_{sub} (for $k = 1$). Like in the 2d case, the number of iterations using the ORAS preconditioner does not vary with the polynomial degree of the FE basis functions, while using the OAS preconditioner it varies and is much higher. Again, the convergence becomes slower when the number of subdomains increases, both with ORAS and OAS. We see that for more than 2 subdomains the number of iterations for the non dissipative problem ($\sigma = 0$) is higher than for the problem with $\sigma = 0.15\,\mathrm{S\,m^{-1}}$.

In Figure 15 we plot the norm of the real part of the solution, which decreases as the wave propagates since there $\sigma = 0.15\,\mathrm{S\,m^{-1}}$ is different from zero.

7 Conclusion

We have adopted a friendly definition of high order edge elements generators and degrees of freedom: both in 2d and 3d their expression is rather simple, and the generators are strictly connected with the degrees of freedom. Their presentation is enriched with illustrative examples, and an operational strategy of implementation of these elements is described in detail. The elements in 3d of polynomial degree $1, 2, 3$ are available in FreeFem++ (since version 3.44), loading the plugin `load "Element_Mixte3d"` and building the finite element space `fespace` with the keywords `Edge03d`, `Edge13d`, `Edge23d` respectively.

Numerical experiments have shown that Schwarz preconditioning significantly improves GMRES convergence for different values of physical and numerical parameters, and that the ORAS preconditioner always performs much better than the OAS preconditioner. Indeed, the only advantage of the OAS method is to preserve symmetry for symmetric problems: that is why it should be used only for symmetric positive definite matrices as a preconditioner for the conjugate gradient method. Moreover, in all the considered test cases, the number of iterations for convergence using the ORAS preconditioner does not vary when the polynomial degree of the adopted high order finite elements increases. We have also seen that it is necessary to take an overlap of at least one layer of simplices from *both* subdomains of a neighbors pair. All these convergence qualities are reflected by the spectrum of the preconditioned matrix.

For higher order discretizations the computational cost per iteration grows since matrices become very large, therefore a parallel implementation as the one of HPDDM [37] (a high-performance unified framework for domain decomposition methods which is interfaced with FreeFem++) should be considered for large scale problems, as in [25]. A two-level

