

# FINITE ELEMENT APPROXIMATION OF WAVE PROBLEMS WITH CORRECTING TERMS BASED ON TRAINING ARTIFICIAL NEURAL NETWORKS WITH FINE SOLUTIONS

ARNAU FABRA<sup>§,‡</sup>, JOAN BAIGES<sup>§</sup> AND RAMON CODINA<sup>§,‡</sup>

ABSTRACT. *In this paper we present a general idea to correct coarse models by introducing a correcting term designed from fine solutions. We apply this idea to the wave equation in both the time domain and the frequency domain. This correcting term is computed and trained by making use of learning algorithms, such as the least squares model or a model constructed from an artificial neural network. The performance of the method is tested through four different acoustic numerical examples, where the fine solutions are characterized for having a finer discretization either in time or in space.*

**Keywords:** Wave equation, stabilized finite element methods, least squares training, artificial neural networks

## 1. INTRODUCTION

Numerical simulations have become one of the most important tools in a considerable amount of scientific fields. In spite of this, the computational cost is still the main limitation, as the growing of computational power leads to deal with more complex problems and to target more accurate simulations. In order to overcome this obstacle, one of the proposed solutions is the use of the so called Reduced Order Models (ROM) (see, e.g. [1]), whose main objective is to reproduce the features and accuracy of Full Order Models (FOM), with models that require less degrees of freedom and therefore, less computational cost. Obviously, *classical* ROM is not the only method that makes use of the idea of reducing the number of degrees of the problem (see for example [2, 3, 4, 5]). The simplest way to achieve this reduction is by *coarsening* the numerical approximation, either in space, in time or both, but having at our disposal samples of FOM solutions. This is what we explore in this paper. In this sense, this work is an extension of [6], where a ROM based on adaptive finite element (FE) meshes and correction terms combined with Artificial Neural Networks (ANNs) is presented. The aim of this paper is to extend how these models can be used in general and to apply them in wave problems; so firstly, a little review of the wave problems is presented, secondly, how this specific method to reduce the number of degrees of freedom works is explained, and finally, the contributions of this paper are exposed.

Wave problems can be analyzed and solved in two separated frameworks, the time domain and the frequency domain, the change from one to the other being provided by the Fourier transform. Furthermore, the equations of each domain can be formulated in two different forms: the irreducible form, where just one variable needs to be solved, and the mixed form, where two or more variables have to be computed. In order to change from one form to the other, some analytical manipulation is needed. The variables of the equations depend on the problem being considered; for instance, in acoustics the irreducible form would only involve the acoustic pressure, whereas the mixed form would require solving

---

*Date:* November 23, 2021.

<sup>§</sup> Universitat Politècnica de Catalunya, Barcelona Tech, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain.

<sup>‡</sup> Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Edifici C1, Campus Nord UPC, Gran Capità S/N, 08034 Barcelona, Spain.

E-mails: [afabra@cimne.upc.edu](mailto:afabra@cimne.upc.edu) (AF), [joan.baiges@upc.edu](mailto:joan.baiges@upc.edu) (JB), [ramon.codina@upc.edu](mailto:ramon.codina@upc.edu) (RC).

for both the acoustic pressure and the acoustic velocity. For conciseness, we will refer to this case, although any other wave problem could be considered.

The irreducible form of the wave equation in the time domain is perhaps the most classical form of the wave equation. This is a very well-known second order linear partial differential equation which describes the temporal evolution of the velocity or the pressure of waves. The irreducible wave equation in the frequency domain is also an equation with vast literature, also known as the Helmholtz equation; it describes the velocity or the pressure of the wave for a certain frequency [7]. The mixed form of the wave equation in the time domain is less common; it describes the temporal evolution of the pressure and the velocity of waves at the same time [8, 9]. One of its advantages compared to irreducible forms is that it can provide a better approximation of the acoustic velocity and can be applied to time dependent domains using an Arbitrary Lagrangian Eulerian (ALE) approach [10]. The last case is the least studied one, it is the mixed form of the wave equation in the frequency domain; it describes both the velocity and the pressure of the wave for a certain frequency [11, 12]. This approach has the advantages of the mixed form of the wave equation and the computational savings associated to solving for a single frequency.

One of the most popular methods to work in the field of acoustics is the boundary element method (see, e.g., [13, 14, 15]). However, in some situations, like small domains or aero-acoustic problems, the FE method is also a good option for the spatial discretization of the problem (see [16]). Since we are interested in working with small domains, the FE method is used in this paper. While for the irreducible form in the time domain the standard Galerkin method can be used for the spatial discretization, all other forms of the wave equation require some sort of stabilization. The Helmholtz equation needs to be stabilized when high frequencies are considered, leading to pollution errors (see [7] and references therein), whereas the mixed forms require stabilization if one wants to use equal interpolation for the acoustic pressure and the acoustic velocity (see [8, 12]). The FE formulation we employ belongs to the computational framework of Variational Multi-Scale (VMS) methods [17, 18], a group of techniques based on splitting the unknowns of the problem into two different scales, where one of those scales can be approximated by the FE mesh and the other one represents the sub-grid scale, which cannot be captured by the FE space. Thanks to these techniques, the instability problems of the Galerkin method can be solved.

To compute accurate solutions, the number of degrees of freedom of the problem can be huge, which causes an enormous computational cost and, many times, results in the need for using supercomputing facilities. This is where the aforementioned ROMs come in and, in particular, the mesh coarsening approach with correcting terms presented in [6], which departs from the work presented in [19]. The idea presented there consists of a training model based on two equal cases that have two different meshes; the case of the coarse mesh plays the role of the ROM and case of the fine mesh plays the role of the FOM. This second one is computationally expensive but necessary, since it provides the information that we need in order to create the correcting model. The required data of the fine mesh is sent to the coarse mesh and the results are compared and introduced to a training algorithm. This algorithm receives the results of the FOM and projects them to the coarse mesh. With this information, it is able to create a correction term that will be added to the coarse approximation in future simulations, thus yielding results closer to the ones that would be obtained with the fine model.

The main objective of this work is to construct and make use of this kind of training models in wave problems, and not only applying these correction models in the space discretization, but also adapting and using them for the time discretization. It is important to note that even though we use these models for this specific purpose, they are actually a general method that allows one to transfer information from fine models (FOMs) to

coarse models (e.g., classical ROMs), so they could be used in many other applications. In this sense, our approach can be considered a *hybrid* one, combining an underlying numerical approximation (a FE approximation, in our case) with a correction term designed by a training model. The most common hybrid models combine physical models with corrections, but the same idea applies by replacing physical models by numerical models.

Regarding the correction model, we will explore two options. On the one hand, a classic Least Squares (LS) model will be used (see [20]); it turns out to work well in simple models but it has limitations in complex scenarios. On the other hand, ANNs will be used, its flexibility and capacity to adapt to almost every model being its main feature. The implementation used for this learning algorithm is the one provided by the open source library FANN (see [21]). The use of ANNs to train models that correct solutions of wave problems is one of the main novelties of this work. However, it is important to note that ANNs have been applied to FE problems in a lot of different ways and approaches (see [22, 23]), although not as a correction tool as we propose here. ANNs have been also used to perform the numerical time integration itself [24, 25], as a way to design ROMs (see [26]) or as a way to approximate the Helmholtz equation with an unsupervised learning technique (see [27]).

This paper is organized as follows. In Section 2, the equations of the wave problem, their variational forms and their discretization in time and space are described. In Section 3, the general concept of the correcting term is presented. In Section 4, the concept that has been explained in general in the previous section is applied to the wave equation. In Section 5, four acoustic numerical examples are shown. Finally, in Section 6 conclusions are drawn.

## 2. PROBLEM STATEMENT AND FINITE ELEMENT APPROXIMATION

**2.1. The continuous wave equation.** Wave phenomena can be described by the wave equation written in different forms. To fix ideas, in this paper we will use the terminology of acoustics, so that the unknowns to compute are the acoustic pressure ( $p$ ) and the acoustic velocity ( $\mathbf{u}$ ) in the case of mixed formulations, and just the acoustic pressure in the case of irreducible formulations.

The problems we consider are initial and boundary value problems posed in a spatial domain  $\Omega \subset \mathbb{R}^d$ , where  $d = 1, 2, 3$  is the spatial dimension. Let  $\partial\Omega$  be the boundary of the domain  $\Omega$ . We split this boundary into two disjoint sets denoted as  $\Gamma_p$  and  $\Gamma_u$ , where the boundary conditions corresponding to the unknowns  $p$  and  $\mathbf{n} \cdot \mathbf{u}$  will be enforced, respectively,  $\mathbf{n}$  being the unit normal to  $\partial\Omega$ . The time domain is the interval  $(0, T)$ . Initial conditions need to be provided at time  $t = 0$ .

Even though we will only present numerical examples corresponding to irreducible formulations, the ideas to be proposed can be also applied to mixed formulations. Let us start writing the mixed formulation in the time domain, which consists of finding the acoustic pressure  $p : \Omega \times (0, T) \rightarrow \mathbb{R}$  and the acoustic velocity  $\mathbf{u} : \Omega \times (0, T) \rightarrow \mathbb{R}^d$  such that

$$\begin{cases} \mu_p \partial_t p + \nabla \cdot \mathbf{u} = f_p \\ \mu_u \partial_t \mathbf{u} + \nabla p = \mathbf{f}_u \end{cases}, \quad (1)$$

with the initial conditions

$$p(\mathbf{x}, 0) = p_0(\mathbf{x}), \quad \mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad (2)$$

$p_0(\mathbf{x})$  and  $\mathbf{u}_0(\mathbf{x})$  being given functions, and with the boundary conditions

$$p = 0 \text{ on } \Gamma_p, \quad \mathbf{n} \cdot \mathbf{u} = 0 \text{ on } \Gamma_u, \quad t \in (0, T). \quad (3)$$

For simplicity, the boundary conditions have been considered homogeneous. Non-reflecting boundary conditions (NRBCs) could also be easily incorporated, and in fact we will use

them in the numerical examples. The forcing terms  $f_p$  and  $f_u$  are assumed to be given and  $\mu_p > 0$ ,  $\mu_u > 0$  are physical properties.

