An adaptive finite element method for high-frequency scattering problems with smoothly varying coefficients

Anton Arnold, Sjoerd Geevers, Ilaria Perugia, Dmitry Ponomarev

AN ADAPTIVE FINITE ELEMENT METHOD FOR HIGH-FREQUENCY SCATTERING PROBLEMS WITH SMOOTHLY VARYING COEFFICIENTS

ANTON ARNOLD[1*], SJOERD GEEVERS[2*], ILARIA PERUGIA[2*], DMITRY PONOMAREV [1,3*]

[1] *Institute for Analysis and Scientific Computing, Vienna University of Technology*
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
[2] *Faculty of Mathematics, University of Vienna*
Oskar-Morgenstern-Platz 1, 1090 Vienna, Austria
[3] *St. Petersburg Department of V. A. Steklov Mathematical Institute, RAS,*
Fontanka 27, 191023 St. Petersburg, Russia

January 14, 2022

Abstract. We introduce a new numerical method for solving time-harmonic acoustic scattering problems. The main focus is on plane waves scattered by smoothly varying material inhomogeneities. The proposed method works for any frequency ω, but is especially efficient for high-frequency problems. It is based on a time-domain approach and consists of three steps: *i)* computation of a suitable incoming plane wavelet with compact support in the propagation direction; *ii)* solving a scattering problem in the time domain for the incoming plane wavelet; *iii)* reconstruction of the time-harmonic solution from the time-domain solution via a Fourier transform in time. An essential ingredient of the new method is a front-tracking mesh adaptation algorithm for solving the problem in *ii)*. By exploiting the limited support of the wave front, this allows us to make the number of the required degrees of freedom to reach a given accuracy significantly less dependent on the frequency ω. We also present a new algorithm for computing the Fourier transform in *iii)* that exploits the reduced number of degrees of freedom corresponding to the adapted meshes. Numerical examples demonstrate the advantages of the proposed method and the fact that the method can also be applied with external source terms such as point sources and sound-soft scatterers. The gained efficiency, however, is limited in the presence of trapping modes.

Keywords Helmholtz equation, scattering problem, time-domain wave problem, limiting amplitude principle, adaptive FEM, front-tracking mesh

Mathematics Subject Classification 35J05, 35L05, 65M60, 65M50

1. Introduction

We consider time-harmonic wave scattering problems in inhomogeneous media with smoothly varying material properties. Such problems become notoriously hard to solve when the angular frequency ω is large. To obtain a given accuracy with a standard finite difference or finite element method requires at least $\mathcal{O}(\omega^d)$ degrees of freedom [6], where d denotes the number of space dimensions. On top of that, standard iterative solvers and multigrid methods break down or converge slowly for high frequencies [18].

*A. Arnold, S. Geevers, and I. Perugia have been funded by the Austrian Science Fund (FWF) through the project F 65 "Taming Complexity in Partial Differential Systems". I. Perugia has also been funded by the FWF through the project P 29197-N32. A. Arnold and D. Ponomarev were supported by the bi-national FWF-project I3538-N32.

One way to reduce the computational complexity for large frequencies is by combining finite element methods with asymptotic methods [23, 36]. An asymptotic method, such as the geometrical optics method, is used to determine the wave propagation directions and a plane-wave finite element method is then used to solve the scattering problem. A drawback of this approach is that standard geometrical optics does not account for diffracted fields and incorporating diffraction phenomena typically requires an *ad hoc* approach. Instead of using an asymptotic method, one can also extract the dominant wave propagation directions from the solution of a lower-frequency scattering problem [19].

Another way to reduce the computational complexity is by using a classical finite difference or finite element method with a standard iterative solver, but in combination with a sweeping preconditioner [16, 17, 40]; for a more recent overview of several sweeping preconditioning methods, see [20]. The number of iterations then becomes nearly independent of the frequency, which means that the computational complexity scales almost as $\mathcal{O}(\omega^d)$.

Instead of solving the scattering problem directly in the frequency domain, one can also solve the scattering problem in the time domain. The time-harmonic solution can be obtained from a solution to a time-dependent wave equation by exploiting the limiting amplitude principle [33, 45, 35] or by applying a Fourier transformation in time. Classical time domain methods are the finite difference and finite element time domain methods [46, 42]. Time-domain methods that are specifically devised for solving frequency-domain problems include the controllability method [12, 24], with its spectral version [30] and its extensions [29, 27], the WaveHoltz method [5], and the time-domain preconditioner of [41].

While both frequency-domain and time-domain methods are commonly used in practice and are expected to remain relevant in the future, this paper will focus on the time-domain approach. Some of the advantages of time-domain methods are that they are inherently parallel and straightforward to implement, without the need of storing Krylov subspaces and matrix factorisations and without the need of implementing linear solvers and moving absorbing boundary layers. However, for classical finite difference and finite element time domain methods, the number of degrees of freedom is at least $\mathcal{O}(\omega^d)$ and the number of time steps is at least $\mathcal{O}(\omega)$ due to the CFL condition, resulting in a computational complexity of at least $\mathcal{O}(\omega^{d+1})$. In this paper, we present an adaptive finite element time-domain method that reduces the average number of degrees of freedom per time step to almost $\mathcal{O}(\omega^{d-1})$, resulting in a computational cost that scales almost as $\mathcal{O}(\omega^d)$.

The main idea of adaptive finite element time-domain methods is to automatically adapt the mesh over time in such a way that fine elements are used near the wave front and coarser elements are used away from the wave front. The mesh adaptation algorithms are typically driven by *a posteriori* error estimators/indicators. Adaptive finite element methods for the wave equation were studied for a conforming finite element discretisation in space combined with a discontinuous Galerkin discretisation in time [31, 34, 44], the Crank–Nicholson scheme in time [9, 26, 25], the implicit Euler scheme in time [11, 21], and the leap-frog scheme in time [22]. Adaptive finite element methods based on a discontinuous Galerkin discretisation in space were presented and analysed in [1, 3, 2]. An anisotropic adaptive mesh refinement algorithm was studied in [39]. Adaptive finite element schemes tailored for a given goal functional were studied in [10, 8]. Adaptive finite element schemes have also been studied for other time-dependent problems such as the Stefan problem [37] and nonlinear wave equations [4]. Here, we present a new adaptive finite element time-domain method that is tailored for efficiently solving time-harmonic scattering problems and has the following distinguishing features:

- The source term is obtained from an incoming plane wavelet with compact support in the direction of propagation of width $\mathcal{O}(\omega^{-1})$.
- The adapted meshes are obtained from a set of nested meshes that are defined *a priori*.

- The mesh is not updated at each time step, but only after every m time steps. In the numerical examples, we have $m \sim 10 - 100$.
- The time-harmonic field is obtained using an adapted algorithm for computing the Fourier transform in time that exploits the reduced number of degrees of freedom of the adapted meshes.

In the numerical section, we present an implementation of the method using a fully explicit conforming finite element time-stepping scheme combined with the perfectly matched layer of [28].

The paper is organised as follows: In Section 2, we explain how we solve the time-harmonic scattering problem using a time-domain approach. The adaptive finite element method for solving the time-dependent problem and the adapted method for computing the Fourier transform in time are then given in Section 3. Details of the numerical implementation and several numerical examples are given in Section 4. Finally, our findings are summarised in Section 5.

2. Solving the time-harmonic scattering problem in the time domain

We are interested in solving the time-harmonic wave scattering problem in d dimensions governed by the Helmholtz equation

$$-\omega^2(U_S + U_I) - \beta^{-1}\nabla \cdot (\alpha \nabla (U_S + U_I)) = 0 \qquad \text{in } \mathbb{R}^d, \tag{1a}$$

$$\text{[far field radiation condition on } U_S\text{]}, \tag{1b}$$

where $U_S = U_S(\mathbf{x})$ is the scattered wave field that needs to be resolved, $U_I = U_I(\mathbf{x})$ is a given incoming plane wave, $\omega > 0$ is the angular frequency, ∇ and $\nabla\cdot$ denote the gradient and divergence operator, respectively, and $\alpha = \alpha(\mathbf{x}) \geq \alpha_{\min} > 0$ and $\beta = \beta(\mathbf{x}) \geq \beta_{\min} > 0$ are two material parameters that are assumed to vary smoothly in space. Our main interest is the case where ω is large. We assume that there exists a bounded domain $\Omega_{in} \subset \mathbb{R}^d$ such that α and β are constant, say $\alpha = \alpha_0$ and $\beta = \beta_0$, in the exterior domain $\Omega_{ex} := \mathbb{R}^d \setminus \overline{\Omega}_{in}$. We also assume that the incoming plane wave is of the form $U_I(\mathbf{x}) = e^{i\omega(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0}$, with i the imaginary unit ($i^2 = -1$), $\hat{\mathbf{r}}$ a unit direction vector, and $c_0 := \sqrt{\alpha_0/\beta_0}$ the wave propagation speed in the exterior domain.

We can rewrite (1) as

$$-\omega^2 U_S - \beta^{-1}\nabla \cdot (\alpha \nabla U_S) = F \qquad \text{in } \mathbb{R}^d, \tag{2a}$$

$$\text{[far field radiation condition on } U_S\text{]}, \tag{2b}$$

with $F := \omega^2 U_I + \beta^{-1}\nabla \cdot (\alpha \nabla U_I)$. Note that $F = 0$ in Ω_{ex}.

Remark 2.1. *In acoustic scattering, U_S is the scattered pressure field, α is the reciprocal of the mass density $\rho = \rho(\mathbf{x})$ of the medium, and β is the reciprocal of ρc^2, with $c = c(\mathbf{x})$ being the wave propagation speed of the medium.*

A common way to obtain the scattered wave field U_S is by solving a wave scattering problem in the time domain for a time-harmonic source term of the form $F(\mathbf{x})e^{-i\omega t}$. If the *limiting amplitude principle* is valid [38, 15], the time-dependent scattered wave field converges to $U_S(\mathbf{x})e^{-i\omega t}$ as t tends to infinity; see Appendix A.

Alternatively, we can compute the scattered wave field $u_S(\mathbf{x}, t)$ for a suitable source term $f(\mathbf{x}, t)$ with compact support in space and time. Let $\mathfrak{F}_t$ denote the Fourier transform with respect to time, namely $\mathfrak{F}_t[\varphi](\omega') = \int_{\mathbb{R}} e^{-i\omega' t}\varphi(t)\,\mathrm{d}t$. If $\mathfrak{F}_t[f](\cdot, -\omega) = F$, then it follows from the limiting amplitude principle that $U_S = \mathfrak{F}_t[u_S](\cdot, -\omega)$; see Lemma A.2 in Appendix A. Our proposed numerical method is based on this latter approach. In particular, we solve the scattered wave field $u_S(\mathbf{x}, t)$ corresponding to a single incoming plane wavelet $u_I(\mathbf{x}, t)$, defined such that $U_I = \mathfrak{F}_t[u_I](\cdot, -\omega)$, and then compute $U_S = \mathfrak{F}_t[u_S](\cdot, -\omega)$. By *plane wavelet*

we mean a plane wave with compact support in the propagation direction. We describe this approach in detail in the three steps below.

Step 1. Defining the incoming plane wavelet. We consider an incoming plane wavelet of the form $u_I(\mathbf{x},t) := \omega\psi(\omega(t-(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0))$, where $\psi = \psi(\xi)$ is some real-valued, smooth function with $\mathrm{supp}(\psi) = [-\xi_0,\xi_0]$, where $\xi_0 > 0$ is some constant independent of ω, and such that its Fourier transform $\mathfrak{F}[\psi]$ satisfies $\mathfrak{F}[\psi](-1) = \int_{-\xi_0}^{\xi_0} e^{i\xi}\psi(\xi)\,\mathrm{d}\xi = 1$. The incoming wave field is thus a traveling plane wavelet of amplitude $\mathcal{O}(\omega)$ and with a support of width $2c_0\xi_0\omega^{-1} = \mathcal{O}(\omega^{-1})$. The Fourier transform of u_I is given by

$$\begin{aligned}
\mathfrak{F}_t[u_I](\mathbf{x},\tilde{\omega}) &= \mathfrak{F}_t[\omega\psi(\omega(t-(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0))](\mathbf{x},\tilde{\omega}) \\
&= \omega\mathfrak{F}_t[\psi(\omega(t-(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0))](\mathbf{x},\tilde{\omega}) \\
&= \omega e^{-i\tilde{\omega}(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0}\mathfrak{F}_t[\psi(\omega t)](\mathbf{x},\tilde{\omega}) \\
&= e^{-i\tilde{\omega}(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0}\mathfrak{F}_t[\psi(t)](\mathbf{x},\tilde{\omega}/\omega) \\
&= e^{-i\tilde{\omega}(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0}\mathfrak{F}[\psi](\tilde{\omega}/\omega)
\end{aligned}$$

and since $\mathfrak{F}[\psi](-1) = 1$, we therefore have $\mathfrak{F}_t[u_I](\cdot,-\omega) = U_I$. An illustration of u_I is given in Figure 1.

Step 2. Solving a wave scattering problem in the time-domain. Having defined the incoming plane wavelet u_I, we next solve the scattered wave field $u_S(\mathbf{x},t)$ given by the wave equation

$$\partial_t^2(u_S+u_I) - \beta^{-1}\nabla\cdot(\alpha\nabla(u_S+u_I)) = 0 \qquad \text{in } \mathbb{R}^d\times(t_0,\infty), \tag{3a}$$

$$[\text{zero initial conditions on } u_S \text{ at } t=t_0], \tag{3b}$$

where ∂_t^2 denotes the second-order time derivative, and $t_0 := \inf_{\mathbf{x}\in\Omega_{in}}(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0 - \xi_0\omega^{-1}$ is the time when the incoming plane wavelet first enters Ω_{in}. We can rewrite this equation as

$$\partial_t^2 u_S - \beta^{-1}\nabla\cdot(\alpha\nabla u_S) = f \qquad \text{in } \mathbb{R}^d\times(t_0,\infty), \tag{4a}$$

$$[\text{zero initial conditions on } u_S \text{ at } t=t_0], \tag{4b}$$

where $f := -\partial_t^2 u_I + \beta^{-1}\nabla\cdot(\alpha\nabla u_I)$. Note that f has only support in $\Omega_{in}\times(t_0,t_f)$, where $t_f := \sup_{\mathbf{x}\in\Omega_{in}}(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0 + \xi_0\omega^{-1}$ is the time when the incoming plane wavelet leaves Ω_{in}. Furthermore, since $\mathfrak{F}_t[u_I](\cdot,-\omega) = U_I$, it follows that $\mathfrak{F}_t[f](\cdot,-\omega) = F$.