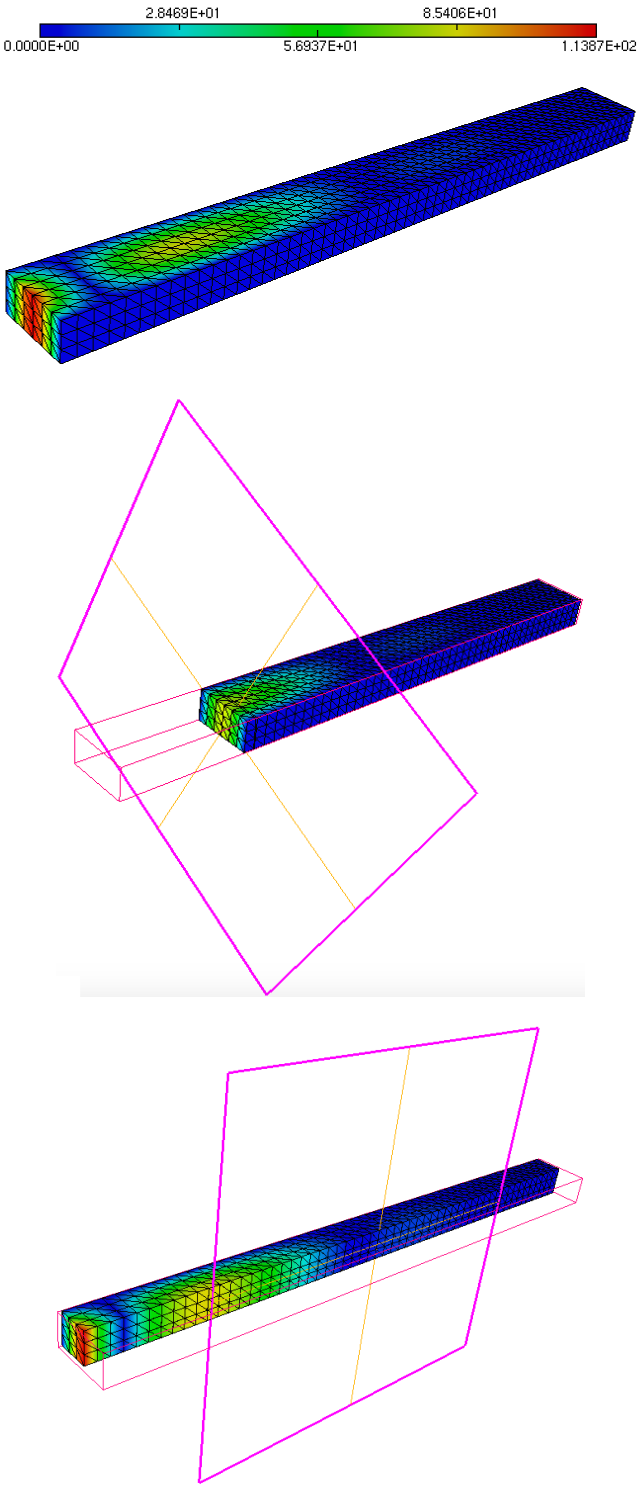


Figure 15: The norm of the real part of the solution for $\sigma = 0.15\,\mathrm{S\,m^{-1}}$, with two sections of the waveguide.

preconditioner via a coarse space correction should be designed for Maxwell's equations in order to fix the dependence on the number of subdomains or on the frequency of the iteration count. Note that in literature very few convergence theory results are available for domain decomposition methods applied to indefinite wave propagation problems. For the Helmholtz equation with absorption, rigorous convergence estimates have been recently presented in [38] for the two-level Additive Schwarz preconditioner, and their extension to the Maxwell case is work in progress.

Acknowledgement This work was financed by the French National Research Agency (ANR) in the framework of the project MEDIMAX, ANR-13-MONU-0012.

References

[1] A. Bossavit. *Computational electromagnetism.* Electromagnetism. Academic Press, Inc., San Diego, CA, 1998. Variational formulations, complementarity, edge elements.

[2] R. Hiptmair. Finite elements in computational electromagnetism. *Acta Numer.*, 11:237–339, 2002.

[3] F. Rapetti. High order edge elements on simplicial meshes. *M2AN Math. Model. Numer. Anal.*, 41(6):1001–1020, 2007.

[4] F. Rapetti and A. Bossavit. Whitney forms of higher degree. *SIAM J. Numer. Anal.*, 47(3):2369–2386, 2009.

[5] M. Ainsworth and J. Coyle. Hierarchic finite element bases on unstructured tetrahedral meshes. *Internat. J. Numer. Methods Engrg.*, 58(14):2103–2130, 2003.

[6] J. Schöberl and S. Zaglmayr. High order Nédélec elements with local complete sequence properties. *COMPEL*, 24(2):374–384, 2005.

[7] R. Hiptmair. Canonical construction of finite elements. *Math. Comp.*, 68(228):1325–1346, 1999.

[8] J. Gopalakrishnan, L. E. García-Castillo, and L. F. Demkowicz. Nédélec spaces in affine coordinates. *Comput. Math. Appl.*, 49(7-8):1285–1294, 2005.

[9] D. N. Arnold, R. S. Falk, and R. Winther. Finite element exterior calculus, homological techniques, and applications. *Acta Numer.*, 15:1–155, 2006.

[10] F. Hecht. New development in FreeFem++. *J. Numer. Math.*, 20(3-4):251–265, 2012.

[11] M. Bonazzoli and F. Rapetti. High-order finite elements in numerical electromagnetism: degrees of freedom and generators in duality. *Numerical Algorithms*, 74(1):111–136, 2017.

[12] S. Reitzinger and J. Schöberl. An algebraic multigrid method for finite element discretizations with edge elements. *Numerical Linear Algebra with Applications*, 9(3):223–238, 2002.

[13] P.B. Bochev, C. J. Garasi, J.J. Hu, A.C. Robinson, and R.S. Tuminaro. An improved algebraic multigrid method for solving Maxwell's equations. *SIAM J. Sci. Comput.*, 25(2):623–642, 2003.

[14] T.V. Kolev and P.S. Vassilevski. Auxiliary space AMG for $H(\mathrm{curl})$ problems. In *Domain decomposition methods in science and engineering XVII*, volume 60 of *Lect. Notes Comput. Sci. Eng.*, pages 147–154. Springer, Berlin, 2008.

[15] A. Bossavit and F. Rapetti. A prolongation/restriction operator for Whitney elements on simplicial meshes. *SIAM J. Numer. Anal.*, 43(5):2077–2097, 2005.

[16] J.H. Lai and L.N. Olson. Algebraic multigrid for high-order hierarchical $H(curl)$ finite elements. *SIAM J. Sci. Comput.*, 33(5):2888–2902, 2011.

[17] Andrea Toselli. Overlapping Schwarz methods for Maxwell's equations in three dimensions. *Numer. Math.*, 86(4):733–752, 2000.