The mixed problem in the frequency domain consists of finding  $p : \Omega \rightarrow \mathbb{C}$  and  $\mathbf{u} : \Omega \rightarrow \mathbb{C}^d$  such that

$$\begin{cases} -i\mu_p\omega p + \nabla \cdot \mathbf{u} = f_p \\ -i\mu_u\omega \mathbf{u} + \nabla p = \mathbf{f}_u \end{cases}, \quad (4)$$

with the same boundary conditions as in Eq. (3). In fact, the unknowns here are the amplitudes associated to the given frequency  $\omega$  of the unknowns in Eqs. (1), although we have used the same symbol in both problems, and likewise for the forcing terms.

Let  $c = (\mu_u\mu_p)^{-1/2}$  be the speed of the wave. The irreducible wave equation in the time domain is obtained by taking the time derivative of the first equation in Eqs. (1) and making use of the second to eliminate the acoustic velocity as unknown. Still calling  $f_p$  the resulting right-hand-side (RHS), the equation found consists of finding  $p : \Omega \times (0, T) \rightarrow \mathbb{R}$  such that

$$\frac{1}{c^2} \partial_{tt} p - \Delta p = f_p, \quad (5)$$

with the initial conditions

$$p(\mathbf{x}, 0) = p_0(\mathbf{x}), \quad \partial_t p|_{t=0} = \dot{p}_0(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad (6)$$

where  $p_0(\mathbf{x})$  and  $\dot{p}_0(\mathbf{x})$  are given functions, and boundary conditions

$$p = 0 \text{ on } \Gamma_p, \quad \mathbf{n} \cdot \nabla p =: \frac{\partial p}{\partial \mathbf{n}} = 0 \text{ on } \Gamma_u. \quad (7)$$

Let  $\omega$  be a given frequency and  $k = \frac{\omega}{c}$  the wave number associated to this frequency and the wave speed  $c$ . The irreducible form of the wave equation in the frequency domain consists of finding  $p : \Omega \rightarrow \mathbb{C}$  solution of the Helmholtz equation

$$-\Delta p - k^2 p = f_p, \quad (8)$$

with the same boundary conditions as in Eq. (7). The same comments regarding the notation as in the mixed form in the frequency domain apply now. This equation can be found considering that  $p$  is the amplitude associated to the frequency  $\omega$  of the solution of Eq. (5) or directly multiplying by  $i\omega$  the first equation in Eqs. (4) and making use of the second to eliminate the acoustic velocity as an unknown of the problem.

To write the weak form of the problem, we need to introduce some notation. Let  $w$  be a subdomain of  $\Omega$ .  $L^2(w)$  denotes the space of square integrable functions in  $w$ , with inner product  $(\cdot, \cdot)_w$ ,  $L^2(w)^d$  is the space of vector valued functions with components in  $L^2(w)$ ,  $H^1(w)$  is the space of functions in  $L^2(w)$  with derivatives in  $L^2(w)$  and  $H(\text{div}, w)$  is the space of vector functions with components and divergence in  $L^2(w)$ . We will use the simplification  $(a, b)_\Omega \equiv (a, b)$ .

Let  $V_p := \{q \in H^1(\Omega) \mid q = 0 \text{ on } \Gamma_p\}$  and let us define the bilinear form  $B(p, q) := (\nabla p, \nabla q)$  and the linear form  $L(q) := (f_p, q)$  (the meaning of  $f_p$  depending on the problem). The weak form of the irreducible wave equation in the time domain consists of finding  $p : (0, T) \rightarrow V_p$  such that

$$\begin{aligned} \frac{1}{c^2} (\partial_{tt} p, q) + B(p, q) &= L(q), \quad \forall q \in V_p, \\ (p(0), q) &= (p_0, q), \quad (\partial_t p(0), q) = (\dot{p}_0, q), \quad \forall q \in L^2(\Omega), \end{aligned}$$

whereas the weak form in the frequency domain consists of finding  $p \in V_p$  such that

$$-k^2 (p, q) + B(p, q) = L(q), \quad \forall q \in V_p.$$

Note that in this case  $p$  may be complex valued, whereas for the problem in the time domain it is real valued.

For the irreducible problems (either in the time or in the frequency domains)  $p$  needs to be in  $H^1(\Omega)$  and the associated acoustic velocity (which could be computed after knowing  $p$ ) belongs to  $L^2(\Omega)^d$ . The weak form of the mixed formulation can be written considering two different functional settings, one which is the same as for the irreducible form and another one in which  $p$  belongs to  $L^2(\Omega)$  in space and  $\mathbf{u}$  to  $H(\text{div}, \Omega)$ . However, since we will present numerical results for the irreducible formulation only, we shall omit details of the mixed formulation, for the sake of conciseness. See [8, 9, 11, 12] for details.

**2.2. Finite element approximation in space.** Let us assume for simplicity that  $\Omega$  is a polyhedral domain and let  $\mathcal{T}_h = \{K\}$  be a FE partition of size  $h$ . The collection of interior edges will be denoted by  $\{E\}$ . The subscript  $h$  will be employed to refer to FE functions and FE spaces. The  $L^2$  inner product in  $K$  (resp.  $E$ ) will be represented as  $(\cdot, \cdot)_K$  (resp.  $E$ ). From  $\mathcal{T}_h$  we may construct conforming FE spaces  $V_{p,h} \subset V_p$  to approximate the pressure. In the case of mixed formulations, special care is needed to construct the velocity FE space to yield a stable numerical formulation; the approach we favor is to use arbitrary interpolations for velocity and pressure and modify the standard Galerkin method using a stabilized FE formulation, in our case based on the Variational Multi-Scale (VMS) approach [17, 18, 8, 12, 28].

For the irreducible formulation in the time domain, the standard Galerkin method is stable. The semi-discrete problem (discretized in space, continuous in time) reads: find  $p_h : (0, T) \rightarrow V_{p,h}$  such that

$$\frac{1}{c^2}(\partial_{tt}p_h, q_h) + B(p_h, q_h) = L(q_h), \quad \forall q_h \in V_{p,h}, \quad (9)$$

$$(p_h(0), q_h) = (p_0, q_h), \quad (\partial_t p_h(0), q_h) = (\dot{p}_0, q_h), \quad \forall q_h \in \tilde{V}_{p,h}, \quad (10)$$

where  $\tilde{V}_{p,h}$  is constructed as  $V_{p,h}$  but without the imposition of boundary conditions.

Let  $\mathbf{P}(t)$  be the array of nodal unknowns of the pressure  $p_h$ . Denoting with a dot the temporal derivatives, the matrix version of Eq. (9) is

$$\frac{1}{c^2} \mathbf{M} \ddot{\mathbf{P}} + \mathbf{K} \mathbf{P} = \mathbf{F}_0,$$

where  $\mathbf{M}$  is the standard mass matrix,  $\mathbf{K}$  the matrix coming from the Laplacian and  $\mathbf{F}_0$  the resulting force vector. The initial conditions for this problem are obtained from Eq. (10).

The Galerkin formulation for the Helmholtz equation reads as follows: find  $p_h \in V_{p,h}$  such that

$$-k^2(p_h, q_h) + B(p_h, q_h) = L(q_h), \quad \forall q_h \in V_{p,h}. \quad (11)$$

It is well known that for high values of the wave number  $k$  this problem suffers from pollution errors. The general approach we propose to stabilize the problem and avoid the pollution effect is based on the VMS framework. Details can be found in [7], here we will just state the formulation. It consists of finding  $p_h \in V_{p,h}$  such that

$$\begin{aligned} & -k^2(p_h, q_h) + B(p_h, q_h) + \sum_K \tau(-k^2 q_h - \Delta q_h, \tilde{P}(-k^2 p_h - \Delta p_h - f_p))_K \\ & + \sum_E \delta \left( \left[ \left[ \frac{\partial q_h}{\partial n} \right] \right]_E, \left[ \left[ \frac{\partial p_h}{\partial n} \right] \right]_E \right) = L(q_h), \quad \forall q_h \in V_{p,h}, \end{aligned}$$

where  $[\![ \cdot ]\!]$  stands for the jump over the edges and  $\tau$  and  $\delta$  are parameters of the formulation (see [7]).  $\tilde{P}$  is a projection which can be taken as the identity (case in which we may set  $\delta = 0$ ) or the projection orthogonal to the FE space. In this last case,  $\tilde{P}(-k^2 p_h - \Delta p_h - f_p) = \tilde{P}(-\Delta p_h - f_p) = 0$  for linear elements and  $f_p$  a FE function; it is thus clear that the stabilizing terms evaluated on the element boundaries are crucial.

Calling

$$\begin{aligned} B_s(p_h, q_h) &:= B(p_h, q_h) + \sum_K \tau(-k^2 q_h - \Delta q_h, \tilde{P}(-k^2 p_h - \Delta p_h))_K \\ &\quad + \sum_E \delta \left( \left[ \frac{\partial q_h}{\partial n} \right], \left[ \frac{\partial p_h}{\partial n} \right] \right)_E \\ L_s(q_h) &:= L(q_h) + \sum_K \tau(-k^2 q_h - \Delta q_h, \tilde{P}(f_p))_K, \end{aligned}$$

the stabilized version of Eq. (11) consists of finding  $p_h \in V_{p,h}$  such that

$$-k^2(p_h, q_h) + B_s(p_h, q_h) = L_s(q_h), \quad \forall q_h \in V_{p,h}. \quad (12)$$

The matrix structure of this problem is

$$-k^2 \mathbf{M} \mathbf{P} + \mathbf{K}_s \mathbf{P} = \mathbf{F}_s, \quad (13)$$

where  $\mathbf{K}_s$  is the matrix arising from the bilinear form  $B_s(p_h, q_h)$  and  $\mathbf{F}_s$  the forcing array arising from  $L_s(q_h)$ .