Step 3. Reconstructing the Helmholtz solution using a Fourier transform. Since $\mathfrak{F}_t[f](\cdot,-\omega) = F$, it follows that, after extending u_S by zero in $\mathbb{R}^d\times(-\infty,t_0)$, we have $\mathfrak{F}_t[u_S](\cdot,-\omega) = U_S$. Having computed the scattered wave field u_S given by (4), we can thus reconstruct the solution to the Helmholtz equation given in (2) by computing the limit

$$U_S = \mathfrak{F}_t[u_S](\cdot,-\omega) = \lim_{t\to\infty}\int_{t_0}^{t} e^{i\omega\tau}u_S(\cdot,\tau)\,\mathrm{d}\tau. \tag{5}$$

The advantage of this approach is that the scattered wave field u_S corresponds to an incoming plane wavelet of amplitude $\mathcal{O}(\omega)$ and with a support of width $\mathcal{O}(\omega^{-1})$. Motivated by geometric optics, we expect that, when ω is large and when there are no trapping modes, the solution u_S is a travelling wave that has a steep gradient near the wave front, and a small gradient that is more or less independent of ω everywhere else; see also Figure 6 in Section 4. In the finite element approximation of u_S, we can exploit this property and significantly reduce the computational cost by using an adaptive, time-dependent spatial mesh, where a fine mesh is used near the wave front and a coarser mesh, with mesh width independent of ω, is used elsewhere.

Remark 2.2. *We can readily extend the approach for a wave scattering problem that includes a sound-soft scatterer Ω_{sc}. The wave scattering problem is then given by equation* (1)*, but*

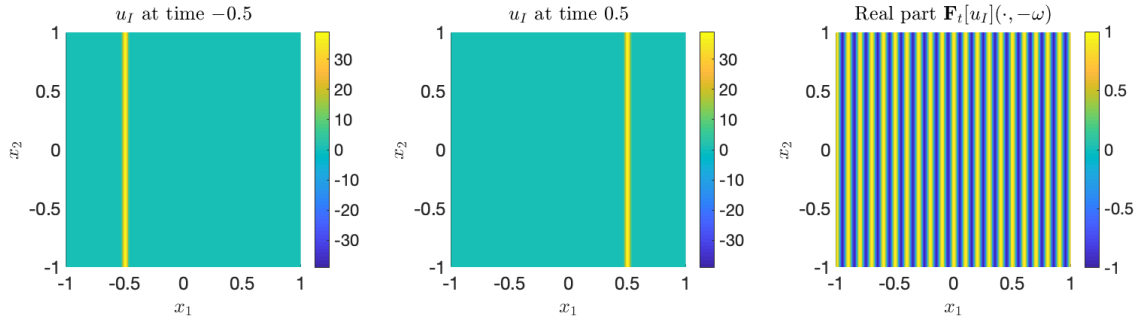

FIGURE 1. Illustration of the incoming plane wavelet $u_I(\mathbf{x},t) = \omega\psi(\omega(t-(\hat{\mathbf{r}}\cdot\mathbf{x})/c_0))$ at time $t=-0.5$ (left) and $t=0.5$ (middle), and its Fourier transform $\mathfrak{F}_t[u_I](\cdot,-\omega)$ (right). We chose $\omega = 20\pi$, $\hat{\mathbf{r}} = (1,0)$, $c_0 = 1$, and ψ defined as in Section 4.2.

with a spatial domain $\mathbb{R}^d \setminus \Omega_{sc}$ instead of $\mathbb{R}^d$, and with an additional boundary condition of the form

$$U_S + U_I = 0 \qquad \text{on } \partial\Omega_{sc}.$$

The approach remains identical, except that in equations (3) *and* (4) *in Step 2, the spatial domain is $\mathbb{R}^d \setminus \Omega_{sc}$ instead of $\mathbb{R}^d$ and the additional boundary condition is of the form*

$$u_S = -u_I \qquad \text{on } (t_0,\infty)\times\partial\Omega_{sc}.$$

Remark 2.3. *With a slight modification, the approach can also be applied to a wave scattering problem of the form in* (2) *with $F = F(\mathbf{x})$ an arbitrary external source term with bounded support. In Step 1, we then only need to define ψ, but not u_I. In Step 2, we then solve equation* (4) *for $f(\mathbf{x},t) = \omega\psi(\omega t)F(\mathbf{x})$, with $t_0 := -\xi_0\omega^{-1}$ the earliest time when $f(\cdot,t)$ is non-zero. Note that f only has support in $supp(F)\times(-\xi_0\omega^{-1},\xi_0\omega^{-1})$. Step 3 remains unaltered. If F is a source term with very local support in space, such as a point source, we again expect that we can solve the time-domain problem efficiently using an adaptive mesh.*

A description of the adaptive finite element method is given in the following section.

3. An adaptive finite element method

We aim to solve the wave equation given in (4) by using a finite element method with an adapted spatial mesh that is constantly updated over time. We simultaneously update the right-hand side of (5) by applying a discretised Fourier transform in time that exploits the reduced number of degrees of freedom of the adapted spatial meshes.

The main idea of the adaptive finite element method is as follows: we consider a finite computational domain $\Omega \supset \Omega_{in}$ with an absorbing boundary and split the time domain into small intervals (T_{j-1},T_j) of length $\mathcal{O}(\omega^{-1})$. Let u_h denote the finite element approximation to u_S. At the beginning of each time interval, we construct an adapted mesh $\mathcal{T}_j$ of the domain Ω based on the current discrete approximation $u_h(\cdot,T_{j-1})$. We then project $u_h(\cdot,T_{j-1})$ into the finite element space of $\mathcal{T}_j$ and solve the discretised wave equation on $\mathcal{T}_j$ for the time interval (T_{j-1},T_j).

For constructing $\mathcal{T}_j$, we aim for the coarsest possible mesh on which u_S can still be approximated accurately during the time interval (T_{j-1},T_j). To construct such a mesh, we assume that the current discrete approximation $u_h(\cdot,T_{j-1})$ is an accurate approximation of $u_S(\cdot,T_{j-1})$ and take into account that u_S travels during the time interval (T_{j-1},T_j) and is generated by the incoming wave u_I. Let $c_{\max} := \sup_{\mathbf{x}\in\Omega}\sqrt{\alpha(\mathbf{x})/\beta(\mathbf{x})}$ denote the maximum wave propagation speed. The construction of $\mathcal{T}_j$ consists of the following steps:

- *Coarsen.* Coarsen the mesh $\mathcal{T}_{j-1}$ and project $u_h(\cdot, T_{j-1})$ onto the discrete space associated with the coarser mesh.
- *Estimate.* Compute the projection error, i.e. compute the difference between $u_h(\cdot, T_{j-1})$ and its projection onto the coarser mesh.
- *Mark 1.* Mark all elements of the coarser mesh where the projection error is above a certain threshold. Also mark all elements of the coarser mesh that overlap with the support of the incoming wave $u_I(\cdot, T_{j-1})$.
- *Mark 2.* Mark all elements of the coarser mesh that are within a distance $c_{\max}(T_j - T_{j-1})$ of elements that were marked in the first round.
- *Refine.* Refine the coarser mesh at all marked elements to obtain $\mathcal{T}_j$.

To accurately approximate the Fourier transform of u_h in time, we normally need to sample u_h at all the degrees of freedom of a uniformly fine mesh at each time step. By computing the Fourier transform using an adapted algorithm, we can significantly reduce the average number of sampling points per time step.

A detailed description of the complete method is given in the following subsections.

3.1. **A finite element method with a time-dependent mesh.** Consider the wave scattering problem given in (4). To approximate the scattered wave field u_S using a finite element method, we consider a bounded polygonal domain $\Omega \supset \Omega_{in}$, and impose an absorbing boundary condition or add an absorbing boundary layer at $\partial\Omega$. Since, however, the main steps of our adaptive finite element method do not depend on the type of boundary condition, we consider in this section a zero Dirichlet boundary condition $u_S|_{\partial\Omega} = 0$ in order to simplify the presentation. We thus consider a wave equation of the form

$$\partial_t^2 u - \beta^{-1}\nabla\cdot(\alpha\nabla u) = f \qquad \text{in } \Omega\times(t_0,\infty), \tag{6a}$$

$$u(\cdot,t_0) = \partial_t u(\cdot,t_0) = 0 \qquad \text{in } \Omega, \tag{6b}$$

$$u = 0 \qquad \text{on } \partial\Omega\times(t_0,\infty). \tag{6c}$$

Let $T_{up} = \mathcal{O}(\omega^{-1})$ be the time after which the mesh is updated and define $T_j := t_0 + jT_{up}$. Also, let $\mathcal{T}_j$ denote the simplicial/square/cubic mesh of Ω used during the time interval $(T_{j-1}, T_j]$. For any mesh $\mathcal{T}$, let $\mathcal{U}_\mathcal{T}$ denote the corresponding finite element space, given by

$$\mathcal{U}_\mathcal{T} := \{u \in H_0^1(\Omega) \mid u\circ\phi_E \in \hat{\mathcal{U}} \text{ for all } E\in\mathcal{T}\},$$

with $\phi_E : \hat{E} \to E$ the affine element mapping, $\hat{E}$ the reference element, and $\hat{\mathcal{U}}$ the polynomial reference space. Also, let $\mathcal{L}_\mathcal{T} : \mathcal{U}_\mathcal{T} \to \mathcal{U}_\mathcal{T}$ denote the discretisation of the spatial operator $u \mapsto -\beta^{-1}\nabla\cdot(\alpha\nabla u)$ for a given mesh $\mathcal{T}$. For simplicity, we consider here the operator $\mathcal{L}_\mathcal{T}$ defined such that

$$(\beta\mathcal{L}_\mathcal{T} u, w) = (\alpha\nabla u, \nabla w) \qquad \forall w\in\mathcal{U}_\mathcal{T},$$

where $(\cdot,\cdot)$ denotes the $L^2(\Omega)$ or $L^2(\Omega)^d$ inner product. A slightly different discretisation that allows for an explicit expression of $\mathcal{L}_\mathcal{T}$ is given in Section 4.1.2. Finally, let Π_j denote a projection operator that projects into the space $\mathcal{U}_{\mathcal{T}_j}$. The semi-discrete finite element formulation can be stated as follows: for $j = 1, 2, \ldots$, find $u_j : [T_{j-1}, T_j] \to \mathcal{U}_{\mathcal{T}_j}$ such that

$$\partial_t^2 u_j + \mathcal{L}_{\mathcal{T}_j} u_j = f_{\mathcal{T}_j} \qquad \text{in } \Omega\times(T_{j-1},T_j), \tag{7a}$$

$$u_j(\cdot,T_{j-1}) = \Pi_j u_{j-1}(\cdot,T_{j-1}), \tag{7b}$$

$$\partial_t u_j(\cdot,T_{j-1}) = \Pi_j \partial_t u_{j-1}(\cdot,T_{j-1}), \tag{7c}$$

with $u_0(\cdot,T_0)\equiv 0$, $\partial_t u_0(\cdot,T_0)\equiv 0$, and $f_{\mathcal{T}_j}(\cdot,t)\in\mathcal{U}_{\mathcal{T}_j}$ a discretisation of $f(\cdot,t)$.

For the time discretisation, we consider here the central difference scheme, although the adaptive method can also be applied to other time integration schemes. Let $\Delta t = T_{up}/m$

denote the time step size, with $m > 0$ some positive integer. Also, let $t^n := t_0 + n\Delta t$ and $u_j^n := u_j(\cdot, t^n)$. We approximate $\partial_t^2 u_j(\cdot, t^n)$ by the central difference scheme

$$D_t^2 u_j^n := \frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2}. \tag{8}$$

Now, let $n_j := mj$. The fully discrete finite element formulation can be stated as follows: for $j = 1, 2, \ldots$ and for $n : n_{j-1} - 1 \le n \le n_j$, find $u_j^n \in \mathcal{U}_{\mathcal{T}_j}$ such that

$$D_t^2 u_j^n + \mathcal{L}_{\mathcal{T}_j} u_j^n = f_{\mathcal{T}_j}(\cdot, t^n) \qquad \text{for } n : n_{j-1} \le n \le n_j - 1, \tag{9a}$$

$$u_j^n = \Pi_j u_{j-1}^n \qquad \text{for } n = n_{j-1} - 1 \text{ and } n = n_{j-1}, \tag{9b}$$

with $u_0^0 \equiv 0$ and $u_0^{-1} \equiv 0$. We can rewrite equation (9a) as

$$u_j^{n+1} = -u_j^{n-1} + 2u_j^n + \Delta t^2(-\mathcal{L}_{\mathcal{T}_j} u_j^n + f_{\mathcal{T}_j}(\cdot, t^n)). \tag{10}$$

An extended time stepping scheme, that takes into account an absorbing boundary layer and the discretisation of f, is given in Section 4.1.2.

In practice, we cannot solve the wave equation for $t \to \infty$, but have to stop at some finite time $t_{stop} = T_{j_{stop}}$. To determine j_{stop}, we use a stopping criterion of the form

$$\sup_{\mathbf{x} \in \Omega} |u_j^{n_j}(\mathbf{x})| \le \epsilon_0, \tag{11}$$

where $\epsilon_0 > 0$ is an *a priori* chosen threshold value. In other words, we stop the computations when the scattered wave field is close to zero, which means it has almost completely left the computational domain.

An overview of how to implement the adaptive finite element method is given in Algorithm 1. Here, we use the following functions:

- $\mathcal{T}_j$ = updateMesh$(\mathcal{T}_{j-1}, u_{j-1}^n, t^n)$: computes the new mesh $\mathcal{T}_j$ given the current mesh $\mathcal{T}_{j-1}$, the current discrete wave field u_{j-1}^n, and the current time t^n. A detailed description of updateMesh and how to choose the initial mesh $\mathcal{T}_0$ is given in Section 3.2 below.
- $u_j^n :=$ project$(u_{j-1}^n, \mathcal{T}_{j-1}, \mathcal{T}_j)$: computes the projection $u_j^n = \Pi_j u_{j-1}^n$.
- $u_j^{n+1} =$ doTimeStep$(u_j^n, u_j^{n-1}, \mathcal{T}_j, t^n)$: computes the wave field at the next time step u_j^{n+1} using the formula in (10).
- Stop$(u_j^n, \mathcal{T}_j, t^n)$: returns **true** if $t^n > t_f$ and if the stopping criterion given in (11) is satisfied. Returns **false** otherwise.

3.2. **Adapting the mesh.** To construct adapted meshes $\mathcal{T}_j$, we define *a priori* a set of nested meshes $\{\mathcal{T}^1, \mathcal{T}^2, \ldots, \mathcal{T}^K\}$, $K \ge 2$, where $\mathcal{T}^1$ is the coarsest mesh with a mesh width h_1 independent of ω, and $\mathcal{T}^K$ is the finest mesh with a mesh width h_K of order ω^{-1} or less. We assume that, in case of no mesh adaptation, the mesh $\mathcal{T}^K$ is sufficiently fine for solving the wave equation with the desired accuracy.