[18] V. Dolean, M. J. Gander, and L. Gerardo-Giorda. Optimized Schwarz methods for Maxwell's equations. *SIAM J. Sci. Comput.*, 31(3):2193–2213, 2009.

[19] O. G. Ernst and M. J. Gander. Why it is difficult to solve Helmholtz problems with classical iterative methods. In *Numerical analysis of multiscale problems*, volume 83 of *Lect. Notes Comput. Sci. Eng.*, pages 325–363. Springer, Heidelberg, 2012.

[20] B. Després, P. Joly, and J. E. Roberts. A domain decomposition method for the harmonic Maxwell equations. In *Iterative methods in linear algebra (Brussels, 1991)*, pages 475–484, Amsterdam, 1992. North-Holland.

[21] P. Collino, G. Delbue, P. Joly, and A. Piacentini. A new interface condition in the non-overlapping domain decomposition method for the Maxwell equations. *Comput. Methods Appl. Mech. Engrg.*, 148(1-2):195–207, 1997.

[22] M. El Bouajaji, V. Dolean, M. J. Gander, and S. Lanteri. Optimized Schwarz methods for the time-harmonic Maxwell equations with dampimg. *SIAM J. Scient. Comp.*, 34(4):2048–2071, 2012.

[23] V. Dolean, M. J. Gander, S. Lanteri, J.-F. Lee, and Z. Peng. Effective transmission conditions for domain decomposition methods applied to the time-harmonic curl-curl Maxwell's equations. *J. Comput. Phys.*, 280:232–247, 2015.

[24] N. Marsic, C. Waltz, J. F. Lee, and C. Geuzaine. Domain decomposition methods for time-harmonic electromagnetic waves with high-order Whitney forms. *IEEE Transactions on Magnetics*, 52(3):1–4, March 2016.

[25] M. Bonazzoli, V. Dolean, F. Rapetti, and P.-H. Tournier. Parallel preconditioners for high-order discretizations arising from full system modeling for brain microwave imaging. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, pages e2229–n/a, 2017. e2229 jnm.2229.

[26] P. Monk. *Finite element methods for Maxwell's equations*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 2003.

[27] J.-C. Nédélec. Mixed finite elements in $\mathbf{R}^3$. *Numer. Math.*, 35(3):315–341, 1980.

[28] A. Bossavit. A rationale for 'edge-elements' in 3-d fields computations. *Magnetics, IEEE Transactions on*, 24(1):74–79, Jan 1988.

[29] A. Bossavit. Solving Maxwell equations in a closed cavity, and the question of 'spurious modes'. *Magnetics, IEEE Transactions on*, 26(2):702–705, Mar 1990.

[30] D. Boffi, P. Fernandes, L. Gastaldi, and I. Perugia. Computational models of electromagnetic resonators: analysis of edge element approximation. *SIAM J. Numer. Anal.*, 36(4):1264–1290, 1999.

[31] A. St-Cyr, M. J. Gander, and S. J. Thomas. Optimized multiplicative, additive, and restricted additive Schwarz preconditioning. *SIAM Journal on Scientific Computing*, 29(6):2402–2425, 2007.

[32] P.-H. Tournier, I. Aliferis, M. Bonazzoli, M. de Buhan, M. Darbas, V. Dolean, F. Hecht, P. Jolivet, I. El Kanfoud, C. Migliaccio, F. Nataf, C. Pichot, and S. Semenov. Microwave Tomographic Imaging of Cerebrovascular Accidents by Using High-Performance Computing. preprint, submitted, 2016.

[33] V. Dolean, P. Jolivet, and F. Nataf. *An Introduction to Domain Decomposition Methods: algorithms, theory and parallel implementation.* SIAM, 2015.

[34] F. Ihlenburg and I. Babuška. Finite element solution of the Helmholtz equation with high wave number. I. The h-version of the FEM. *Comput. Math. Appl.*, 30(9):9–37, 1995.

[35] M. Bonazzoli, E. Gaburro, V. Dolean, and F. Rapetti. High order edge finite element approximations for the time-harmonic Maxwell's equations. In *2014 IEEE Conference on Antenna Measurements Applications (CAMA) Proceedings*, November 2014.

[36] P. Amestoy, I. Duff, J. L'Excellent, and J. Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[37] P. Jolivet, F. Hecht, F. Nataf, and C. Prud'Homme. Scalable domain decomposition preconditioners for heterogeneous elliptic problems. In *Proc. of the Int. Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–11. IEEE, 2013.

[38] I. G. Graham, E. A. Spence, and E. Vainikko. Domain decomposition preconditioning for high-frequency Helmholtz problems with absorption. *Math. Comp.*, Feb. 8 2017.