**2.3. Time integration and fully discrete problem.** In the case of the wave equation in the time domain, we need to discretize the problem in time. Our discussion is independent of the time integration scheme employed, but in the numerical examples we will use backward differences (BDF) of second order (BDF2).

Let  $0 = t^0 < \dots < t^n < \dots < t^{n_f} = T$  be a partition of the time interval  $(0, T)$ , for simplicity uniform and of size  $\delta t$ . Let  $f(t)$  be a generic time dependent function and let  $f^n$  be an approximation to  $f(t^n)$ . The BDF2 scheme consists of approximating

$$\left. \frac{d^2 f}{dt^2} \right|_{t=t^n} \approx \frac{2f^n - 5f^{n-1} + 4f^{n-2} - f^{n-3}}{\delta t^2}. \quad (14)$$

Thus, the fully discrete version of Eq. (9) is

$$\frac{1}{c^2 \delta t^2} (2p_h^n - 5p_h^{n-1} + 4p_h^{n-2} - p_h^{n-3}, q_h) + B(p_h^n, q_h) = L(q_h), \quad \forall q_h \in V_{p,h}.$$

For  $n = 1$ ,  $p_h^0$  and  $p_h^{-1}$  can be determined from the initial conditions in Eq. (10), but since  $p_h^{-2}$  is unknown the time integration algorithm has to be initialized with a first order BDF scheme for the second derivative, as usual.

The algebraic version of the problem can be written as

$$\frac{2}{c^2 \delta t^2} \mathbf{M} \mathbf{P}^n + \mathbf{K} \mathbf{P}^n = \mathbf{F}_0 + \frac{5}{c^2 \delta t^2} \mathbf{M} \mathbf{P}^{n-1} - \frac{4}{c^2 \delta t^2} \mathbf{M} \mathbf{P}^{n-2} + \frac{1}{c^2 \delta t^2} \mathbf{M} \mathbf{P}^{n-3}, \quad (15)$$

for  $n = 2, \dots, n_f$ .

### 3. CORRECTION TERMS BASED ON TRAINING: THE GENERAL CONCEPT

In the previous section we have presented the numerical approximation we propose to solve the wave equation. In this section we present a general concept to correct fully discrete problems based on the knowledge of *fine* solutions, i.e., solutions corresponding to finer discretizations. It is in fact irrelevant the reason because we have at our disposal finer solutions. It can be due to a finer time discretization, to a finer FE mesh approximation or to an interpolation of higher degree.

**3.1. Coarse and fine problems.** Suppose that we have a *coarse* and *linear* algebraic problem of the form:

$$AU_c = F, \quad (16)$$

with  $U_c, F$  arrays of  $m$  components, and  $A$  an  $m \times m$  matrix, i.e.,  $U_c, F \in \mathbb{R}^m$ ,  $A \in \mathbb{R}^{m \times m}$ . Given  $F$  and matrix  $A$ , the unknown  $U_c$  could be computed by solving (16), but we intend to modify this system to enhance the accuracy of  $U_c$ . The example of interest in this work is that in which problem (16) is obtained from a coarse discretization of the wave equation, but the idea to be presented is general. In wave problems we would have:

$$A = -k^2 M + K_s, \quad F = F_s, \quad U_c = P,$$

in the case of the Helmholtz equation in Eq. (13) and

$$A = \frac{2}{c^2 \delta t^2} M + K, \quad F = F_0 + \frac{5}{c^2 \delta t^2} M P^{n-1} - \frac{4}{c^2 \delta t^2} M P^{n-2} + \frac{1}{c^2 \delta t^2} M P^{n-3}, \quad U_c = P^n,$$

in the case of the discrete wave equation in the time domain in Eq. (15).

Suppose that, by any means, we know a *finer* approximation to the same continuous problem, most likely obtained from a discrete problem like (16) but with dimension  $M \geq m$ . Different possibilities to obtain this finer solution for wave propagation problems will be discussed in the following section. Let  $U_f \in \mathbb{R}^M$  be this finer solution (the system which it solves is in fact irrelevant), and suppose that we can construct a projection

$$P_{fc} : \mathbb{R}^M \longrightarrow \mathbb{R}^m,$$

i.e., a projection from the space where the fine solution belongs to the space where the coarse solution is defined. Again, the way to construct this projection is irrelevant, and will be discussed in the following section for the particular case of the wave equation.

The correction we will propose is based on the following fundamental assumption:

**Assumption 1.** *The best coarse solution is  $U_c = P_{fc}(U_f)$ .*

In the case of a single linear problem, it is trivial to introduce a *correction* to (16), say  $D_{ex} \in \mathbb{R}^m$ , so that the solution is  $P_{fc}(U_f)$ . Indeed, the solution to

$$AU_c + D_{ex} = F \quad \text{with} \quad D_{ex} = F - AP_{fc}(U_f), \quad (17)$$

is precisely  $U_c = P_{fc}(U_f)$ , as it is trivially checked ( $A$  is assumed to be invertible, obviously).

For a single solve, such a correction is unnecessary, as we already have  $U_f$ . However, system (16) may be parameter dependent, and the fine solution may be known for some of the parameters, but not for all of them.

**3.2. Configurations.** Given a partial differential equation to be solved, the structure of the (coarse) discretization can be written as problem (16), but the particular expressions of matrix  $A$  and the RHS vector  $F$  depend on the data of the problem, including parameters. We will call *configuration* a particular realization of these data and parameters. Therefore, different configurations may correspond to changes in:

- Boundary conditions, which would affect both matrix  $A$  and the array  $F$ . Even though changes in the type of boundary conditions could change the dimension  $m$  of the problem, we will consider it fixed.
- Forcing terms, which would affect the array  $F$ .
- Physical properties, which again would affect both matrix  $A$  and the array  $F$ .
- Time instants, which would correspond to taking time  $t$  as a parameter of the problem. In a time evolution problem as the wave equation, problem (16) has to be understood as the problem to be solved at a certain time step, and thus different time steps can be considered different configurations.

Let us assume that we have  $N$  configurations for which we know the fine solution,  $\mathbf{U}_{f,\alpha}$ ,  $\alpha = 1, \dots, N$ . We will use a subscript in matrix  $\mathbf{A}$ , array  $\mathbf{F}$  and the unknown of the coarse problem,  $\mathbf{U}_c$ , to refer to each particular configuration. Thus, the *uncorrected* coarse problem for each configuration is

$$\mathbf{A}_\alpha \mathbf{U}_{c,\alpha} = \mathbf{F}_\alpha, \quad \alpha = 1, \dots, N.$$

Since these are the configurations in which we know the fine solution,  $\mathbf{U}_{f,\alpha}$ , we can introduce an *exact* correction  $\mathbf{D}_{\text{ex},\alpha}$ , given by

$$\mathbf{D}_{\text{ex},\alpha} = \mathbf{F}_\alpha - \mathbf{A}_\alpha \mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha}), \quad \alpha = 1, \dots, N, \quad (18)$$

so that the solution to the *corrected* coarse problem

$$\mathbf{A}_\alpha \mathbf{U}_{c,\alpha} + \mathbf{D}_{\text{ex},\alpha} = \mathbf{F}_\alpha, \quad (19)$$

is  $\mathbf{U}_{c,\alpha} = \mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha})$ , considered optimal, for  $\alpha = 1, \dots, N$ .

**3.3. The correcting term and main modelling assumption.** As in the case of a single solve in which we know the fine solution, the corrected coarse problems (19) are unnecessary, since we know the fine solution  $\mathbf{U}_{f,\alpha}$ ,  $\alpha = 1, \dots, N$ . We will call the set of these fine solutions *the training set*, and the associated configurations the *training configurations*.

The objective now is to introduce a correction  $\mathbf{D} \in \mathbb{R}^m$  to the coarse problem (16) for *any* configuration, not only for the training configurations, so that the solution to

$$\mathbf{A}\mathbf{U}_c + \mathbf{D} = \mathbf{F},$$

has improved accuracy with respect to that of problem (16). We now have to decide the structure of the correcting term  $\mathbf{D}$  and the way it is computed. Our proposal is the following:

**Assumption 2** (Main modelling assumption). *The correcting term  $\mathbf{D}$  depends only on the unknown  $\mathbf{U}_c$ , i.e.,  $\mathbf{D} = \mathbf{D}(\mathbf{U}_c)$ , and it is designed under the condition that it is as close as possible to  $\mathbf{D}_{\text{ex},\alpha}$  in (18),  $\alpha = 1, \dots, N$ , at the training configurations.*

According to this assumption, the problem to be solved is of the form

$$\mathbf{A}\mathbf{U}_c + \mathbf{D}(\mathbf{U}_c) = \mathbf{F}. \quad (20)$$

What remains open is how to construct  $\mathbf{D}(\mathbf{U}_c)$  to make it close to  $\mathbf{D}_{\text{ex},\alpha}$ ,  $\alpha = 1, \dots, N$ , at the training configurations. In the following we describe two possibilities, namely, the case in which  $\mathbf{D}(\mathbf{U}_c)$  is a linear function and that in which  $\mathbf{D}(\mathbf{U}_c)$  is an ANN.