We construct adapted meshes $\mathcal{T}_j$ from elements in $\bigcup_{k=1}^K \mathcal{T}^k$. The initial mesh $\mathcal{T}_0$ is chosen as the finest mesh $\mathcal{T}^K$. The algorithm $\mathcal{T}_j =$ updateMesh$(\mathcal{T}_{j-1}, u_{j-1}^n, t^n)$ for updating the mesh is given in Algorithm 2, which consists of the following functions:

- $\mathcal{P} =$ getParentElements$(\mathcal{T})$: returns the set of all elements in $\bigcup_{k=1}^{K-1} \mathcal{T}^k$ that are coarser than those of the given mesh $\mathcal{T}$, i.e. it returns
$$\mathcal{P} = \{E \in \bigcup_{k=1}^{K-1} \mathcal{T}^k \setminus \mathcal{T} \mid E \supset E' \text{ for some } E' \in \mathcal{T}\}.$$
- $\mathcal{T} =$ getChildElements$(\mathcal{P})$: returns the mesh $\mathcal{T} \subset \bigcup_{k=1}^K \mathcal{T}^k$, given its parent elements $\mathcal{P} =$ getParentElements$(\mathcal{T})$. In particular, getChildElements returns $\mathcal{T} = \bigcup_{E \in \mathcal{P}}$ getSubElements$(E) \setminus \mathcal{P}$.

Algorithm 1 solving the wave equation using a time-dependent mesh

procedure SOLVEWAVEEQUATION
  $\mathcal{T}_h \leftarrow \mathcal{T}_0$ ▷ set initial mesh
  $u_h \leftarrow 0$, $u_h^{new} \leftarrow 0$, and $u_h^{old} \leftarrow 0$ ▷ initialise wave field
  for $j = 1, 2, \ldots$ **do**
    $n = n_{j-1}$ ▷ at this point, $u_h = u_{j-1}^n$, $u_h^{old} = u_{j-1}^{n-1}$, $\mathcal{T}_h = \mathcal{T}_{j-1}$
    if STOP$(u_h, \mathcal{T}_h, t^n)$ **then**
      return u_h
    end if
    $\mathcal{T}_h^{new} \leftarrow$ UPDATEMESH$(\mathcal{T}_h, u_h, t^n)$ ▷ $\mathcal{T}_h^{new} \leftarrow \mathcal{T}_j$
    $u_h \leftarrow$ PROJECT$(u_h, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $u_h \leftarrow u_j^n$
    $u_h^{old} \leftarrow$ PROJECT$(u_h^{old}, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $u_h^{old} \leftarrow u_j^{n-1}$
    $\mathcal{T}_h \leftarrow \mathcal{T}_h^{new}$ ▷ $\mathcal{T}_h \leftarrow \mathcal{T}_j$
    for $\ell = 0, 1, 2, \ldots, m-1$ **do**
      $n \leftarrow n_{j-1} + \ell$ ▷ at this point, $u_h = u_j^n$, $u_h^{old} = u_j^{n-1}$
      $u_h^{new} \leftarrow$ DOTIMESTEP$(u_h, u_h^{old}, \mathcal{T}_h, t^n)$ ▷ $u_h^{new} \leftarrow u_j^{n+1}$
      $u_h^{old} \leftarrow u_h$
      $u_h \leftarrow u_h^{new}$
    end for
  end for
end procedure

- $\mathcal{T}_E =$ GETSUBELEMENTS(E): returns, for a given element $E \in \mathcal{T}^k$ with $k \leq K-1$, the set of the elements in $\mathcal{T}^{k+1}$ that are a subset of E.
- $\mathcal{P}_{j-1}^* =$ MARKELEMENTS$(\mathcal{P}_{j-1}, u_{j-1}^n, t^n)$: returns the set of all elements in $\mathcal{P}_{j-1}$ that need to be refined. We mark all elements $E \in \mathcal{P}_{j-1}$ that also have subelements in $\mathcal{P}_{j-1}$. For the elements $E \in \mathcal{P}_{j-1}$ that have no further subelements in $\mathcal{P}_{j-1}$, we only mark those for which the function NEEDSREFINEMENT(E, u_{j-1}^n, t^n) returns **true**. Pseudocode of the function MARKELEMENTS is given in Algorithm 3.
- NEEDSREFINEMENT(E, u_{j-1}^n, t^n): returns **true** if and only if

$$E \cap \operatorname{support}(u_I(\cdot, t^n)) \neq \emptyset, \tag{12}$$

or

$$\eta_E := \sup_{\mathbf{x} \in E} |u_{j-1}^n(\mathbf{x}) - \Pi_E u_{j-1}^n(\mathbf{x})| > \eta_0, \tag{13}$$

where Π_E denotes a projection operator that projects into the discrete space of E and $\eta_0 > 0$ is some threshold value defined *a priori*.
- $\mathcal{P}_j =$ MARKNEARBYELEMENTS$(\mathcal{P}_{j-1}^*)$: for $k = 1, 2, \ldots, K-1$, returns all elements in $\mathcal{T}^k$ that are within a distance $c_{\max} T_{up}$ of an element in $\mathcal{P}_{j-1}^* \cap \mathcal{T}^k$. The distance between two elements E_1 and E_2 is defined as $\operatorname{dist}(E_1, E_2) := \inf_{\mathbf{x} \in E_1, \mathbf{y} \in E_2} |\mathbf{x} - \mathbf{y}|$.

An illustration of the mesh adaptation algorithm is given in Figure 8 below.

3.3. **Computing the Fourier Transformation.** We can approximate the Fourier transform in (5) by a discrete Fourier transformation:

$$U_S = \mathfrak{F}_t[u_S](\cdot, -\omega) = \int_{t_0}^{\infty} e^{i\omega t} u_S(\cdot, t)\, \mathrm{d}t \approx \sum_{n=1}^{\infty} \Delta t e^{i\omega t^n} u_S(\cdot, t^n).$$

Furthermore, we can approximate $u_S(\cdot, t^n)$ by the finite element approximation u_h^n, where $u_h^n := u_j^n$ for $n : n_{j-1} + 1 \leq n \leq n_j$ and where u_j^n is the solution to the fully discrete problem

Algorithm 2 update the mesh

function UPDATEMESH($\mathcal{T}_{j-1}, u^n_{j-1}, t^n$)
  $\mathcal{P}_{j-1} \leftarrow$ GETPARENTELEMENTS($\mathcal{T}_{j-1}$)
  $\mathcal{P}^*_{j-1} \leftarrow$ MARKELEMENTS($\mathcal{P}_{j-1}, u^n_{j-1}, t^n$)
  $\mathcal{P}_j \leftarrow$ MARKNEARBYELEMENTS($\mathcal{P}^*_{j-1}$)
  $\mathcal{T}_j \leftarrow$ GETCHILDELEMENTS($\mathcal{P}_j$)
  return $\mathcal{T}_j$
end function

Algorithm 3 mark elements for refinement

function MARKELEMENTS($\mathcal{P}_{j-1}, u^n_{j-1}, t^n$)
  $\mathcal{P}^*_{j-1} \leftarrow \emptyset$ ▷ initialise the set of marked elements
  for $E \in \mathcal{P}_{j-1}$ **do**
    if GETSUBELEMENTS(E) $\cap\, \mathcal{P}_{j-1} \neq \emptyset$ **then**
      $\mathcal{P}^*_{j-1} \leftarrow \mathcal{P}^*_{j-1} \cup E$ ▷ mark E
    else if NEEDSREFINEMENT(E, u^n_{j-1}, t^n) **then**
      $\mathcal{P}^*_{j-1} \leftarrow \mathcal{P}^*_{j-1} \cup E$ ▷ mark E
    end if
  end for
  return $\mathcal{P}^*_{j-1}$
end function

formulated in (9). We assume that $u_S(\cdot, t^n) \approx 0$ in Ω for $n > n_{stop} := n_{j_{stop}}$. We then obtain the approximation

$$U_S \approx U_h^{n_{stop}} := \sum_{n=1}^{n_{stop}} \Delta t e^{i\omega t^n} u_h^n.$$

We can compute $U_h^{n_{stop}}$ by setting $U_h^0 \equiv 0$ in Ω and by using the recursive relation

$$U_h^n = U_h^{n-1} + \Delta t e^{i\omega t^n} u_h^n \qquad \text{for } n = 1, 2, \ldots, n_{stop}. \tag{14}$$

To accurately approximate U_S on the entire computational domain Ω, we need to compute $U_h^{n_{stop}}$ on a globally fine mesh $\mathcal{T}^K$, which means that, if we use the formula in (14), we would need to evaluate u_h^n at each time step at all the degrees of freedom of $\mathcal{T}^K$. Since u_h^n is only known at the degrees of freedom of some adapted mesh $\mathcal{T}_j$, this means we would need to interpolate u_h^n at the degrees of freedom of $\mathcal{T}^K$ at each time step. We can significantly reduce the average number of interpolation points per time step in the computation of $U_h^{n_{stop}}$ by using an adapted space-time mesh.

Let $\mathcal{Q}$ be a space-time mesh for the space-time domain $\Omega \times (T_0, T_{j_{stop}})$ with space-time elements Q of the form $Q = E_Q \times (T_{j_{Q,0}}, T_{j_{Q,1}})$. We choose the space-time elements such that $E_Q \in \mathcal{T}_j$ for $j : j_{Q,0} < j \leq j_{Q,1}$. Let $\chi_Q(\mathbf{x}, t)$ be the characteristic function given by

$$\chi_Q(\mathbf{x}, t) := \begin{cases} \chi_{E_Q}, & t \in (T_{j_{Q,0}}, T_{j_{Q,1}}], \\ 0, & \text{otherwise}, \end{cases} \qquad \chi_E(\mathbf{x}) := \begin{cases} 1, & \mathbf{x} \in E \\ 0, & \text{otherwise}. \end{cases}$$

A discrete version of χ_E is given in Section 4.1.2 below. We have the partition of unity property

$$\sum_{Q \in \mathcal{Q}} \chi_Q(\mathbf{x}, t^n) = 1 \qquad \text{for a.e. } \mathbf{x} \in \Omega, \forall n : 1 \leq n \leq n_{stop}. \tag{15}$$

We can therefore write

$$U_h^{n_{stop}} = \sum_{n=1}^{n_{stop}} \Delta t e^{i\omega t^n} u_h^n \overset{(15)}{=} \sum_{n=1}^{n_{stop}} \left(\sum_{Q\in\mathcal{Q}} \chi_Q(\cdot, t^n) \right) \Delta t e^{i\omega t^n} u_h^n$$

$$= \sum_{Q\in\mathcal{Q}} \left(\sum_{n=1}^{n_{stop}} \chi_Q(\cdot, t^n) \Delta t e^{i\omega t^n} u_h^n \right) =: \sum_{Q\in\mathcal{Q}} \Delta U_Q. \tag{16}$$

Note that ΔU_Q has support only in E_Q. With a slight abuse of notation, we let ΔU_Q also denote its restriction to E_Q. Which definition is used will be clear from the context.

Let $n_{Q,0} := n_{j_{Q,0}}$ and $n_{Q,1} := n_{j_{Q,1}}$. We can compute ΔU_Q by first setting $\Delta U_Q = \Delta U_Q^{n_{Q,1}}$, where $\Delta U_Q^{n_{Q,1}}$is computed by setting $\Delta U_Q^{n_{Q,0}} = 0$ in E_Q and by using the recursive relation

$$\Delta U_Q^n = \Delta U_Q^{n-1} + \Delta t e^{i\omega t^n} u_h^n|_{E_Q} \qquad \text{for } n : n_{Q,0} + 1 \le n \le n_{Q,1}.$$

Note that, in order to compute ΔU_Q, we only need the values of u_h^n at the degrees of freedom corresponding to the spatial element E_Q. To compute $U_h^{n_{stop}} = \sum_{Q\in\mathcal{Q}} \Delta U_Q$, we need to interpolate ΔU_Q at the degrees of freedom of the finest mesh $\mathcal{T}^K$ at E_Q. However, we only need to do this once for each space time element.

To minimise the computational cost, we choose the space-time elements as large as possible. This means that we choose the time intervals $(T_{Q,0}, T_{Q,1})$ as large as possible, namely such that $E_Q \in \mathcal{T}_j$ for all $j : j_{Q,0} < j \le j_{Q,1}$ and such that $E_Q \notin \mathcal{T}_j$ for $j = j_{Q,0}$ and $j = j_{Q,1} + 1$. An illustration of a space-time mesh is given in Figure 4 below.

In practice, we do not need to construct the space-time mesh explicitly. Let $Q(j, E)$ denote the unique space-time element $Q \in \mathcal{Q}$ such that $E = E_Q$ and $j_{Q,0} < j \le j_{Q,1}$. We define $\Delta U_{j,E} := \Delta U_{Q(j,E)}$ and $\Delta U_{j,E}^n := \Delta U_{Q(j,E)}^n$. We also define

$$U_h^{n_j} := \sum_{Q\in\mathcal{Q}:\ j_{Q,1}\le j} \Delta U_Q.$$

We have that $U_h^0 \equiv 0$ and we can compute $U_h^{n_{stop}}$ by computing, for $j = 1, 2, \ldots, j_{stop}$, the following:

$$U_h^{n_j} = U_h^{n_{j-1}} + \sum_{Q\in\mathcal{Q}:\ j_{Q,1}=j} \Delta U_Q,$$

or, equivalently,

$$U_h^{n_j} = U_h^{n_{j-1}} + \sum_{E\in\mathcal{T}_j\setminus\mathcal{T}_{j+1}} \Delta U_{j,E}, \tag{17}$$

with $\mathcal{T}_{j_{stop}+1} := \emptyset$. We can compute $\Delta U_{j,E} = \Delta U_{j,E}^{n_j}$ by first setting $\Delta U_{0,E}^0 \equiv 0$ for all $E \in \mathcal{T}_0$ and by computing, for $j = 1, 2, \ldots, j_{stop}$, the following:

$$\Delta U_{j,E}^n = \Delta U_{j,E}^{n-1} + \Delta t e^{i\omega t^n} u_h^n|_E \qquad \forall E \in \mathcal{T}_j \ \ \&\ \ n : n_{j-1} + 1 \le n \le n_j, \tag{18a}$$

$$\Delta U_{j,E}^{n_{j-1}} \equiv 0 \qquad \forall E \in \mathcal{T}_j \setminus \mathcal{T}_{j-1}, \tag{18b}$$

$$\Delta U_{j,E}^{n_{j-1}} = \Delta U_{j-1,E}^{n_{j-1}} \qquad \forall E \in \mathcal{T}_j \cap \mathcal{T}_{j-1}. \tag{18c}$$

Therefore, letting $\Delta U_j^n := \{\Delta U_{j,E}^n\}_{E\in\mathcal{T}_j}$, the algorithm for computing $U_h^{n_{stop}}$ is given in Algorithm 4, which consists of the following functions:

- U_h = UPDATEFT$(U_h, \Delta U_{j-1}^n, \mathcal{T}_{j-1}, \mathcal{T}_j)$: computes $U_h|_E \leftarrow U_h|_E + \Delta U_{j-1,E}^n$ for all $E \in \mathcal{T}_{j-1} \setminus \mathcal{T}_j$.
- ΔU_j^n = INITIALISENEWINCREMENTS$(\Delta U_{j-1}^n, \mathcal{T}_{j-1}, \mathcal{T}_j)$: sets $\Delta U_{j,E}^n \leftarrow 0$ for all $E \in \mathcal{T}_j \setminus \mathcal{T}_{j-1}$ and $\Delta U_{j,E}^n \leftarrow \Delta U_{j-1,E}^n$ for all $E \in \mathcal{T}_j \cap \mathcal{T}_{j-1}$.

- $\Delta U_j^n = \text{UPDATEINCREMENTS}(\Delta U_j^{n-1}, u_h^n, \mathcal{T}_j, t^n)$: computes $U_{j,E}^n = U_{j,E}^{n-1} + \Delta t e^{i\omega t^n} u_h^n|_E$ for all $E \in \mathcal{T}_j$.

Algorithm 4 compute the discrete Fourier transform with respect to time

function COMPUTEFT($\{u_h^n\}_{n=1}^{n_{stop}}, \{\mathcal{T}_j\}_{j=1}^{j_{stop}}$)
  $U_h \leftarrow 0$ ▷ initialise Fourier transform
  $\Delta U_h \leftarrow 0$ ▷ initialise increments
  for $j = 1, 2, \ldots, j_{stop}$ **do**
    $n \leftarrow n_{j-1}$ ▷ at this point, $U_h = U_h^{n_{j-2}}$, $\Delta U_h = \Delta U_{j-1}^n$
    $U_h \leftarrow \text{UPDATEFT}(U_h, \Delta U_h, \mathcal{T}_{j-1}, \mathcal{T}_j)$ ▷ $U_h \leftarrow U_h^n$
    $\Delta U_h \leftarrow \text{INITIALISENEWINCREMENTS}(\Delta U_h, \mathcal{T}_{j-1}, \mathcal{T}_j)$ ▷ $\Delta U_h \leftarrow \Delta U_j^n$
    for $\ell = 1, 2, \ldots, m$ **do**
      $n \leftarrow n_{j-1} + \ell$
      $\Delta U_h \leftarrow \text{UPDATEINCREMENTS}(\Delta U_h, u_h^n, \mathcal{T}_j, t^n)$ ▷ $\Delta U_h \leftarrow \Delta U_j^n$
    end for
  end for
  $U_h \leftarrow \text{UPDATEFT}(U_h, \Delta U_h, \mathcal{T}_{j_{stop}}, \emptyset)$ ▷ $U_h \leftarrow U_h^{n_{stop}}$
  return U_h
end function

3.4. **Overview of the complete algorithm.** To approximate the solution U_S to the Helmholtz equation given in (2), we approximate the solution u_S to the time-dependent wave equation in (4) using Algorithm 1 and then approximate the Fourier transform $\mathfrak{F}_t[u_S](\cdot, -\omega)$ using Algorithm 4. We can solve the wave equation and compute the Fourier transform simultaneously, resulting in Algorithm 5. This last algorithm gives a complete overview of the proposed method for solving the Helmholtz equation.

4. Numerical Examples

We present numerical examples for wave scattering problems in 1 and 2 spatial dimensions. Details of the finite element discretisation are provided for the 2-dimensional case. The discretisation for the 1-dimensional case can be readily deduced from the 2-dimensional case. All the numerical experiments presented in this section have been carried out in MATLAB R2017a.

4.1. **Absorbing boundary layer and finite element discretisation.** We start by specifying how we impose an absorbing boundary layer and then provide details of the finite element discretisation.

4.1.1. *Absorbing boundary layer.* We consider the wave equation in (6) for a rectangular domain and add an additional absorbing boundary layer. Let $\Omega_0 = (-L_1, L_1) \times (-L_2, L_2) \supset \Omega_{in}$ be the region of interest, and let Ω_{ABL} be an additional absorbing boundary layer of width W surrounding Ω_0. We define the computational domain by $\Omega = \Omega_0 \cup \Omega_{ABL} = (-L_1 - W, L_1 + W) \times (-L_2 - W, L_2 + W)$. We apply the perfectly matched layer that was introduced in [28] and further analysed in [32, 7]. The resulting wave equation is given by

$$\partial_t^2 u + (\zeta_1 + \zeta_2)\partial_t u + \zeta_1\zeta_2 u - \beta^{-1}\nabla \cdot (\alpha(\nabla u + \mathbf{s})) = f \qquad \text{in } \Omega \times (t_0, \infty), \tag{19a}$$

$$\partial_t \mathbf{s} + Z_1 \mathbf{s} + Z_2 \nabla u = \mathbf{0} \qquad \text{in } \Omega \times (t_0, \infty), \tag{19b}$$

$$u(\cdot, t_0) = \partial_t u(\cdot, t_0) = 0 \qquad \text{in } \Omega, \tag{19c}$$

$$\mathbf{s}(\cdot, t_0) = \mathbf{0} \qquad \text{in } \Omega, \tag{19d}$$

$$u = 0 \qquad \text{on } \partial\Omega \times (t_0, \infty), \tag{19e}$$

Algorithm 5 solving the Helmholtz equation using a time-domain approach

procedure SOLVEHELMHOLTZEQUATION
  $\mathcal{T}_h \leftarrow \mathcal{T}^K$ ▷ set initial mesh
  $u_h \leftarrow 0$, $u_h^{new} \leftarrow 0$, and $u_h^{old} \leftarrow 0$ ▷ initialise wave field
  $U_h \leftarrow 0$ ▷ initialise Fourier transform
  $\Delta U_h \leftarrow 0$ ▷ initialise increments
  for $j = 1, 2, \ldots$ **do**
    $n = n_{j-1}$ ▷ at this point, $\mathcal{T}_h = \mathcal{T}_{j-1}$, $u_h = u_{j-1}^n, u_h^{old} = u_{j-1}^{n-1}$
    ▷ also, at this point, $U_h = U_h^{n_{j-2}}$, $\Delta U_h = \Delta U_{j-1}^n$
    if STOP$(u_h, \mathcal{T}_h)$ **then**
      $U_h \leftarrow$ UPDATEFT$(U_h, \Delta U_h, \mathcal{T}_h, \emptyset)$ ▷ $U_h \leftarrow U_h^n$
      return U_h
    end if
    $\mathcal{T}_h^{new} \leftarrow$ UPDATEMESH$(\mathcal{T}_h, u_h, t^n)$ ▷ $\mathcal{T}_h^{new} \leftarrow \mathcal{T}_j$
    $U_h \leftarrow$ UPDATEFT$(U_h, \Delta U_h, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $U_h \leftarrow U_h^n$
    $\Delta U_h \leftarrow$ INITIALISENEWINCREMENTS$(\Delta U_h, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $\Delta U_h \leftarrow \Delta U_j^n$
    $u_h \leftarrow$ PROJECT$(u_h, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $u_h \leftarrow u_j^n$
    $u_h^{old} \leftarrow$ PROJECT$(u_h^{old}, \mathcal{T}_h, \mathcal{T}_h^{new})$ ▷ $u_h^{new} \leftarrow u_j^{n-1}$
    $\mathcal{T}_h \leftarrow \mathcal{T}_h^{new}$ ▷ $\mathcal{T}_h \leftarrow \mathcal{T}_j$
    for $\ell = 0, 1, 2, \ldots, m-1$ **do**
      $n \leftarrow n_{j-1} + \ell$ ▷ at this point, $u_h = u_j^n, u_h^{old} = u_j^{n-1}, \Delta U_h = \Delta U_j^n$
      $u_h^{new} \leftarrow$ DOTIMESTEP$(u_h, u_h^{old}, \mathcal{T}_h, t^n)$ ▷ $u_h^{new} \leftarrow u_j^{n+1}$
      $u_h^{old} \leftarrow u_h$ ▷ $u_h^{old} \leftarrow u_j^n$
      $u_h \leftarrow u_h^{new}$ ▷ $u_h \leftarrow u_j^{n+1}$
      $\Delta U_h \leftarrow$ UPDATEINCREMENTS$(\Delta U_h, u_h, \mathcal{T}_h, t^{n+1})$ ▷ $\Delta U_h \leftarrow \Delta U_j^{n+1}$
    end for
  end for
end procedure

with

$$Z_1 := \begin{bmatrix} \zeta_1 & 0 \\ 0 & \zeta_2 \end{bmatrix} \text{ and } Z_2 := \begin{bmatrix} \zeta_1 - \zeta_2 & 0 \\ 0 & \zeta_2 - \zeta_1 \end{bmatrix}.$$

Here, $\mathbf{s} = \mathbf{s}(\mathbf{x}) = (s_1(\mathbf{x}), s_2(\mathbf{x}))$ is an auxiliary vector field that has support only in Ω_{ABL}, and $\zeta_1 = \zeta_1(x_1) \geq 0$ and $\zeta_2 = \zeta_2(x_2) \geq 0$ are two additional parameters that are nonzero only in Ω_{ABL}. In particular, $\zeta_1(x_1)$ is nonzero only for $x_1 : L_1 \leq |x_1| \leq L_1 + W$ and $\zeta_2(x_2)$ is nonzero only for $x_2 : L_2 \leq |x_2| \leq L_2 + W$.

4.1.2. *Finite element discretisation.* To construct the adapted meshes, we use a set of nested rectangular meshes $\{\mathcal{T}^k\}_{k=1}^K$ that are constructed as follows. First, we define a sequence of mesh widths $h_1 > h_2 > \cdots > h_K$. Mesh $\mathcal{T}^k$ is then constructed using elements of size $h_{k,1} \times h_{k,2}$, where $h_{k,i} = h_k$ in the region $|x_i| < L_i$ and $h_{k,i} = h_K$ in the region $|x_i| > L_i$, for $i = 1, 2$. In other words, $\mathcal{T}^k$ is a Cartesian mesh of width h_k in the main domain Ω_0, but has thin elements of the finest resolution h_K in the absorbing boundary layer Ω_{ABL}. An illustration of the nested meshes is given in Figure 2.

To construct the conforming finite element space $\mathcal{U}_\mathcal{T}$ for a given mesh $\mathcal{T}$, we use tensor-product polynomials of degree at most p, namely the polynomial reference space is $\hat{\mathcal{U}} = \text{span}\{x_1^k x_2^\ell \mid k, \ell \leq p\}$. As degrees of freedom, we use the values at the nodes $\mathcal{X}_\mathcal{T}$, where $\mathcal{X}_\mathcal{T}$ consists of the $(p+1) \times (p+1)$ tensor-product Gauss–Lobatto points of each element $E \in \mathcal{T}$. In case $\mathcal{T}$ has a hanging node, $\mathcal{X}_\mathcal{T}$ contains the Gauss–Lobatto points corresponding to the

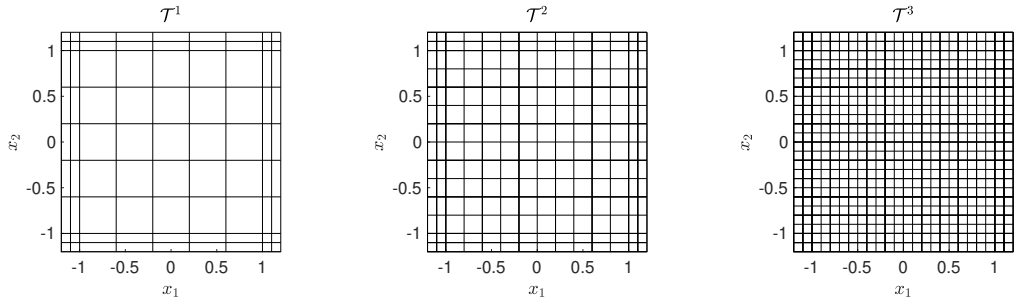

FIGURE 2. Illustration of $\mathcal{T}^k$, $k = 1, 2, 3$, with $\{h_k\} = \{\frac{2}{5}, \frac{1}{5}, \frac{1}{10}\}$, $\Omega_0 = (-1, 1)^2$, $W = \frac{1}{5}$.

coarsest edge adjacent to the hanging node, but not the points corresponding to the finer edges adjacent to the hanging node.

For the discretisation of $\mathbf{s}$, we use the discontinuous finite element space $\mathcal{S}^2_{\mathcal{T}}$, where $\mathcal{S}_{\mathcal{T}}$ is given by

$$\mathcal{S}_{\mathcal{T}} := \{s \in L^2(\Omega) \mid s|_{\Omega_0} \equiv 0 \text{ and } s \circ \phi_E \in \hat{\mathcal{U}} \text{ for all } E \in \mathcal{T} : E \subset \Omega_{ABL}\}.$$

Let $\mathbf{s}_E := \mathbf{s}|_E$. The degrees of freedom of $\mathbf{s} \in \mathcal{S}^2_{\mathcal{T}}$ are given by $\mathbf{s}_E(\mathbf{x})$ for all $E \in \mathcal{T}$ and all $\mathbf{x} \in \mathcal{X}_E$, where $\mathcal{X}_E := \mathcal{X}_{\{E\}}$ denotes the set of nodes on E.