**3.4. Linear correction term: a least-squares approach.** The simplest model for  $\mathbf{D} = \mathbf{D}(\mathbf{U}_c)$  is a linear one:

$$\mathbf{D}(\mathbf{U}_c) = \mathbf{A}_{\text{ls}}\mathbf{U}_c + \mathbf{F}_{\text{ls}}, \quad (21)$$

where  $\mathbf{A}_{\text{ls}}$  is an  $m \times m$  matrix and  $\mathbf{F}_{\text{ls}}$  an array of  $m$  components, both to be determined. The motivation for such a simple expression of the correction term can be found in stabilized FE methods. Indeed, *the matrix structure of the stabilizing terms is precisely (21)*, derived from different heuristic arguments and with matrix  $\mathbf{A}_{\text{ls}}$  and array  $\mathbf{F}_{\text{ls}}$  possibly mesh-dependent.

Here we take as starting point expression (21) and design  $\mathbf{A}_{\text{ls}}$  and  $\mathbf{F}_{\text{ls}}$  according to Assumption 2. A possible way to fulfil it is *to impose that  $\mathbf{D}(\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha}))$  be as close as possible to  $\mathbf{D}_{\text{ex},\alpha}$ ,  $\alpha = 1, \dots, N$ , in a least-square (LS) sense*. Thus, we obtain  $\mathbf{A}_{\text{ls}}$  and  $\mathbf{F}_{\text{ls}}$  by solving the optimization problem:

$$[\mathbf{A}_{\text{ls}}, \mathbf{F}_{\text{ls}}] = \arg \min_{\mathbf{A}_d, \mathbf{F}_d} \sum_{\alpha=1}^N \|\mathbf{D}_{\text{ex},\alpha} - (\mathbf{A}_d \mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha}) + \mathbf{F}_d)\|^2, \quad (22)$$



where the norm involved in this expression can be taken as the standard Euclidian one, possibly scaled if the components of  $\mathbf{F}$  have different units in the particular example of application.

As it will be shown in the numerical examples, the simple approach given by (21)-(22) is effective and the accuracy of the solution  $\mathbf{U}_c$  to the resulting problem (20) is higher than that of problem (16). However, it suffers from two inconveniences: first, the limited scope of (21) and, secondly, the typical instabilities of the LS approximation (22). The latter are manifested for example by the fact that very different solutions  $[\mathbf{A}_{\text{ls}}, \mathbf{F}_{\text{ls}}]$  can be obtained by introducing small changes in the training set, such as the introduction of a new element in this set.

**3.5. Correcting term obtained from an artificial neural network.** A much more general approach than using (21)-(22) is to use an ANN to model  $\mathbf{D}(\mathbf{U}_c)$ . In this work we consider ANNs simply as *best fitting functions* to a set of data. In particular, the ANN we wish to construct aims to fit the inputs  $\{\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha})\}_{\alpha=1}^N$  and the outputs  $\{\mathbf{D}_{\text{ex},\alpha}\}_{\alpha=1}^N$  given by equation (18). Thus, the ANN to be constructed will be a highly nonlinear mapping

$$\mathbf{D} : \mathbb{R}^m \longrightarrow \mathbb{R}^m,$$

trained with the pairs  $\{\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha}), \mathbf{D}_{\text{ex},\alpha}\}_{\alpha=1}^N$ .

The use of ANNs clearly solves the two deficiencies mentioned for the linear approach with a LS fitting. First, ANNs are highly flexible, allowing to construct in an efficient way complex nonlinear functions. And, second, new terms can be added to the training set with little computational effort and in a very stable manner. Moreover, the correction term constructed from ANNs can be introduced both to linear *and nonlinear* problems.

We will not analyze here the particular implementation of the ANN we use, and we defer its description to the numerical examples. Let us just say that we have found sufficient to use simple topologies (architectures) of the network, with only a few hidden layers (shallow networks) and a few neurons per layer. For the moment, we have only considered feedforward architectures with back-propagation algorithms to determine the weights and the biases, and with the sigmoid as activation function. However, the flexibility of ANNs would allow for a lot of possibilities in the context of correcting coarse models that we are describing.

Let us describe how to apply ANNs to our problem:

- We design  $\mathbf{D}(\mathbf{U}_c)$  *component-wise*. This means that in fact we construct  $m$  fitting functions corresponding to the components of  $\mathbf{D}$ ,  $\mathbf{D}_j$ ,  $j = 1, \dots, m$ . These are the *outputs* of the ANN.
- The input of the ANN is in principle the set  $\{\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha})\}_{\alpha=1}^N$ . However, a particular implementation is possible in our context, since each  $\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha})$  can be considered as the vector of nodal values of a certain function  $u_{c,\alpha}(\mathbf{x})$  defined on the coarse mesh (scalar or vectorial). Thus, it makes perfect sense to use only a few components of  $\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha})$  when trying to obtain the  $j$ th component of  $\mathbf{D}$ , for example those associated to nodes linked to the node with the  $j$ th degree of freedom. Moreover, a way to introduce the effect of the neighboring nodes to that node can be through derivatives of  $u_{c,\alpha}(\mathbf{x})$ . Thus, considering for simplicity the case in which  $u_{c,\alpha}(\mathbf{x})$  is a scalar field, a possible input to compute  $\mathbf{D}_j$  can be  $u_{c,\alpha}(\mathbf{x}_j)$  and  $\nabla u_{c,\alpha}(\mathbf{x}_j)$ , where  $\mathbf{x}_j$  are the coordinates of the  $j$ th node,  $j = 1, \dots, m$ ,  $\alpha = 1, \dots, N$ .
- The weights and biases of the ANN are determined by minimizing a certain loss function. According to Assumption 2, we take as loss function

$$\mathcal{C} = \sum_{\alpha=1}^N \|\mathbf{D}_{\text{ex},\alpha} - \mathbf{D}(\mathbf{P}_{\text{fc}}(\mathbf{U}_{f,\alpha}))\|^2. \quad (23)$$

This means that *the ANN is trained with the configurations for which the exact correction is known*. As in the case of the LS fitting for linear corrections, the norm in this expression can be scaled when the components of  $\mathbf{D}$  have different physical meaning.

Let us stress again that the construction of  $\mathbf{D}(\mathbf{U}_c)$  only requires the fine solutions, but not the problem they solve. In fact, these could be obtained by means not necessarily numerical (such as experimental, analytical or semi-analytical). Our focus, however, is the case in which the training configurations are solved numerically with an approximation finer than the coarse one.

Once we have constructed the correction term  $\mathbf{D}(\mathbf{U}_c)$ , the problem to be solved is (20). While for the linear approximation with a LS fitting it reduces to the linear problem

$$(\mathbf{A} + \mathbf{A}_{\text{ls}})\mathbf{U}_c = \mathbf{F} - \mathbf{F}_{\text{ls}},$$

for ANNs problem (20) is highly non-linear. Let  $\mathbf{D}_{\text{ann}}$  be the correction term in this case. Standard linearization techniques can be used to linearize it. In our examples we have found efficient the simplest iterative procedure

$$\mathbf{A}\mathbf{U}_c^{(k)} = \mathbf{F} - \mathbf{D}_{\text{ann}}(\mathbf{U}_c^{(k-1)}), \quad (24)$$

where the superscript denotes the iteration counter. In the case in which the problem corresponds to the system to be solved at a certain time step, the obvious initialization is to take  $\mathbf{U}_c^{(0)}$  as the value of  $\mathbf{U}_c$  at the previous time step. This often suffices to improve the accuracy of the coarse solution and no iterations are needed, although it implies an explicit treatment of the correction term that could introduce instability problems in time.

The alternative to the simplest iteration (24) is a Newton-Raphson scheme:

$$(\mathbf{A} + \mathbf{J}(\mathbf{U}_c^{(k-1)}))\mathbf{U}_c^{(k)} = \mathbf{F} - \mathbf{D}_{\text{ann}}(\mathbf{U}_c^{(k-1)}) + \mathbf{J}(\mathbf{U}_c^{(k-1)})\mathbf{U}_c^{(k-1)}, \quad (25)$$

where

$$\mathbf{J}(\mathbf{U}_c) = \frac{\partial \mathbf{D}_{\text{ann}}(\mathbf{U}_c)}{\partial \mathbf{U}_c},$$

is the Jacobian of the ANN. This implies computing derivatives of the ANN with respect to the inputs, a feasible alternative that we have not explored.

#### 4. DISCRETE WAVE EQUATION WITH CORRECTION TERMS

This section intends to be the combination of the two previous sections; the general concept of the correcting terms based on training explained in Section 3 will be applied to the wave problems that have been explained in Section 2.

**4.1. Fine solutions and definition of the projection  $\mathbf{P}_{\text{fc}}$ .** As it has been explained in the previous section, a reference solution has to be provided in order to train the correction term for the coarse case. This solution is supposed to be more accurate than the coarse one, so it has to be supplied by a more accurate version of the same problem. In this case, this improvement on accuracy is produced by using a finer space discretization or a finer time discretization, in other words, with a smaller mesh size or a smaller time step. In the first case we would have  $M > m$ , whereas in the second  $M = m$ . The possibility of taking  $M < m$  will not be considered; it could correspond to a solution obtained on a mesh coarser than that of  $\mathbf{U}_c$  but with a smaller time step.

Once the coarse and the fine cases are created, the connection between them is crucial to perform the best possible training for the correction term. The two most important aspects at this point are what data we want to use from the fine case and how we project it to the coarse case.

On the one hand, three possible entry parameters for the correction term model could be the solution of the problem and its space derivatives, just the derivatives or just the

solution. Physical parameters or other derivatives of the solution of the problem could also be used to improve the training. It would also be possible to just enter the solution and make the model calculate the space derivatives from it; however, we rather prefer to facilitate the construction of the model by calculating it separately.