We define the discretisation $\mathcal{L}_{\mathcal{T}}(u, \mathbf{s}) : (\mathcal{U}_{\mathcal{T}}, \mathcal{S}^2_{\mathcal{T}}) \to \mathcal{U}_{\mathcal{T}}$ of the spatial operator $(u, \mathbf{s}) \mapsto -\beta^{-1}\nabla \cdot (\alpha(\nabla u + \mathbf{s}))$ such that

$$(\beta \mathcal{L}_{\mathcal{T}}(u, \mathbf{s}), w)_{\mathcal{T},ML} = (\alpha(\nabla u + \mathbf{s}), \nabla w)_{\mathcal{T}} \qquad \forall w \in \mathcal{U}_{\mathcal{T}}.$$

Here, $(\cdot, \cdot)_{\mathcal{T}}$ denotes the approximation of the L^2 inner product $(\cdot, \cdot)$ using the tensor-product $(p+1)$-point Gauss–Lobatto quadrature rule for each element in $\mathcal{T}$. Furthermore, $(\cdot, \cdot)_{\mathcal{T},ML}$ denotes the approximation of $(\cdot, \cdot)$ using a mass-lumping technique, i.e.

$$(u, w)_{\mathcal{T},ML} = \sum_{\mathbf{x} \in \mathcal{X}_{\mathcal{T}}} u(\mathbf{x}) w(\mathbf{x}) \sigma_{\mathbf{x},\mathcal{T}},$$

where $\sigma_{\mathbf{x},\mathcal{T}} := \int_\Omega w_{\mathbf{x},\mathcal{T}}(\mathbf{y})\, \mathrm{d}\mathbf{y}$ and $w_{\mathbf{x},\mathcal{T}} \in \mathcal{U}_{\mathcal{T}}$ denotes the nodal basis function corresponding to node $\mathbf{x}$. We can give an explicit expression for $\mathcal{L}_{\mathcal{T}}$:

$$\mathcal{L}_{\mathcal{T}}(u, \mathbf{s})(\mathbf{x}) = \frac{\big(\alpha(\nabla u + \mathbf{s}), \nabla w_{\mathbf{x},\mathcal{T}}\big)_{\mathcal{T}}}{\beta(\mathbf{x}) \sigma_{\mathbf{x},\mathcal{T}}} \qquad \forall \mathbf{x} \in \mathcal{X}_{\mathcal{T}}.$$

For the projection operators, let $\mathcal{T}_{j-1} + \mathcal{T}_j$ denote the mesh constructed from elements of $\mathcal{T}_{j-1}$ and $\mathcal{T}_j$ by always selecting the finest elements. We define the projection operators $\Pi_j : \mathcal{U}_{\mathcal{T}_{j-1}} \to \mathcal{U}_{\mathcal{T}_j}$ and $\Pi^S_j : \mathcal{S}_{\mathcal{T}_{j-1}} \to \mathcal{S}_{\mathcal{T}_j}$ in such a way that

$$\begin{aligned} (\Pi_j u, w)_{\mathcal{T}_j,ML} &= (u, w)_{\mathcal{T}_{j-1}+\mathcal{T}_j} && \forall w \in \mathcal{U}_{\mathcal{T}_j}, \\ (\Pi^S_j s, w)_{\mathcal{T}_j} &= (s, w)_{\mathcal{T}_{j-1}+\mathcal{T}_j} && \forall w \in \mathcal{S}_{\mathcal{T}_j}. \end{aligned}$$

Furthermore, let $E \in \mathcal{T}^k$ with $k \leq K - 1$, and let $\mathcal{T}_E = \textsc{getSubelements}(E)$. We define the projection operator $\Pi_E : \mathcal{U}_E := \mathcal{U}_{\{E\}} \to \mathcal{U}_{\mathcal{T}_E}$, used for the refinement criterion in (13), in such a way that

$$(\Pi_E u, w)_{\{E\}} = (u, w)_{\mathcal{T}_E} \qquad \forall w \in \mathcal{U}_E.$$

We can give explicit expressions for these projection operators:

$$\Pi_j u(\mathbf{x}) = \frac{\big(u, w_{\mathbf{x},\mathcal{T}_j}\big)_{\mathcal{T}_{j-1}+\mathcal{T}_j}}{\sigma_{\mathbf{x},\mathcal{T}_j}} \qquad \forall \mathbf{x} \in \mathcal{X}_{\mathcal{T}_j},$$

$$(\Pi_j^S s)_E(\mathbf{x}) = \frac{\big(s, w_{\mathbf{x},E}\big)_{\mathcal{T}_{j-1}+\mathcal{T}_j}}{\sigma_{\mathbf{x},E}} \qquad \forall E \in \mathcal{T}_j \text{ and } \mathbf{x} \in \mathcal{X}_E,$$

$$\Pi_E u(\mathbf{x}) = \frac{(u, w_{\mathbf{x},E})_{\mathcal{T}_E}}{\sigma_{\mathbf{x},E}} \qquad \forall \mathbf{x} \in \mathcal{X}_E.$$

Here $\sigma_{\mathbf{x},E} := \int_E w_{\mathbf{x},E}(\mathbf{y})\,\mathrm{d}\mathbf{y}$ and $w_{\mathbf{x},E} \in \mathcal{S}_E$ denotes the discontinuous nodal basis function corresponding to element E and node $\mathbf{x}$.

For the time discretisation, let $\mathbf{s}^n := \mathbf{s}(\cdot, t^n)$. We define $u^{n+1/2} := \frac{1}{2}(u^n + u^{n+1})$ and $\mathbf{s}^{n+1/2} := \frac{1}{2}(\mathbf{s}^n + \mathbf{s}^{n+1})$. We also define the discrete second-order time derivative $D_t^2 u_j^n$ as in (8) and we define the discrete time derivatives $D_{2t}u^n$ and $D_t\mathbf{s}^{n+1/2}$ as

$$D_{2t}u^n := \frac{u^{n+1} - u^{n-1}}{2\Delta t}, \quad D_t\mathbf{s}^{n+1/2} := \frac{\mathbf{s}^{n+1} - \mathbf{s}^n}{\Delta t}.$$

The fully discrete finite element formulation can then be stated as follows: for $j = 1, 2, \ldots,$ find $u_j^n \in \mathcal{U}_{\mathcal{T}_j}$ for $n : n_{j-1} - 1 \le n \le n_j$ and $\mathbf{s}_j^n \in \mathcal{S}_{\mathcal{T}_j}$ for $n : n_{j-1} \le n \le n_j$ such that

$$D_t^2 u_j^n + (\zeta_1 + \zeta_2) D_{2t} u_j^n + \zeta_1\zeta_2 u_j^n + \mathcal{L}_{\mathcal{T}_j}(u_j^n, \mathbf{s}_j^n) = f_{\mathcal{T}_j}(\cdot, t^n) \qquad \text{at all } \mathbf{x} \in \mathcal{X}_{\mathcal{T}_j}, \tag{20a}$$

$$D_t\mathbf{s}_{j,E}^{n+1/2} + Z_1\mathbf{s}_{j,E}^{n+1/2} + Z_2\nabla u_j^{n+1/2}|_E = 0 \qquad \begin{array}{l}\text{at all } \mathbf{x} \in \mathcal{X}_E,\\ \forall E \in \mathcal{T}_j,\end{array} \tag{20b}$$

for $n : n_{j-1} + 1 \le n \le n_j$, and

$$\begin{aligned} u_j^n &= \Pi_j u_{j-1}^n && \text{for } n = n_{j-1} - 1 \text{ and } n = n_{j-1},\\ \mathbf{s}_j^n &= \Pi_j^S \mathbf{s}_{j-1}^n && \text{for } n = n_{j-1}, \end{aligned}$$

with $u_0^0 \equiv 0$, $u_0^{-1} \equiv 0$, and $\mathbf{s}^0 \equiv \mathbf{0}$.

We can rewrite (20) as

$$u_j^{n+1} = \frac{\tilde{z}_1 u_j^{n-1} + \tilde{z}_2 u_j^n + \Delta t^2(-\mathcal{L}_{\mathcal{T}_j}(u_j^n, \mathbf{s}_j^n) + f_{\mathcal{T}_j}(\cdot, t^n))}{\tilde{z}_3} \qquad \text{at all } \mathbf{x} \in \mathcal{X}_{\mathcal{T}_j} \tag{21a}$$

$$\mathbf{s}_{j,E}^{n+1} = \tilde{Z}_3^{-1}\big(\tilde{Z}_1\mathbf{s}_{j,E}^n + \tilde{Z}_2\nabla u_j^{n+1/2}|_E\big) \qquad \begin{array}{l}\text{at all } \mathbf{x} \in \mathcal{X}_E,\\ \forall E \in \mathcal{T}_j,\end{array} \tag{21b}$$

where

$$\begin{aligned} \tilde{z}_1 &:= -1 + \frac{1}{2}\Delta t(\zeta_1 + \zeta_2), & \tilde{Z}_1 &:= I - \frac{1}{2}\Delta t Z_1,\\ \tilde{z}_2 &:= 2 - \Delta t^2\zeta_1\zeta_2, & \tilde{Z}_2 &:= -\Delta t Z_2,\\ \tilde{z}_3 &:= 1 + \frac{1}{2}\Delta t(\zeta_1 + \zeta_2), & \tilde{Z}_3 &:= I + \frac{1}{2}\Delta t Z_1, \end{aligned}$$

with $I \in \mathbb{R}^{2\times 2}$ the identity matrix.

For the discretisation of the source term f, recall that $f = -\partial_t^2 u_I + \beta^{-1}\nabla\cdot(\alpha\nabla u_I)$. One can check that $-\partial_t^2 u_I + \beta_0^{-1}\nabla\cdot(\alpha_0\nabla u_I) \equiv 0$ and therefore we have $-\beta_0\beta^{-1}\partial_t^2 u_I + \beta^{-1}\nabla\cdot(\alpha_0\nabla u_I) \equiv 0$. We can therefore write $f = -(\beta - \beta_0)\beta^{-1}\partial_t^2 u_I + \beta^{-1}\nabla\cdot((\alpha - \alpha_0)\nabla u_I)$. We can discretise the time- and spatial derivatives in a similar way as before. The discrete source term $f_{\mathcal{T}}(\cdot, t^n) \in \mathcal{U}_{\mathcal{T}}$ can then be given by

$$f_{\mathcal{T}}(\mathbf{x}, t^n) := -\frac{\beta(\mathbf{x}) - \beta_0}{\beta(\mathbf{x})} D_t^2 u_I^n(\mathbf{x}) - \frac{\big((\alpha - \alpha_0)\nabla u_I^n, \nabla w_{\mathbf{x},\mathcal{T}}\big)_{\mathcal{T}}}{\beta(\mathbf{x})\sigma_{\mathbf{x},\mathcal{T}}} \qquad \forall \mathbf{x} \in \mathcal{X}_{\mathcal{T}},$$

where $u_I^n := u_I(\cdot, t^n)$. Note that, due to this discretisation, the discrete source term is still zero in the exterior domain Ω_{ex} and for all $t^n > t_f$.

To discretise the characteristic function χ_E for a square element $E = (x_{1,E}, x_{1,E} + h_E) \times (x_{2,E}, x_{2,E} + h_E)$, we define

$$\chi_E(\mathbf{x}) = \begin{cases} 1, & x_1 \in [x_{1,E}, x_{1,E} + h_E) \cup (\{x_{1,E} + h_E\} \cap \{L_1 + W\}) \text{ and} \\ & x_2 \in [x_{2,E}, x_{2,E} + h_E) \cup (\{x_{2,E} + h_E\} \cap \{L_2 + W\}), \\ 0, & \text{otherwise.} \end{cases} \tag{22}$$

Then the partition of unity property (15) is valid for every $\mathbf{x} \in \Omega$ (not just a.e. $\mathbf{x} \in \Omega$) and, in particular, for all nodes $\mathbf{x} \in \mathcal{X}_{\mathcal{T}^K}$.

4.1.3. *Algorithm for solving the Helmholtz equation.* To solve the Helmholtz equation given in (2), we use Algorithm 5 with a few small modifications given below. These modifications take the absorbing boundary layer into account and ensure that all the steps are fully computable.

- At the start, we also initialise the auxiliary variables $\mathbf{s}_h \leftarrow \mathbf{0}$.
- After we compute $u_h \leftarrow \text{PROJECT}(u_h, \mathcal{T}_h, \mathcal{T}_h^{new})$, we also compute the auxiliary variable $\mathbf{s}_h \leftarrow \text{PROJECTS}(\mathbf{s}_h, \mathcal{T}_h, \mathcal{T}_h^{new})$, where the function $\mathbf{s}_j^n = \text{PROJECTS}(\mathbf{s}_{j-1}^n, \mathcal{T}_{j-1}, \mathcal{T}_j)$ computes $\mathbf{s}_j^n = \Pi_j^S \mathbf{s}_{j-1}^n$.
- Instead of computing $u_h^{new} \leftarrow \text{DOTIMESTEP}(u_h, u_h^{old}, \mathcal{T}_h, t^n)$, we now compute the pair $(u_h^{new}, \mathbf{s}_h) \leftarrow \text{DOTIMESTEP}(u_h, u_h^{old}, \mathbf{s}_h, \mathcal{T}_h, t^n)$, where we use the modified function $(u_j^{n+1}, \mathbf{s}_j^{n+1}) = \text{DOTIMESTEP}(u_j^n, u_j^{n-1}, \mathbf{s}_j^n, \mathcal{T}_j, t^n)$ that computes u_j^{n+1} and $\mathbf{s}_j^{n+1}$ with the formulae in (21).
- For the stopping criterion given in (11), we do not take the supremum over all $\mathbf{x} \in \Omega$, but instead we compute the supremum over all nodes $\mathbf{x} \in \mathcal{X}_{\mathcal{T}_j}$. Similarly, for the refinement criterion given in (13), we do not take the supremum over all $\mathbf{x} \in E$, but only over all nodes $\mathbf{x} \in \mathcal{X}_{\mathcal{T}_E}$, where $\mathcal{T}_E = \text{GETSUBELEMENTS}(E)$.
- For the function $U_h = \text{UPDATEFT}(U_h, \Delta U_{j-1}^n, \mathcal{T}_{j-1}, \mathcal{T}_j)$, we now compute $U_h(\mathbf{x}) \leftarrow U_h(\mathbf{x}) + \chi_E(\mathbf{x}) \Delta U_{j-1,E}^n(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}_{\mathcal{T}^K} \cap \overline{E}$ and all $E \in \mathcal{T}_{j-1} \setminus \mathcal{T}_j$, with χ_E defined as in (22).

4.2. **Numerical example in 1D.** As a first numerical example, we consider the 1D domain $\Omega = (-1, 1)$, with spatial parameters

$$\alpha(x) = \begin{cases} 1 + 3(1-2x)^2(1+2x)^2, & x \in (-\frac{1}{2}, \frac{1}{2}), \\ 1, & \text{otherwise,} \end{cases}$$
$$\beta(x) = 1,$$

and an incoming plane wave $U_I(x) = e^{i\omega x/c_0}$, with $c_0 = 1$. An illustration of α is given in Figure 3.

For the numerical approximation, we consider an incoming wavelet of the form $u_I(x,t) = \omega\psi(\omega(t - x/c_0))$, with

$$\psi(\xi) = \begin{cases} \dfrac{(\xi - \pi)^4(\xi + \pi)^4}{3840\pi(21 - 2\pi^2)}, & \xi \in (-\pi, \pi), \\ 0, & \text{otherwise.} \end{cases}$$

An illustration of ψ is given in Figure 3.

At the boundary, we apply a perfectly matched layer of width $W = c_0\pi/\omega$, i.e. of width half a wave length. The equations for the absorbing boundary layer in 1D are the same as those in 2D given in (19), but with $\zeta_2 \equiv 0$ and with a scalar field $s(x)$ instead of $\mathbf{s}(\mathbf{x})$. We

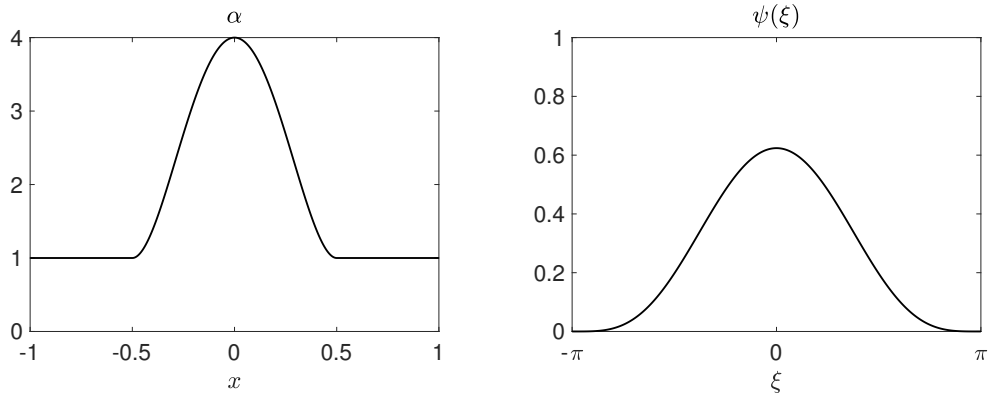

FIGURE 3. Illustration of parameter $\alpha(x)$ (left) and wavelet $\psi(\xi)$ (right).

choose the damping parameter $\zeta_1(x) = \zeta(x)$ as in [13], where

$$\zeta(x) = \begin{cases} |\log(R)|\frac{3}{2W}\left(\frac{W-(1-|x|)}{W}\right)^2, & |x| > 1 \\ 0, & \text{otherwise,} \end{cases} \tag{23}$$

with $R = 10^{-10}$ the expected artificial reflection.

For the time step size, we choose $\Delta t = T_{up}/m$, with m the smallest positive integer such that

$$\Delta t = \frac{T_{up}}{m} \leq c_{CFL}\frac{2}{\sqrt{\lambda^*_{\max}}},$$

where the CFL number is chosen as $c_{CFL} = 0.9$, and where $\lambda^*_{\max}$ is an upper bound of the largest eigenvalue $\lambda_{\max}$ of the discrete spatial differential operator $\beta^{-1}\nabla\cdot\alpha\nabla$, given by

$$\lambda^*_{\max} := \frac{1}{h_K^2}\frac{\alpha_{\max}}{\beta_{\min}}\sup_{u\in\hat{U}\setminus\{0\}}\frac{(\nabla u,\nabla u)_{\{\hat{E}\}}}{(u,u)_{\{\hat{E}\}}},$$

where $\alpha_{\max} := \sup_{x\in\Omega}\alpha(x)$, and $\beta_{\min} := \inf_{x\in\Omega}\beta(x)$. A smaller value of T_{up} results in fewer neighbouring elements being marked for refinement, and therefore in fewer degrees of freedom on average. However, it also means that the mesh needs to be updated more frequently. In other words, choosing T_{up} very small might slow down the method due to computational overhead, while choosing T_{up} very large might render the method less efficient due to many additional neighbouring elements being refined. In the numerical examples, we choose T_{up} as half a time-period, i.e. $T_{up} = \pi\omega^{-1}$.

For the spatial discretisation, we use quadratic elements (so degree $p = 2$). We compute for the time interval $(t_0, t_{stop}) = (t_0, T_{j_{stop}})$, where $t_0 = -0.5-\pi\omega^{-1}$ and where j_{stop} is determined using the stopping criterion in (11) with $\epsilon_0 = \omega/100$. For the mesh refinement criterion in (13), we use $\eta_0 = \omega/100$. An overview of the $L^2(\Omega_0)$ error $err_2 := \|U_S - U_h(\cdot, T_{j_{stop}})\|_{\Omega_0}$ and average number of degrees of freedom $\overline{n}_{DOF} := \frac{1}{j_{stop}}\sum_{j=1}^{j_{stop}}|\mathcal{X}(\mathcal{T}_j)|$ is given in Table 1 for the adapted finite element method. The results are compared with the classical finite element method using a uniform mesh of width $h = h_K$ and using the same polynomial degree, time step size, and stopping time as for the adapted finite element method. From this table, we can see that the error of the adapted finite element method and classical finite element method behave very similarly, whereas the average number of degrees of freedom grows at a significantly slower rate for the adapted finite element method as the frequency ω increases. In particular, Tables 1 and 4 illustrate that the average number of degrees of freedom is almost independent of ω for the adaptive finite element method in 1D.

	AFEM						FEM	
ω	$\{h_k\}$	T_{up}	m	j_{stop}	err_2	$\overline{n}_{DOF}$	err_2	$\overline{n}_{DOF}$
10π	$\{\frac{1}{5}, \frac{1}{50}\}$	$\frac{1}{10}$	28	20	1.40e-02	1.14e+02	1.00e-02	2.21e+02
20π	$\{\frac{1}{5}, \frac{1}{10}, \frac{1}{100}\}$	$\frac{1}{20}$	28	36	1.67e-02	1.43e+02	1.71e-02	4.21e+02
40π	$\{\frac{1}{5}, \frac{1}{20}, \frac{1}{200}\}$	$\frac{1}{40}$	28	63	3.92e-02	1.78e+02	3.93e-02	8.21e+02
80π	$\{\frac{1}{5}, \frac{1}{40}, \frac{1}{400}\}$	$\frac{1}{80}$	28	123	6.49e-02	2.12e+02	5.92e-02	1.62e+03

TABLE 1. Estimated $L^2(\Omega_0)$ error and average number of degrees of freedom for the quadratic adapted (AFEM) and classical (FEM) finite element approximation to the 1D Helmholtz problem for different angular frequencies ω. To estimate the error, we take the numerical approximation on a uniform mesh of width $h_K/2$ as reference solution.

To compute the errors in Table 1, we use the discrete solution obtained on a uniform mesh of width $h_{K/2}$ as reference solution. However, this error does not take into account the error produced by the absorbing boundary layer or the truncation of the wave field at time t_{stop}. To measure these errors, we compute the following numerical approximations and reference solutions.

- U_h: the adapted finite element approximation considered in Table 1.
- U^1: the reference solution used in Table 1.
- U^2: similar to U^1, but using the exact absorbing boundary condition $\partial_t u + c_0 \partial_x u = 0$ on $\partial\Omega_0$ instead of an absorbing boundary layer.
- U^3: similar to U^2, but using a time interval $(t_0, t_0 + 100)$ instead of (t_0, t_{stop}) ($t_{stop} - t_0 < 2.5$ for all cases in Table 1).

The difference $U_h - U^1$ indicates the error due to the spatial and time discretisation, $U^1 - U^2$ indicates the error due to the absorbing boundary layer, and $U^2 - U^3$ indicates the error due to the truncation in time. An overview of these errors is given in Table 2. From this table, we can see that the error is dominated by the discretisation error, whereas the errors due to the absorbing boundary layer and truncation in time are negligible. We will use this as a motivation to also estimate the error by $\|U_h - U^1\|_{\Omega_0}$ in the 2D case.

ω	$\|U_h - U^1\|_{\Omega_0}$	$\|U^1 - U^2\|_{\Omega_0}$	$\|U^2 - U^3\|_{\Omega_0}$
10π	1.40e-02	2.42e-04	4.50e-03
20π	1.67e-02	2.00e-04	3.45e-03
40π	3.92e-02	1.38e-04	3.94e-03
80π	6.49e-02	1.21e-04	2.12e-03

TABLE 2. Estimated $L^2(\Omega_0)$ error due to the spatial and time discretisation $\|U_h - U^1\|_{\Omega_0}$, due to the absorbing boundary layer $\|U^1 - U^2\|_{\Omega_0}$, and due to the truncation in time $\|U^2 - U^3\|_{\Omega_0}$ for the 1D test cases.

An illustration of u_h for the case $\omega = 40\pi$ and an illustration of the corresponding space-time mesh as described in Section 3.3 are given in Figure 4.

4.3. **Numerical example in 2D: incoming plane wave.** For the first 2D example, we consider a domain $\Omega_0 = (-1, 1)^2$, with spatial parameters

$$\alpha(x_1, x_2) = \begin{cases} 1 + 3(1 - 2\sqrt{x_1^2 + x_2^2})^2(1 + 2\sqrt{x_1^2 + x_2^2})^2, & \sqrt{x_1^2 + x_2^2} \leq \frac{1}{2}, \\ 1, & \text{otherwise}, \end{cases}$$

$$\beta(x_1, x_2) = 1,$$

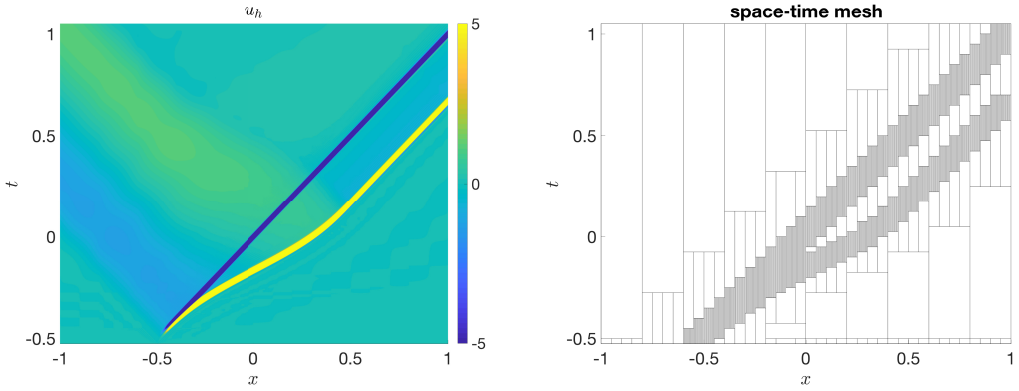

FIGURE 4. Illustration of u_h and the space-time mesh described in Section 3.3 for the 1D test case with $\omega = 40\pi$. In the actual algorithm, we never explicitly compute the space-time mesh.

and an incoming plane wave $U_I(x_1, x_2) = e^{i\omega x_1/c_0}$, with $c_0 = 1$. An illustration of α is given in Figure 5.

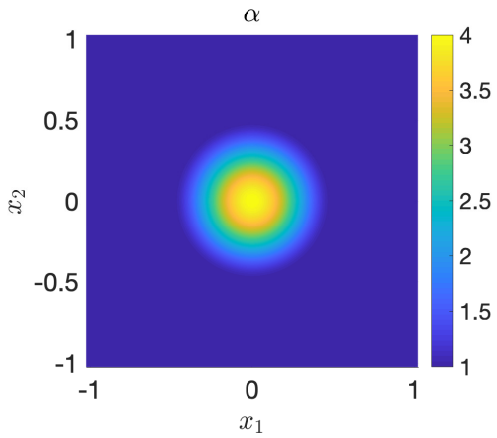

FIGURE 5

For the numerical approximation, we consider an incoming wavelet of the form $u_I(x_1, x_2, t) = \omega\psi(\omega(t - x_1/c_0))$, with ψ as in the 1D example. At the boundary, we apply a perfectly matched layer of width $W = c_0\pi/\omega$, i.e. of width half a wave length. We choose the damping parameters $\zeta_1(x_1) = \zeta(x_1)$ and $\zeta_2(x_2) = \zeta(x_2)$, with $\zeta(x)$ as in the 1D example.

We compute for the time interval $(t_0, t_{stop}) = (t_0, T_{j_{stop}})$, where $t_0 = -0.5 - \pi\omega^{-1}$. For the spatial discretisation, we use biquadratic elements. The time step size Δt and the parameters η_0 and ϵ_0 are chosen in the same way as in the 1D case. We compute the $L^2(\Omega_0)$ error and compare the results with a classical finite element method with a uniform mesh in the same way as we did for the 1D case. The results are presented in Table 3. From this table, we can see that, just as in the 1D case, the errors of the adaptive finite element method and classical finite element method behave similarly, whereas the average number of degrees of freedom grows at a significantly slower rate for the adapted finite element method as ω increases. In particular, Table 4 illustrates that for the adaptive finite element method, the average number of degrees of freedom grows almost linearly with ω instead of as ω^2.

Snapshots of the time-dependent wave field for different frequencies are shown in Figure 6. This figure shows that the wavefront gets sharper as ω increases, while away from the wavefront, e.g. on the left of $x_1 = 0$, the wave field is similar for different frequencies. An

	AFEM						FEM	
ω	$\{h_k\}$	T_{up}	m	j_{stop}	err_2	$\overline{n}_{DOF}$	err_2	$\overline{n}_{DOF}$
10π	$\{\frac{1}{5}, \frac{1}{50}\}$	$\frac{1}{10}$	39	22	7.47e-03	1.93e+04	4.97e-03	4.88e+04
20π	$\{\frac{1}{5}, \frac{1}{10}, \frac{1}{100}\}$	$\frac{1}{20}$	39	39	8.71e-03	4.15e+04	8.43e-03	1.77e+05
40π	$\{\frac{1}{5}, \frac{1}{20}, \frac{1}{200}\}$	$\frac{1}{40}$	39	73	1.77e-02	9.33e+04	1.74e-02	6.74e+05
80π	$\{\frac{1}{5}, \frac{1}{40}, \frac{1}{400}\}$	$\frac{1}{80}$	39	139	3.48e-02	2.09e+05	3.44e-02	2.63e+06

TABLE 3. Estimated $L^2(\Omega_0)$ error and average number of degrees of freedom for the biquadratic adapted (AFEM) and classical (FEM) finite element approximation to the 2D plane wave Helmholtz problem for different angular frequencies ω. To estimate the error, we take the numerical approximation on a uniform mesh of width $h_K/2$ as the exact solution.

	1D			2D		
ω	$\overline{n}_{DOF}$	ratio	rate	$\overline{n}_{DOF}$	ratio	rate
10π	1.14e+02			1.93e+04		
20π	1.43e+02	1.25	0.33	4.15e+04	2.15	1.10
40π	1.78e+02	1.24	0.31	9.33e+04	2.25	1.17
80π	2.12e+02	1.19	0.25	2.09e+05	2.24	1.16

TABLE 4. Estimated growth rate of the average number of degrees of freedom $\overline{n}_{DOF}$ with respect to the frequency ω for the adaptive finite element method. The results correspond to the incoming plane wave problem in 1D and 2D.

illustration of the total time-harmonic field and the error for the case $\omega = 40\pi$ is given in Figure 7.

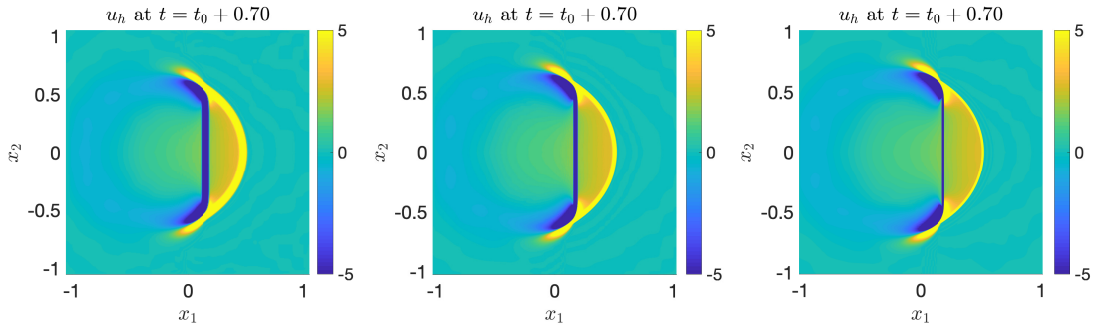

FIGURE 6. Snapshot of u_h at time $t = t_0 + 0.70$ for the 2D plane wave test case with $\omega = 20\pi$ (left), $\omega = 40\pi$ (middle), and $\omega = 80\pi$ (right).