On the other hand, how we project these data to the coarse case will depend on the type of the fine case. If the fine case corresponds to a finer time discretization, but the mesh used for the spatial discretization is the same, then  $P_{fc}$  is simply the identity, i.e.,  $P_{fc}(U_f) = U_f$ . This projection needs to be performed at the time levels in which the fine solution is available and we wish to include in the training to solve the optimization problem in Eq. (22) in the LS approach or in Eq. (23) when ANNs are used. However, these problems require the system matrix  $A$  and the RHS vector  $F$  to be known at these time levels. For linear problems,  $A$  is constant, but  $F$  changes with the time level  $n$ . If the time step of the coarse model is not a multiple of the time step of the fine model, interpolation *in time* is required, for example interpolating the fine solution at the time levels in which the coarse solution is to be computed. In this case it has to be remarked that there are many options to perform the training, in particular it can be performed in certain time windows. For example, we can train the model with the fine solution during a certain number of time steps at the beginning of the simulation and then run the coarse model for longer time periods with the correction determined by this training. This is what we do in two of the numerical examples.

Let us consider now the situation in which the fine case corresponds to a finer space discretization. If the finer mesh is obtained from a hierarchical refinement of the coarse mesh,  $P_{fc}(U_f)$  can be taken simply as the restriction of the fine solution to the nodes of the coarse mesh that also belong to the fine mesh. In fact, this would be a particular case of projection on general meshes, namely, the *interpolation* of the fine solution to the coarse mesh. To do this, the position of each node of the coarse mesh on the fine one needs to be determined, and once this is done the value of the fine solution on the node of the coarse mesh can be calculated. This requires a standard search algorithm to obtain the element of the fine mesh to which a node of the coarse one belongs. Let us call  $I_{fc}$  the resulting interpolation.

The same search algorithm described above is required for another option to choose  $P_{fc}(U_f)$ , namely, the  $L^2$  projection from the fine mesh to the coarse one. Considering the unknown to be the acoustic pressure for conciseness, if  $p_f$  is the solution on the fine mesh and  $P_h(p_f)$  its  $L^2$  projection, it is obtained by solving

$$(q_h, P_h(p_f)) = (q_h, p_f) \quad \forall q_h \in \tilde{V}_{p,h}.$$

Let  $P_{2,fc}$  be the  $L^2$  projection at the algebraic level. The two options described are

$$P_{fc} = \begin{cases} I_{fc} & \text{in the case of standard interpolation} \\ P_{2,fc} & \text{in the case of the } L^2 \text{ projection} \end{cases}$$

Both options can be modified to incorporate restrictions of the projected solution [29].

The projection for the spatial derivatives of the solution of the problem can be done in the same way for both situations (finer solutions corresponding to finer time discretizations or to finer space discretizations). Physical parameters of the problem do not have to be projected, they can be directly sent from the fine case to the coarse one.

**4.2. Fully discrete system.** At this point, it is straightforward to obtain the fully discrete form of the problem. We just need to combine the matrix equations of the previous sections with the correction term of Eq. (21) in the case of the LS approach, or with the procedure of Eq. (24) in the case of ANNs.

Thus, the fully discrete form of the irreducible wave equation in the time domain for the LS training system is

$$\frac{2}{\delta t^2} \mathbf{M} \mathbf{U}^n + \mathbf{K} \mathbf{U}^n + \mathbf{A}_{\text{ls}} \mathbf{U}^n = \mathbf{F} - \mathbf{F}_{\text{ls}} + \frac{5}{\delta t^2} \mathbf{M} \mathbf{U}^{n-1} - \frac{4}{\delta t^2} \mathbf{M} \mathbf{U}^{n-2} + \frac{1}{\delta t^2} \mathbf{M} \mathbf{U}^{n-3}, \quad (26)$$

and the fully discrete form for the ANN training system, using an explicit treatment of the correction, is

$$\frac{2}{\delta t^2} \mathbf{M} \mathbf{U}^n + \mathbf{K} \mathbf{U}^n = \mathbf{F} + \frac{5}{\delta t^2} \mathbf{M} \mathbf{U}^{n-1} - \frac{4}{\delta t^2} \mathbf{M} \mathbf{U}^{n-2} + \frac{1}{\delta t^2} \mathbf{M} \mathbf{U}^{n-3} - \mathbf{D}_{\text{ann}}(\mathbf{U}^{n-1}). \quad (27)$$

Two remarks are in order:

- In Eq. (26) the correction term depends only on  $\mathbf{U}^n$  and in Eq. (27) only on  $\mathbf{U}^{n-1}$ , in this case due to the explicit treatment of the correction. It could be argued that this correction could depend also on *previous* values of the unknown. The fact that we do not consider this dependence in our model can be understood as a sort of orthogonality of the correction term with respect to the unknown. In fact, this type of correction is what is encountered in stabilized FE methods based on the VMS concept using orthogonal sub-grid scales (see [28] for a review on the topic).
- Note that in Eq. (27) no iterations have been used in this approximation. This is possible because the equation belongs to the time domain and the previous time step can play the role that used to play the previous iteration. An implicit treatment of the correction term would require a linearization of  $\mathbf{D}_{\text{ann}}(\mathbf{U}^n)$  as explained in the previous section.

We can proceed similarly in the case of the equations in the frequency domain. The corrected version of Eq. (13) using the LS approach is

$$-k^2 \mathbf{M} \mathbf{P} + (\mathbf{K}_s + \mathbf{A}_{\text{ls}}) \mathbf{P} = \mathbf{F}_s - \mathbf{F}_{\text{ls}}, \quad (28)$$

whereas the corrected version using ANNs and a simple fix point iteration is

$$-k^2 \mathbf{M} \mathbf{P}^{(i)} + \mathbf{K}_s \mathbf{P}^{(i)} = \mathbf{F}_s - \mathbf{D}_{\text{ann}}(\mathbf{P}^{(i-1)}), \quad (29)$$

where  $i$  corresponds now to the iteration counter.

**4.3. Practical implementation.** Let us explain now how our algorithm specifically works. First of all, it is important to differentiate the two main phases of the algorithm: the training phase and the execution phase. It is obvious that before starting the execution phase, the training one has to be completed at least for one case, so let us start with the training phase.

Once the coarse and fine training cases are ready, the training model is selected and the data that we want to use from the fine problem is defined; we then launch the coarse version of the problem with one processor and the fine version with another parallel processor. At this point, depending on which kind of training we want, the communication between the processors is different. On the one hand, in the situation of the finer mesh discretization and in the case of nested meshes, the processor that is launching the fine case only sends the desired information on the coincident nodes but at every single time step; in the case of non-nested meshes, non-trivial projections are required, as explained earlier. On the other hand, in the situation of the finer time discretization, the processor that is launching the fine case only sends the desired information at the coincident times, in other words, at the times when the coarse case solves the problem, so the coarse case has to wait for the fine case to do its extra time steps; again, in the case in which time levels never coincide, interpolation in time is required. Obviously, in this second case the data is sent to every node, assuming the mesh of the two cases to be exactly the same. When the processor of the coarse case receives the data, it is entered to the model. Depending on whether it is the LS or the ANN model, the information is treated in a different way; however, the objective of the model is the same in both situations. While this process is underway, the

model creates a new file and writes on it its specific correction parameters. Once the two cases are finished this new file is saved, this is where the information that will be used in the execution phase is stored.

Now that the training phase is finished, it is time to prepare the execution case. The possible changes in the execution cases are called configurations and have been explained in Section 3. When this new case is launched, the algorithm acts in a completely different way. Firstly, the data that has been written in the previous phase is read, so the model has all the information that needs to improve the solution of the problem. Secondly, the first time step or iteration of the problem is solved without any correction; nevertheless, this solution will be used to make the correction that is used to improve the solution of the second time step or iteration. Finally, each solution is used to calculate the correction of the next one until the problem ends. An important point is that if we are dealing with a problem that is unstable in its first time steps, we can add a delay so that the correction starts at the time we choose.

Additionally, there is something that can be significant in the performance of the correction algorithm, which is to use the data of the previous time step or iteration in the training phase instead of the data of the same time step or iteration. This is pretty obvious if we take into account that in the execution phase we are forced to use the data of the previous time step or iteration, so doing this we would be training the model in the same way as we use it.

## 5. NUMERICAL EXAMPLES

In this section, some numerical examples that illustrate the improvement of wave problem results with correcting terms based on training with ANN and LS models are shown. These specific examples have been solved with the irreducible form of the wave equation, the first two in the frequency domain and the other two in the time domain. Moreover, the two corresponding to the time domain have been improved using a finer time discretization, and the two corresponding to the frequency domain have been improved using a finer space discretization. However, other combinations as using mixed forms or improving time domain problems with finer space discretization could be considered. In all the examples we have used linear triangular elements for the spatial discretization, and the BDF2 scheme in the case of transient problems. When using ANNs, the sigmoid has been employed as activation function.

**5.1. Dipole localization coarse mesh training.** This first example consists in training a model to give a better solution for whatever is the position of the source in a circular domain that has a radius of  $R = 300$  m. More precisely, the source is a circular dipole of radius  $r = 10$  m. In this case, we use a simple ANN model of 2 layers and 3 neurons each, with a learning momentum of 0.1. This problem has been solved with the irreducible wave equation in the frequency domain. NRBCs of Sommerfeld's type, the same as in the following examples, have been used in order to avoid reflected waves.

The training set is a grid of dipoles launched one by one in the training phase. The coarse and the fine meshes and their element sizes can be seen in Figure 1 (a) and (b), respectively. The element size is smaller at the border of the dipoles in both meshes so as to have better results. The specific position of the training set dipoles and the external ones that will be used to test the model can be seen in Figure 1 (c).

It is easy to see that the results of the trained cases are more similar to the results of the finer cases than to the coarse cases. For example, if we look at the top of the circular domain of Figure 4, we can see that the fine and the trained cases have a clear last red stripe, while in the coarse case this red stripe is almost imperceptible. In the other dipoles similar qualitative differences can be found.