To illustrate the adaptive mesh refinement procedure, we define, for each set of parent elements $\mathcal{P}$, the function $\text{LEVEL}(\mathcal{P}) : \Omega \to \mathbb{R}$ as

$$\text{LEVEL}(\mathcal{P})(\mathbf{x}) := \max(\{0\} \cup \{k \mid \mathbf{x} \in E \text{ for some } E \in \mathcal{P} \cap \mathcal{T}^k\}).$$

In other words, $\text{LEVEL}(\mathcal{P})(\mathbf{x})$ returns the level of the finest element in $\mathcal{P}$ that contains $\mathbf{x}$. If no element in $\mathcal{P}$ contains $\mathbf{x}$, then $\text{LEVEL}(\mathcal{P})(\mathbf{x})$ returns 0. An illustration of the mesh adaptation algorithm is given in Figure 8.

4.4. **Numerical example in 2D: point source.** As a next 2D numerical example, we consider the Helmholtz equation of the form in (2) with a slightly smeared out point source

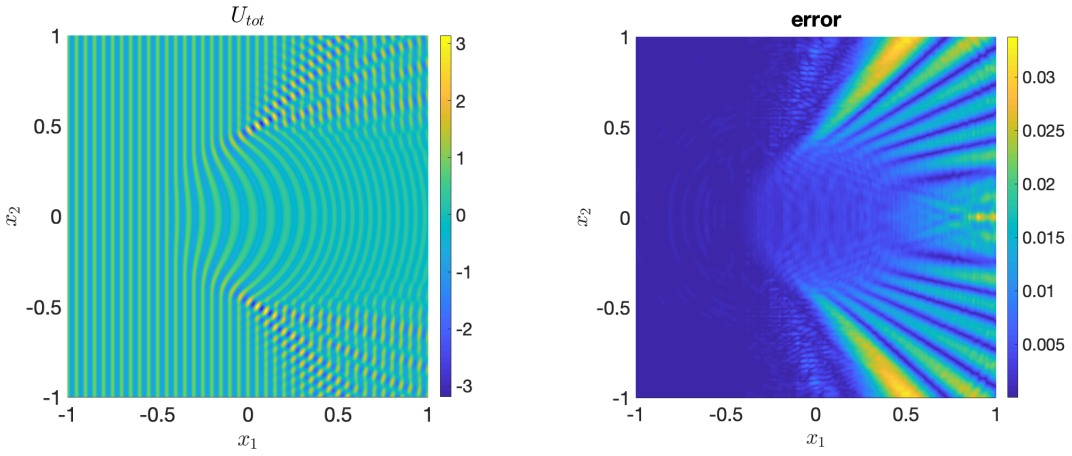

FIGURE 7. Total time-harmonic field $U_{tot} = U_h + U_I$ (left) and error $|U_S - U_h|$ (right) for the 2D plane wave test case with $\omega = 40\pi$.

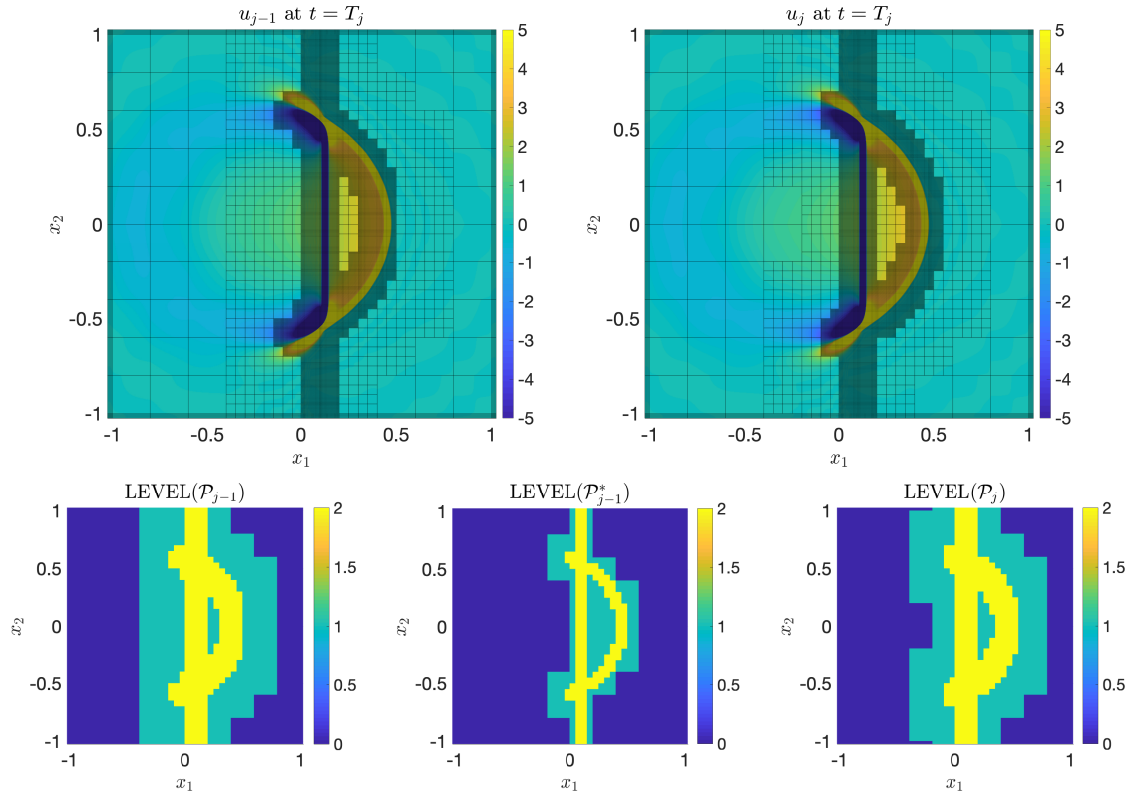

FIGURE 8. Top: illustration of wave field u_{j-1} and the updated wave field u_j at time $T_j = t_0 + 0.65$, for the 2D plane wave test case with $\omega = 40\pi$ using three levels of nested meshes with $\{h_k\} = \{\frac{1}{5}, \frac{1}{20}, \frac{1}{200}\}$. Bottom: parent elements LEVEL($\mathcal{P}_{j-1}$), marked elements LEVEL($\mathcal{P}^*_{j-1}$), and marked neighbour elements LEVEL($\mathcal{P}_j$). Here, $\mathcal{P}_{j-1}$, $\mathcal{P}^*_{j-1}$, and $\mathcal{P}_j$ are defined as in Algorithm 2.

F of the form

$$F(x_1, x_2) := \frac{200}{\lambda^2} f_0\left(\frac{\rho(x_1, x_2)}{\frac{1}{2}\lambda}\right) \qquad f_0(\xi) := \begin{cases} (\xi^2 - 1)^4, & |\xi| \le 1, \\ 0, & \text{otherwise}, \end{cases}$$

where $\lambda = 2\pi/k_0$ denotes the wave length in the exterior domain, $k_0 = \omega/c_0$ denotes the wave number in the exterior domain, $\rho(x_1, x_2) = \sqrt{(x_1 - x_{1,0})^2 + (x_2 - x_{2,0})^2}$ denotes the distance to $(x_{1,0}, x_{2,0})$, and $(x_{1,0}, x_{2,0}) := (0.5, 0.5)$ is the position of the point source. Note that the support of F is centered at $(x_{1,0}, x_{2,0})$ and has a diameter of one wave length λ. The domain, absorbing boundary layer, and spatial parameters α and β are chosen as in the previous example.

For the numerical approximation, we use the same spatial discretisation and parameters η_0 and ϵ_0 as in the previous example. The initial time is $t_0 = -\pi\omega^{-1}$. Similar to the previous examples, we compare the numerical results of the adaptive method with the classical finite element method for different frequencies. The results are presented in Table 5. Again, the accuracy of the adaptive and classical method are comparable, whereas the average number of degrees of freedom is significantly smaller for the adaptive method and grows almost linearly with ω instead of quadratically.

	AFEM						FEM	
ω	$\{h_k\}$	T_{up}	m	j_{stop}	err_2	$\overline{n}_{DOF}$	err_2	$\overline{n}_{DOF}$
10π	$\{\frac{1}{5}, \frac{1}{50}\}$	$\frac{1}{10}$	39	55	1.26e-02	9.79e+03	8.07e-03	4.88e+04
20π	$\{\frac{1}{5}, \frac{1}{10}, \frac{1}{100}\}$	$\frac{1}{20}$	39	65	1.16e-02	2.81e+04	1.01e-02	1.77e+05
40π	$\{\frac{1}{5}, \frac{1}{20}, \frac{1}{200}\}$	$\frac{1}{40}$	39	87	1.55e-02	7.89e+04	1.40e-02	6.74e+05
80π	$\{\frac{1}{5}, \frac{1}{40}, \frac{1}{400}\}$	$\frac{1}{80}$	39	147	2.14e-02	1.82e+05	1.96e-02	2.63e+06

TABLE 5. Estimated $L^2(\Omega_0)$ error and average number of degrees of freedom for the biquadratic adapted (AFEM) and classical (FEM) finite element approximation to the 2D point source Helmholtz problem for different angular frequencies ω. To estimate the error, we take the numerical approximation on a uniform mesh of width $h_K/2$ as the exact solution.

An illustration of the adaptive method is given in Figure 9. The left image of Figure 9 shows a sharp circular wave front generated by the point source and an adapted mesh that is only refined near this wave front. The right image shows the approximated time-harmonic wave field.

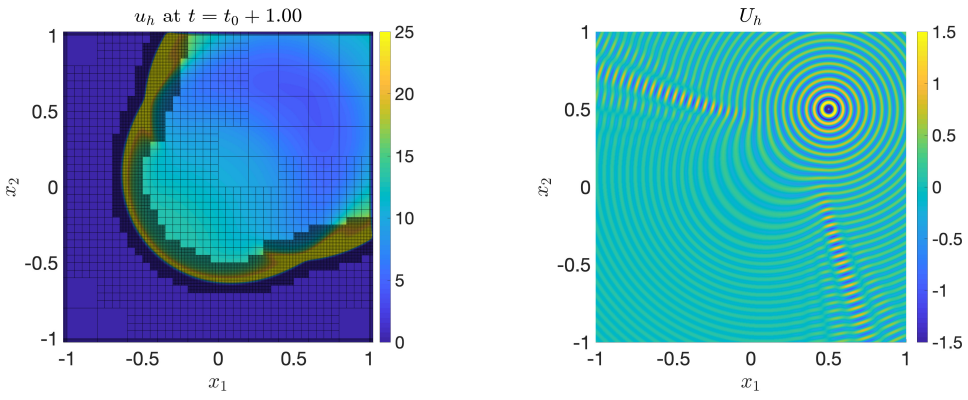

FIGURE 9. Snapshot of the wave field u_h at time $T_j = t_0 + 1$ (left) and time-harmonic field U_h (right) for the 2D point source test case with $\omega = 40\pi$.

4.5. **Numerical example in 2D: trapping mode.** As a final 2D numerical example, we consider a scattering problem with a sound-soft scatterer that can trap waves. The scatterer

is illustrated in Figure 10. The domain, absorbing boundary layer, and incoming wave are chosen as in Section 4.3 and the spatial parameters are given by $\alpha \equiv 1$, $\beta \equiv 1$.

For the numerical test, we only consider the case $\omega = 30\pi$. We use two nested meshes with mesh size $\{h_k\} = \{\frac{1}{10}, \frac{1}{150}\}$ and with biquadratic elements, a mesh update time $T_{up} = 1/30$, a mesh refinement criterion $\xi_0 = \frac{1}{100}\omega$ and a stopping criterion $\epsilon_0 = \frac{5}{100}\omega$. The initial time is $t_0 = 0.4 - \pi\omega^{-1}$, the number of time steps between each mesh update is $m = 20$, and the stopping criterion is triggered at $t_{stop} = T_{j_{stop}}$, with $j_{stop} = 332$. We compare the results of the adaptive finite element method and classical finite element method as in the previous examples. The results are listed in Table 6. An illustration of the adapted mesh and of the computed time-harmonic wave field is also given in Figure 10. Due to trapping, it takes much longer before the wave field vanishes, which makes time-domain approaches less efficient. Furthermore, as illustrated in Figure 10, a large region around the trapping area requires a fine mesh, which makes the adaptive method less efficient. The average number of degrees of freedom is still smaller than for the classical finite element method, since the adaptive method correctly determines in which part of the domain the wave field is active.

$\omega = 30\pi$	err_2	$\overline{n}_{DOF}$
AFEM	1.77e-01	9.95e+04
FEM	1.57e-01	3.31e+05

TABLE 6. Estimated $L^2(\Omega_0)$ error and average number of degrees of freedom for the biquadratic adapted (AFEM) and classical (FEM) finite element approximation to the 2D trapping mode Helmholtz problem for angular frequencies $\omega = 30\pi$. To estimate the error, we take the numerical approximation on a uniform mesh of width $h_K/2 = 1/300$ as the exact solution.

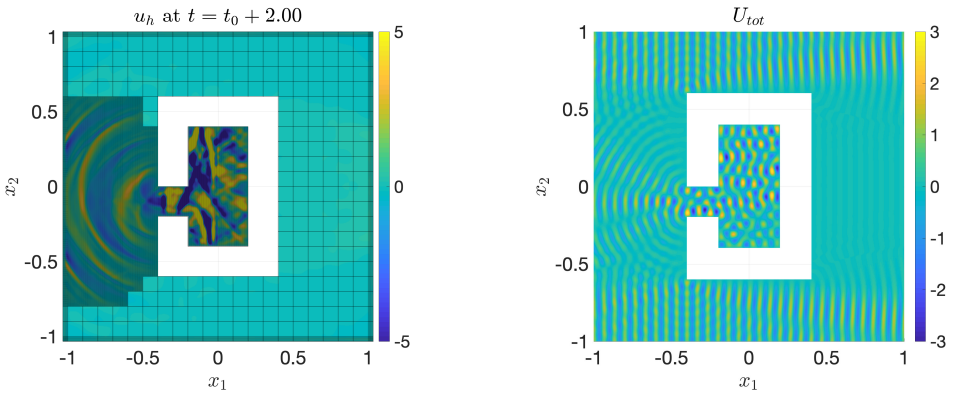

FIGURE 10. Snapshot of the wave field u_h at time $T_j = t_0 + 2$ (left) and total time-harmonic field $U_{tot} = U_h + U_I$ (right) for the 2D trapping mode test case with $\omega = 30\pi$.

5. Conclusion

We considered the time-harmonic acoustic scattering problem with smoothly varying coefficients for an incoming plane wave of angular frequency ω. The proposed method consists of solving the wave equation in the time domain for a single incoming plane wavelet using an adaptive mesh. The time-harmonic solution is then recovered by computing the Fourier transform in time using an adaptive algorithm that exploits the reduced number of degrees of freedom corresponding to the adapted meshes. We compared our adaptive finite element

method to a standard classical finite element time domain method and show that the accuracy is comparable, whereas the average number of degrees of freedom for our adaptive method grows at a significantly smaller rate as the frequency ω increases. In particular, numerical examples indicate that the average number of degrees of freedom for the adaptive finite element method scales almost like $\mathcal{O}(\omega^{d-1})$, with d the number of dimensions in space, instead of $\mathcal{O}(\omega^d)$. Numerical examples also demonstrate that our method can be extended to include external source terms and sound-soft scatterers. The method, however, provides only a limited advantage in the presence of trapping modes.

Appendix A. Limiting amplitude principle

Let $U = U(\mathbf{x})$ be the solution to the Helmholtz problem

$$-\omega^2 U - \beta^{-1}\nabla\cdot(\alpha\nabla U) = F \qquad \text{in } \mathbb{R}^d, \tag{24a}$$

$$[\text{far field radiation condition on } U], \tag{24b}$$

and let $u = u(\mathbf{x}, t)$ be the solution to the wave problem in the time domain

$$\partial_t^2 u - \beta^{-1}\nabla\cdot(\alpha\nabla u) = f \qquad \text{in } \mathbb{R}^d \times (t_0, \infty), \tag{25a}$$

$$[\text{zero initial conditions on } u \text{ at } t = t_0], \tag{25b}$$

with spatial parameters $\alpha = \alpha(\mathbf{x})$ and $\beta = \beta(\mathbf{x})$, source terms $f = f(\mathbf{x}, t)$ and $F = F(\mathbf{x})$, and frequency $\omega > 0$.

We assume that $\alpha(\mathbf{x}) \geq \alpha_{\min}$ and $\beta(\mathbf{x}) \geq \beta_{\min}$ for $\mathbf{x} \in \mathbb{R}^d$, and that $\alpha(\mathbf{x}) \equiv \alpha_0$ and $\beta(\mathbf{x}) \equiv \beta_0$ for $\mathbf{x} \in \mathbb{R}^d \backslash \Omega_{in}$, where Ω_{in} is a bounded domain and $\alpha_{\min}$, $\beta_{\min}$, α_0, and β_0 are positive constants. We also assume that $f(\cdot, t)$ and F are supported within Ω_{in}.

Let $\mathcal{U}$ be a Hilbert space on a bounded domain Ω with $\Omega \supset \Omega_{in}$. The limiting amplitude principle states that, if f is of the form $f(\mathbf{x}, t) = F(\mathbf{x})e^{-i\omega t}$, then $u(\cdot, t)$ converges in $\mathcal{U}$ to $Ue^{-i\omega t}$ as t tends to infinity. In particular, we can define the limiting amplitude principle as follows:

Definition A.1 (limiting amplitude principle)**.** *Let u be the solution to the wave equation given in* (25)*, with $f(\mathbf{x}, t) := F(\mathbf{x})e^{-i\omega t}$ for all $t > T$ for some $T > t_0$, and let U be the solution to the Helmholtz equation given in* (24)*. The limiting amplitude principle states that*

$$\lim_{t\to\infty} \left\| u(\cdot, t) - U(\cdot)\, e^{-i\omega t} \right\|_{\mathcal{U}} = 0. \tag{26}$$

The following is known about the validity of the limiting amplitude principle, with $\operatorname{supp}(F) \subset \Omega \subset \mathbb{R}^d$.

- For $d = 3$, (26) was derived in [43] for $F \in L^2(\Omega)$, $\alpha \in \mathcal{C}^2(\mathbb{R}^3)$, $\beta \in \mathcal{C}^1(\mathbb{R}^3)$, $\mathcal{U} = L^2(\Omega)$.
- For $d \geq 2$, it follows from [15, Ch. 2] that (26) holds true for $F \in L^2(\Omega)$, $\alpha \in \mathcal{C}^2(\mathbb{R}^d)$, $\beta \equiv \beta_0$, $\mathcal{U} = H^1(\Omega)$.
- For $d = 1$, the form (26) of the limiting amplitude principle is not valid [14, Sect. 3]; a modified form is currently under investigation and will be presented in a forthcoming paper.

Whenever the limiting amplitude principle is valid for $\mathcal{U} = L^2(\Omega)$, we have the following result.

Lemma A.2. *Let u be the solution to* (25) *with a source term f that has compact support in space and time. Extend u and f by zero to $\mathbb{R}^d \times (-\infty, t_0)$ and define, for any frequency $\omega > 0$, $\tilde{U}_\omega := \mathfrak{F}_t[u](\cdot, -\omega)$ and $\tilde{F}_\omega := \mathfrak{F}_t[f](\cdot, -\omega)$, where $\mathfrak{F}_t$ denotes the Fourier transform with respect to time. If the limiting amplitude principle is valid for $\mathcal{U} = L^2(\Omega)$, then, for any bounded domain $\Omega \subset \mathbb{R}^d$ with $\operatorname{supp}(f(\cdot, t)) \subset \Omega$, we have that $\tilde{U}_\omega$ is the solution to the Helmholtz equation given in* (24) *with source term $F = \tilde{F}_\omega$.*

Proof. Fix $\omega > 0$ and let U be the solution to (24) with source term $F = \tilde{F}_\omega$. We need to show that $\tilde{U}_\omega = U$. To do so, define $G(t) := H(t)e^{-i\omega t}$, where $H(t)$ denotes the Heaviside step function ($H(t) = 1$ for $t \geq 0$ and $H(t) = 0$ for $t < 0$). Also, let $*_t$ denote the convolution operator with respect to time. If we apply $G *_t$ to (25), we obtain

$$\partial_t^2(G *_t u) - \beta^{-1}\nabla \cdot (\alpha \nabla (G *_t u)) = (G *_t f) \qquad \text{in } \mathbb{R}^d \times (t_0, \infty), \tag{27a}$$

$$[\text{zero initial conditions on } (G *_t u) \text{ at } t = t_0]. \tag{27b}$$

Since f has finite support in time, we have that $f(\cdot, t) = 0$ for all $t > T$ for some $T > t_0$. Therefore, $(G *_t f) = e^{-i\omega t}\tilde{F}_\omega$ for $t > T$. It then follows from the limiting amplitude principle that $(G *_t u)(\cdot, t)$ converges to $e^{-i\omega t}U$ as $t \to \infty$ in $\mathcal{U} = L^2(\Omega)$. In other words, $\lim_{t\to\infty} e^{i\omega t}(G *_t u)(\cdot, t) = U$. By definition of $\tilde{U}_\omega$, we also have that $\lim_{t\to\infty} e^{i\omega t}(G *_t u) = \tilde{U}_\omega$ and hence, $\tilde{U}_\omega = U$. □

References

[1] S. Adjerid. A posteriori finite element error estimation for second-order hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 191(41-42):4699–4719, 2002.

[2] S. Adjerid and M. Baccouch. Asymptotically exact a posteriori error estimates for a one-dimensional linear hyperbolic problem. *Applied Numerical Mathematics*, 60(9):903–914, 2010.

[3] S. Adjerid and T. C. Massey. A posteriori discontinuous finite element error estimation for two-dimensional hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 191(51-52):5877–5897, 2002.

[4] F. Alauzet, P. L. George, B. Mohammadi, P. Frey, and H. Borouchaki. Transient fixed point-based unstructured mesh adaptation. *International Journal for Numerical Methods in Fluids*, 43(6-7):729–745, 2003.

[5] D. Appelo, F. Garcia, and O. Runborg. Waveholtz: Iterative solution of the Helmholtz equation via the wave equation. *SIAM Journal on Scientific Computing*, 42(4):A1950–A1983, 2020.

[6] I. Babuška and S. Sauter. Is the pollution effect of the FEM avoidable for the Helmholtz equation considering high wave numbers? *SIAM Journal on Numerical Analysis*, 34(6):2392–2423, 1997.

[7] D. H. Baffet, M. J. Grote, S. Imperiale, and M. Kachanovska. Energy decay and stability of a perfectly matched layer for the wave equation. *Journal of Scientific Computing*, 81(3):2237–2270, 2019.

[8] W. Bangerth, M. Geiger, and R. Rannacher. Adaptive Galerkin finite element methods for the wave equation. *Computational Methods in Applied Mathematics*, 10(1):3–48, 2010.

[9] W. Bangerth, M. Grote, and C. Hohenegger. Finite element method for time dependent scattering: nonreflecting boundary condition, adaptivity, and energy decay. *Computer Methods in Applied Mechanics and Engineering*, 193(23-26):2453–2482, 2004.

[10] W. Bangerth and R. Rannacher. Finite element approximation of the acoustic wave equation: Error control and mesh adaptation. *East West Journal of Numerical Mathematics*, 7(4):263–282, 1999.

[11] C. Bernardi and E. Süli. Time and space adaptivity for the second-order wave equation. *Mathematical Models and Methods in Applied Sciences*, 15(02):199–225, 2005.

[12] M.-O. Bristeau, R. Glowinski, and J. Périaux. Controllability methods for the computation of time-periodic solutions; application to scattering. *Journal of Computational Physics*, 147(2):265–292, 1998.

[13] G. Cohen. *Higher-order numerical methods for transient wave equations*. Springer, 2002.

[14] D. M. Èidus. On the principle of limiting absorption. *Matematicheskii Sbornik*, 57 (99):13–44, 1962. English translation: American Mathematical Society Translations (2) 47 (1965), 157-191.

[15] D. M. Èidus. The principle of limit amplitude. *Russian Mathematical Surveys*, 24(3):97, 1969.

[16] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: hierarchical matrix representation. *Communications on Pure and Applied Mathematics*, 64(5):697–735, 2011.

[17] B. Engquist and L. Ying. Sweeping preconditioner for the Helmholtz equation: moving perfectly matched layers. *Multiscale Modeling & Simulation*, 9(2):686–710, 2011.

[18] O. G. Ernst and M. J. Gander. Why it is difficult to solve Helmholtz problems with classical iterative methods. *In: I. Graham, T. Hou, O. Lakkis, R. Scheichl (Eds.), Numerical analysis of multiscale problems*, pages 325–363, 2012.

[19] J. Fang, J. Qian, L. Zepeda-Núñez, and H. Zhao. Learning dominant wave directions for plane wave methods for high-frequency Helmholtz equations. *Research in the Mathematical Sciences*, 4(1):1–35, 2017.

[20] M. J. Gander and H. Zhang. A class of iterative solvers for the Helmholtz equation: Factorizations, sweeping preconditioners, source transfer, single layer potentials, polarized traces, and optimized Schwarz methods. *Siam Review*, 61(1):3–76, 2019.

[21] E. H. Georgoulis, O. Lakkis, and C. Makridakis. A posteriori $L^\infty(L^2)$-error bounds for finite element approximations to the wave equation. *IMA Journal of Numerical Analysis*, 33(4):1245–1264, 2013.
[22] E. H. Georgoulis, O. Lakkis, C. G. Makridakis, and J. M. Virtanen. A posteriori error estimates for leap-frog and cosine methods for second order evolution problems. *SIAM Journal on Numerical Analysis*, 54(1):120–136, 2016.
[23] E. Giladi and J. B. Keller. A hybrid numerical asymptotic method for scattering problems. *Journal of Computational Physics*, 174(1):226–247, 2001.
[24] R. Glowinski and T. Rossi. A mixed formulation and exact controllability approach for the computation of the periodic solutions of the scalar wave equation. I. Controllability problem formulation and related iterative solution. *Comptes Rendus Mathématique. Académie des Sciences. Paris*, 343(7):493–498, 2006.
[25] O. Gorynina, A. Lozinski, and M. Picasso. An easily computable error estimator in space and time for the wave equation. *ESAIM. Mathematical Modelling and Numerical Analysis*, 53(3):729–747, 2019.
[26] O. Gorynina, A. Lozinski, and M. Picasso. Time and space adaptivity of the wave equation discretized in time by a second-order scheme. *IMA Journal of Numerical Analysis*, 39(4):1672–1705, 2019.
[27] M. J. Grote, F. Nataf, J. H. Tang, and P.-H. Tournier. Parallel controllability methods for the Helmholtz equation. *Computer Methods in Applied Mechanics and Engineering*, 362:112846, 2020.
[28] M. J. Grote and I. Sim. Efficient PML for the wave equation. *arXiv:1001.0319*, 2010.
[29] M. J. Grote and J. H. Tang. On controllability methods for the Helmholtz equation. *Journal of Computational and Applied Mathematics*, 358:306–326, 2019.
[30] E. Heikkola, S. Mönkölä, A. Pennanen, and T. Rossi. Controllability method for the Helmholtz equation with higher-order discretizations. *Journal of Computational Physics*, 225(2):1553–1576, 2007.
[31] C. Johnson. Discontinuous Galerkin finite element methods for second order hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 107(1-2):117–129, 1993.
[32] B. Kaltenbacher, M. Kaltenbacher, and I. Sim. A modified and stable version of a perfectly matched layer technique for the 3-d second order wave equation in time domain with an application to aeroacoustics. *Journal of Computational Physics*, 235:407–422, 2013.
[33] O. Ladyzhenskaya. On the principle of limit amplitude. *Uspekhi Matematicheskikh Nauk*, 12(3):161–164, 1957.
[34] X. Li and N.-E. Wiberg. Implementation and adaptivity of a space-time finite element method for structural dynamics. *Computer Methods in Applied Mechanics and Engineering*, 156(1-4):211–229, 1998.
[35] C. S. Morawetz. The limiting amplitude principle. *Communications on Pure and Applied Mathematics*, 15(3):349–361, 1962.
[36] N. C. Nguyen, J. Peraire, F. Reitich, and B. Cockburn. A phase-based hybridizable discontinuous Galerkin method for the numerical solution of the Helmholtz equation. *Journal of Computational Physics*, 290:318–335, 2015.
[37] R. Nochetto, M. Paolini, and C. Verdi. An adaptive finite element method for two-phase Stefan problems in two space dimensions. I. Stability and error estimates. *Mathematics of Computation*, 57(195):73–108, 1991.
[38] F. M. Odeh. Principles of limiting absorption and limiting amplitude in scattering theory. II. The wave equation in an inhomogeneous medium. *Journal of Mathematical Physics*, 2(6):800–802, 1961.
[39] M. Picasso. Numerical study of an anisotropic error estimator in the $L^2(H^1)$ norm for the finite element discretization of the wave equation. *SIAM Journal on Scientific Computing*, 32(4):2213–2234, 2010.
[40] C. C. Stolk. An improved sweeping domain decomposition preconditioner for the Helmholtz equation. *Advances in Computational Mathematics*, 43(1):45–76, 2017.
[41] C. C. Stolk. A time-domain preconditioner for the Helmholtz equation. *arXiv preprint arXiv:2006.16861*, 2020.
[42] A. Taflove and S. C. Hagness. *Computational electrodynamics: the finite-difference time-domain method.* Artech house, 2005.
[43] H. Tamura. Resolvent estimates at low frequencies and limiting amplitude principle for acoustic propagators. *Journal of the Mathematical Society of Japan*, 41(4):549–575, 1989.
[44] L. L. Thompson and D. He. Adaptive space–time finite element methods for the wave equation on unbounded domains. *Computer Methods in Applied Mechanics and Engineering*, 194(18-20):1947–2000, 2005.
[45] B. Vainberg. Principles of radiation, limit absorption and limit amplitude in the general theory of partial differential equations. *Russian Mathematical Surveys*, 21(3):115–193, 1966.
[46] K. Yee. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Transactions on Antennas and Propagation*, 14(3):302–307, 1966.