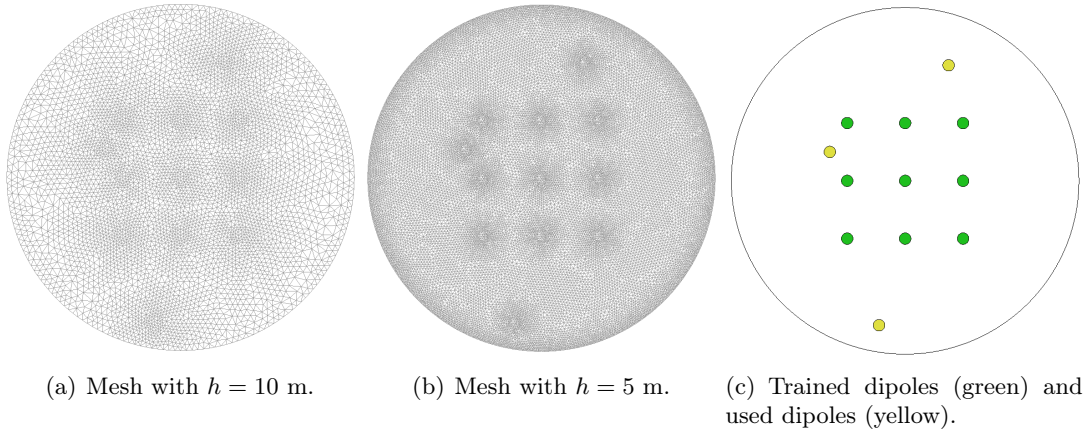


FIGURE 1.

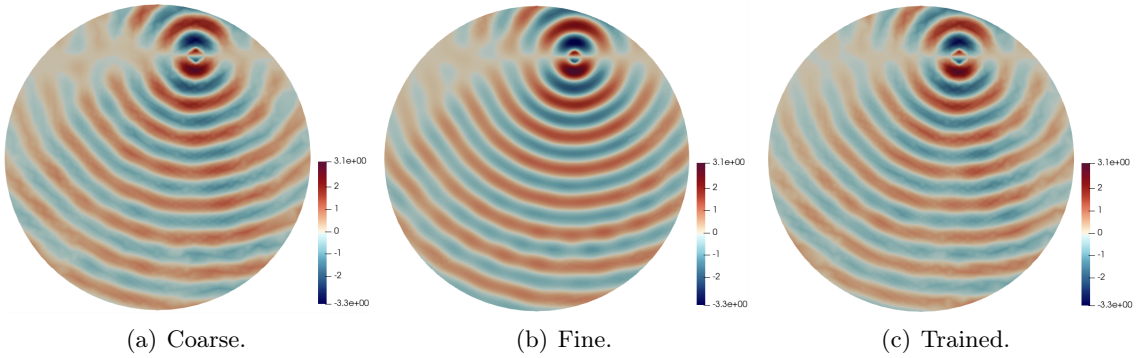


FIGURE 2. Imaginary part of the acoustic pressure for the first dipole.

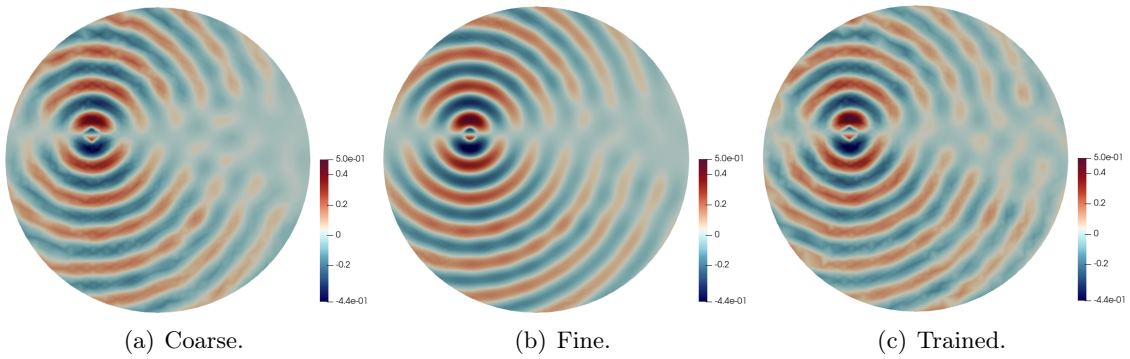


FIGURE 3. Imaginary part of the acoustic pressure for the second dipole.

**5.2. Plane wave propagation coarse mesh training.** The second example consists in training a model to give a better solution for a determined range of frequencies. We will consider a freely propagating two-dimensional plane wave, the same example that has been used in [7]. The domain is a square with side  $s = 1$  m. The element size is  $h = 0.05$  m for the coarse mesh and  $h = 0.025$  m for the fine mesh. This problem has been solved with the irreducible wave equation in the frequency domain and without stabilization, i.e., using the standard Galerkin method, to explore the possibility of the correction to alleviate the pollution error.

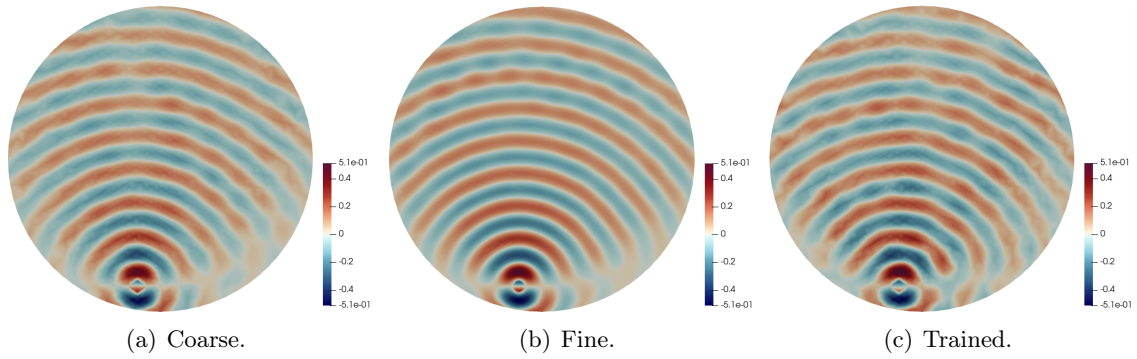
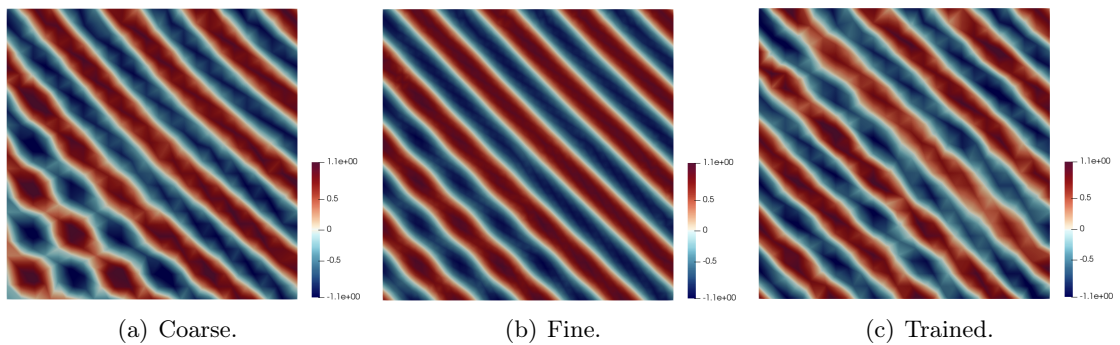


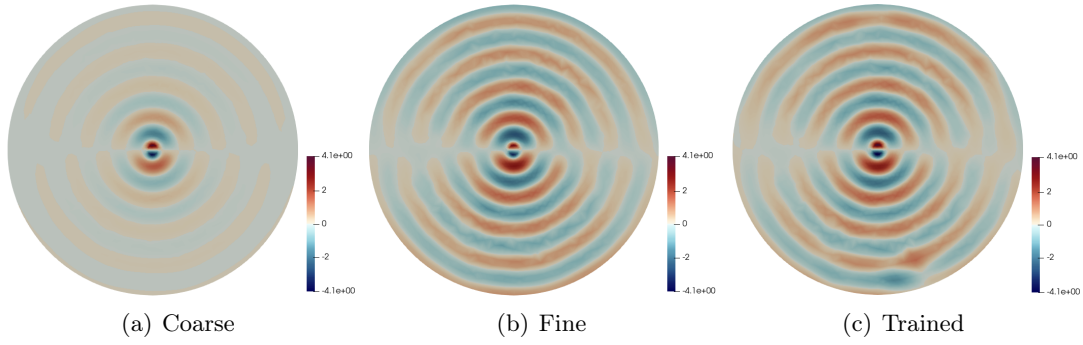
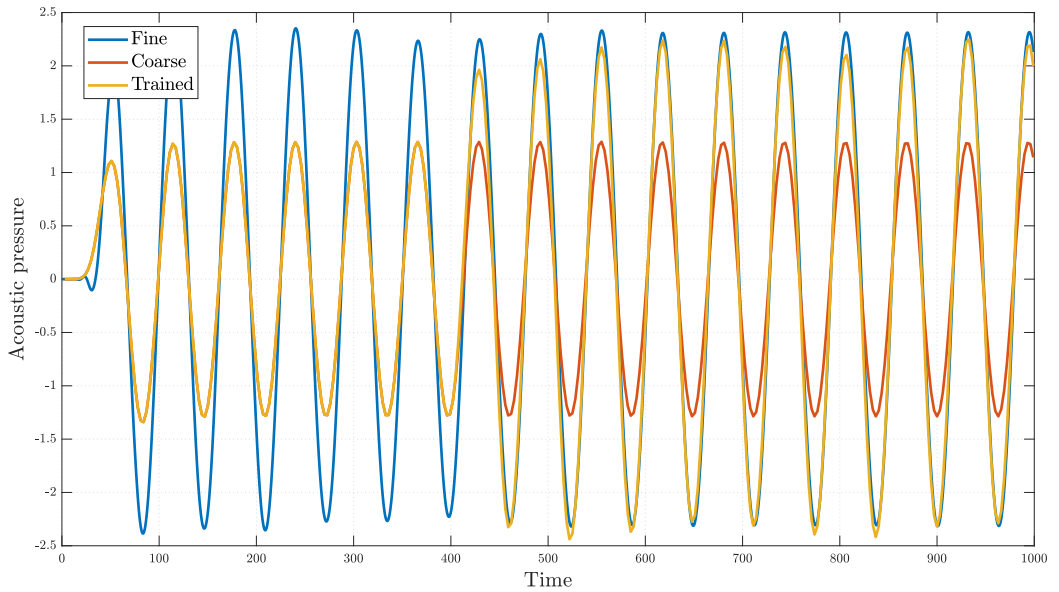
FIGURE 4. Imaginary part of the acoustic pressure for the third dipole.

FIGURE 5. Real part of the acoustic pressure for  $\omega = 32 \text{ s}^{-1}$ .

The training set is a range of frequencies that goes from  $\omega = 10 \text{ s}^{-1}$  to  $\omega = 42 \text{ s}^{-1}$  with intervals of  $4 \text{ s}^{-1}$ . This time, an ANN model of 3 layers and 3 neurons each has been used, with a learning momentum of 0.5. For the purpose of testing this model, the execution phase is launched with a frequency of  $\omega = 32 \text{ s}^{-1}$ , which does not appear in the training set.

In Figure 5, it can be seen that the coarse case has stability problems, the stripes are not perfectly straight and there is some pollution down left the domain. On the contrary, in the fine case these problems are solved due to the smaller element size. Although it is not exactly equal to the fine case, a clear improvement can be seen in the trained case, the lines are more straight than in the coarse case and there is no pollution error anywhere. Therefore, it is obvious that the model has made better the solution of this problem.

**5.3. Dipole coarse time training in the time domain.** This third example consists in training a model to give a better solution with any time step inside a time interval. The domain here is exactly the same that we have used in the first numerical example, the source is also the same and has the same size but this time it is fixed and positioned at the center of the circle. A triangular mesh of element size  $h = 5 \text{ m}$  around the source and progressively increased to  $h = 20 \text{ m}$  at the external borders has been used. NRBCs have been used in order to avoid reflected waves. The training set is composed by the solution in all the time steps between  $t = 400 \text{ s}$  and  $t = 700 \text{ s}$ , the finer time step for the whole training being fixed at  $\delta t = 0.1 \text{ s}$ . On the contrary, the time step of the coarse case is not fixed, so in the same run we can use different time steps for various time intervals. Specifically, a coarse time step of  $\delta t = 1 \text{ s}$  from  $t = 400 \text{ s}$  to  $t = 500 \text{ s}$  has been used, and  $\delta t = 2 \text{ s}$  from  $t = 500 \text{ s}$  to  $t = 600 \text{ s}$  and  $\delta t = 4 \text{ s}$  from  $t = 600 \text{ s}$  to  $t = 700 \text{ s}$  have been

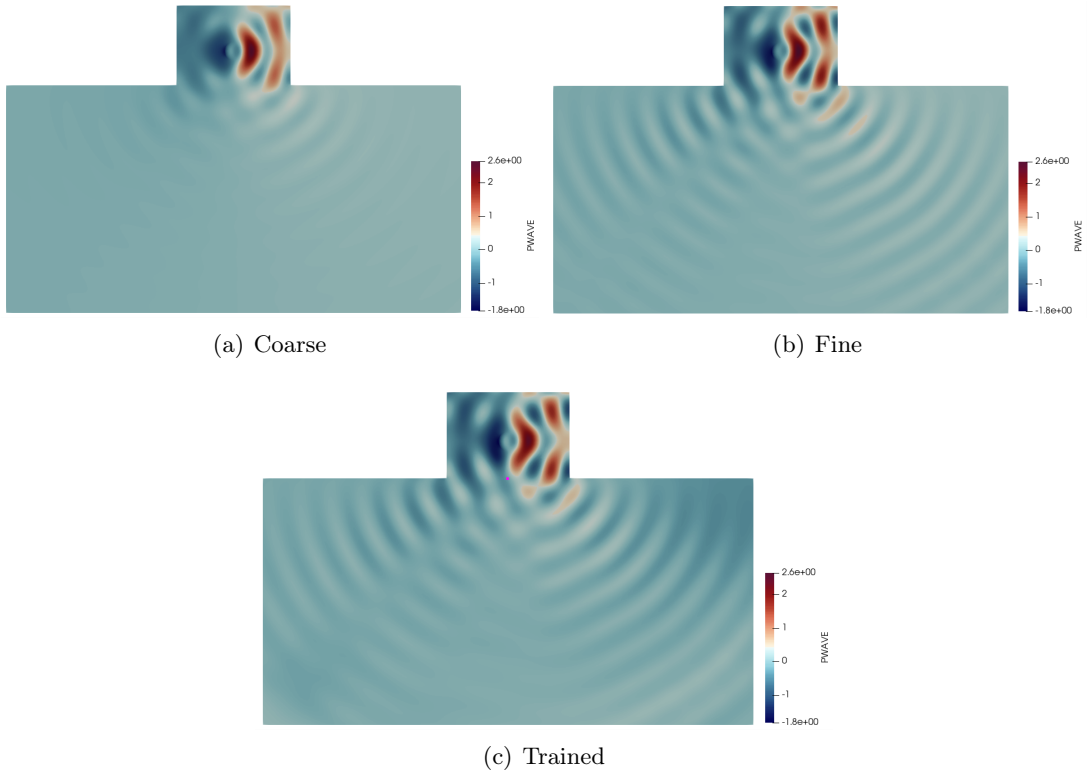
FIGURE 6. Acoustic pressure at  $t = 891$  s.FIGURE 7. Acoustic pressure of the three cases at point  $(0,100)$  m of the dipole that we can see in Figure 6.

employed. This problem has been solved with the irreducible wave equation in the time domain.

To show that not only ANNs can correct and improve the solutions of wave problems, in this example we have used a LS model. For the purpose of testing this model, the execution phase is launched with a fixed time step of  $\delta t = 3$  s, a time step that has not been used in the training phase.

On the one hand, a big qualitative improvement of the solution can be seen in Figure 6, which shows the solution of the problem for a concrete time step. Even though the stripes are perfectly located in the three situations, the amplitudes of the waves are clearly better in the fine and trained cases. On the other hand, in order to make a quantitative analysis, in Figure 7 the time evolution of the solution of the problem at one specific point is plotted for the three cases. It can be seen that from  $t = 0$  s to  $t = 400$  s, the trained case and the coarse case are exactly equal; this is because the correcting term is activated at  $t = 400$  s. At this point, the trained case starts to change, approaching to the behavior of the fine case almost perfectly. In spite of the fact that the training set ends at  $t = 700$  s, the



FIGURE 8. Acoustic pressure at  $t = 400$  s.

periodicity of the solution allows us to extend the time of the simulation as far as we want and get the same improvement.

**5.4. Dipole coarse time step training in a cavity.** This example is similar to the previous one, the kind of model that we want to create and the used equation are the same, so the training is performed in the same way. The source is also a circular dipole with radius  $r = 4$  m. A triangular mesh of element length  $h = 1$  m has been used. However, this time the domain is different and we will create the model using an ANN with 3 layers, 7 neurons per layer and with a learning momentum of 0.7. The domain in this case is a cavity composed by a big rectangle of length  $L = 200$  m and width  $W = 100$  m and a small rectangle of length  $l = 50$  m and width  $w = 35$  m centered at the top of the big one (see Figure 8). NRBCs have only been used on the borders of the big rectangle, and therefore there are reflecting waves inside the cavity, adding complexity to the problem.

The training set is composed by the solution in all the time steps between times  $t = 200$  s and  $t = 400$  s; the finer time step for the whole training is fixed and it is  $\delta t = 0.1$  s. As it has been done in the previous example, we use different time steps in the coarse case. Concretely, a coarse time step of  $\delta t = 0.2$  s has been used from  $t = 200$  s to  $t = 240$  s, and  $\delta t = 0.4$  s from  $t = 240$  s to  $t = 300$  s. For the purpose of testing the model, the execution phase is launched with a fixed time step of  $\delta t = 0.3$  s.

The results of this example are similar to the results of the previous one. On the one hand, a big qualitative improvement of the solution of the trained coarse model can be seen in Figure 8; the waves that are far from the source are almost imperceptible in the coarse case but perfectly defined in the fine and trained cases. On the other hand, in order to make a quantitative evaluation of the problem, the time evolution of the acoustic pressure at a specific point can be seen in Figure 9. Although the point is really close to the source, it can be seen that even there the improvement in the trained case is enormous. As in

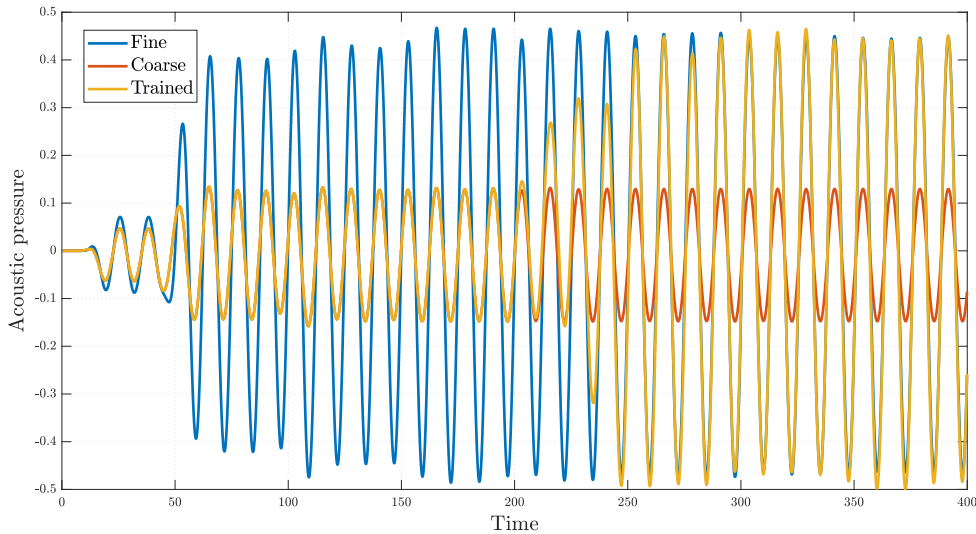


FIGURE 9. Acoustic pressure of the three cases at point (0,50) m, marked in Figure 8 c.

the previous example, the correcting term is activated at time  $t = 200$  s; at this point the amplitude of the oscillations starts to increase until it reaches the amplitude of the fine case, making the correction almost perfect.

## 6. CONCLUSIONS

In this paper, a general idea to enhance the solutions of coarse simulations through learning algorithms from fine solutions has been presented. This improvement is possible thanks to the training of correcting terms, that can be performed by LS or ANN models. This idea has been applied in this paper to the wave equation, both in the time and in the frequency domain.

The general concept of correction terms has been explained and some of its different possibilities have been described. Moreover, the specific correcting terms of the two different learning algorithms (LS and ANN) have been set and compared. The need for using iterative schemes in the case of using an ANN model has been explained, as well as the algorithmic issues of the application of this model in the time domain version of the wave equation.

Finally, the performance of the correcting terms based on training has been tested in different numerical examples. The first one corresponds to the Helmholtz equation trained with an ANN. The fine solutions have been chosen to be computed from a set of source positions with a fine mesh. It has been checked that the algorithm improves the coarse solution no matter which is the position of the source for a given frequency. The second example corresponds to the Helmholtz equation, also trained with an ANN. The fine solutions have been chosen to be those obtained from a set of different frequencies and computed with a fine mesh. Again, it has been checked that the algorithm improves the coarse solution no matter what is the frequency in a certain range. The third and the fourth examples are similar, both have been done with the irreducible form of the wave equation in the time domain, the former has a circular domain and it is solved with the LS model and the latter has a rectangular domain with a cavity and it is solved with an ANN model. The training cases have been chosen to be computed from a set of different coarse time steps while the fine solution in each case is computed using a fixed finer time

step. It has been shown in both examples that the algorithm improves the coarse solution, no matter what is the chosen coarse time step.

#### DECLARATION OF COMPETING INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

#### ACKNOWLEDGEMENTS

A. Fabra acknowledges the support obtained from the Spanish Government through the FPI grant RTI2018-098276-B-I00. J. Baiges gratefully acknowledges the support of the Spanish Government through the Ramón y Cajal grant RYC-2015-17367. R. Codina gratefully acknowledges the support received through the ICREA Acadèmia Research Program of the Catalan Government. This work was partially funded through the TOP-FSI: RTI2018-098276-B-I00 project of the Spanish Government. CIMNE is a recipient of a “Severo Ochoa Programme for Centers of Excellence in R&D” grant (CEX2018-000797-S) by the Spanish Ministry of Economy and Competitiveness.

#### REFERENCES

- [1] David J Lucia, Philip S Beran, and Walter A Silva. Reduced-order modeling: new approaches for computational physics. *Progress in aerospace sciences*, 40(1-2):51–117, 2004.
- [2] Francisco Chinesta, Pierre Ladeveze, and Elias Cueto. A short review on model order reduction based on proper generalized decomposition. *Archives of Computational Methods in Engineering*, 18(4):395–404, 2011.
- [3] Peng Chen and Christoph Schwab. Adaptive sparse grid model order reduction for fast Bayesian estimation and inversion. In *Sparse Grids and Applications-Stuttgart 2014*, pages 1–27. Springer, 2016.
- [4] Lihong Feng. Review of model order reduction methods for numerical simulation of nonlinear circuits. *Applied Mathematics and Computation*, 167(1):576–591, 2005.
- [5] John Burkardt, Max Gunzburger, and Hyung-Chun Lee. POD and CVT-based reduced-order modeling of Navier–Stokes flows. *Computer methods in applied mechanics and engineering*, 196(1-3):337–355, 2006.
- [6] Joan Baiges, Ramon Codina, Inocencio Castanar, and Ernesto Castillo. A finite element reduced-order model based on adaptive mesh refinement and artificial neural networks. *International Journal for Numerical Methods in Engineering*, 121(4):588–601, 2020.
- [7] Joan Baiges and Ramon Codina. A variational multiscale method with subscales on the element boundaries for the Helmholtz equation. *International Journal for Numerical Methods in Engineering*, 93(6):664–684, 2013.
- [8] Hector Espinoza, Ramon Codina, and Santiago Badia. A Sommerfeld non-reflecting boundary condition for the wave equation in mixed form. *Computer Methods in Applied Mechanics and Engineering*, 276:122–148, 2014.
- [9] Ramon Codina. Finite element approximation of the hyperbolic wave equation in mixed form. *Computer Methods in Applied Mechanics and Engineering*, 197(13-16):1305–1322, 2008.
- [10] Oriol Guasch, Marc Arnela, Ramon Codina, and Hector Espinoza. A stabilized finite element method for the mixed wave equation in an ALE framework with application to dipthong production. *Acta Acustica united with Acustica*, 102(1):94–106, 2016.
- [11] Peter Monk, Joachim Schöberl, and Astrid Sinwel. Hybridizing Raviart-Thomas elements for the Helmholtz equation. *Electromagnetics*, 30(1-2):149–176, 2010.
- [12] Joan Baiges, Ramon Codina, and Arnau Fabra. Stability and convergence of stabilized finite element methods for the helmholtz equation in mixed form. *Submitted*.
- [13] Luiz C Wrobel. *The boundary element method, volume 1: Applications in thermo-fluids and acoustics*, volume 1. John Wiley & Sons, 2002.
- [14] Stephen Martin Kirkup. *The boundary element method in acoustics*. Integrated sound software, 2007.
- [15] Mingsian R Bai. Application of BEM (boundary element method)-based acoustic holography to radiation analysis of sound sources with arbitrarily shaped geometries. *The Journal of the Acoustical Society of America*, 92(1):533–549, 1992.
- [16] Isaac Harari and Frédéric Magoulès. Numerical investigations of stabilized finite element computations for acoustics. *Wave Motion*, 39(4):339–349, 2004.

- [17] Thomas JR Hughes. Multiscale phenomena: Green’s functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer methods in applied mechanics and engineering*, 127(1-4):387–401, 1995.
- [18] Thomas JR Hughes, Gonzalo R Feijóo, Luca Mazzei, and Jean-Baptiste Quincy. The variational multiscale method—a paradigm for computational mechanics. *Computer methods in applied mechanics and engineering*, 166(1-2):3–24, 1998.
- [19] Joan Baiges, Ramon Codina, and Sergio Idelsohn. Reduced-order subscales for POD models. *Computer Methods in Applied Mechanics and Engineering*, 291:173–196, 2015.
- [20] Steven J Miller. The method of least squares. In *The Probability Lifesaver*, pages 625–635. Princeton University Press, 2017.
- [21] Steffen Nissen et al. Implementation of a fast artificial neural network library (FANN). *Report, Department of Computer Science University of Copenhagen (DIKU)*, 31:29, 2003.
- [22] Pradeep Ramuhalli, Lalita Udpa, and Satish S Udpa. Finite-element neural networks for solving differential equations. *IEEE transactions on neural networks*, 16(6):1381–1392, 2005.
- [23] Haidong Qu, Xuan Liu, and Zihang She. Neural network method for fractional-order partial differential equations. *Neurocomputing*, 414:225–237, 2020.
- [24] Angelos A Anastassi. Constructing Runge–Kutta methods with the use of artificial neural networks. *Neural Computing and Applications*, 25(1):229–236, 2014.
- [25] Zeng Zhe-Zhao, Wang Yao-Nan, and Wen Hui. Numerical integration based on a neural network algorithm. *Computing in science & engineering*, 8(4):42–48, 2006.
- [26] Jan S Hesthaven and Stefano Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.
- [27] Antonio Stanzola, Simon R Arridge, Ben T Cox, and Bradley E Treeby. A Helmholtz equation solver using unsupervised learning: Application to transcranial ultrasound. *Journal of Computational Physics*, 441:110430, 2021.
- [28] Ramon Codina, Santiago Badia, Joan Baiges, and Javier Principe. *Variational Multiscale Methods in Computational Fluid Dynamics*, in Encyclopedia of Computational Mechanics (eds E. Stein, R. Borst and T. J. R. Hughes), pages 1–28. John Wiley & Sons Ltd., 2017.
- [29] Arnau Pont, Ramon Codina, and Joan Baiges. Interpolation with restrictions between finite element meshes for flow problems in an ALE setting. *International Journal for Numerical Methods in Engineering*, 110:1203–1226, 2017.
- [30] Ramon Codina. Analysis of a stabilized finite element approximation of the Oseen equations using orthogonal subscales. *Applied Numerical Mathematics*, 58(3):264–283, 2008.
- [31] Ramon Codina. Stabilized finite element approximation of transient incompressible flows using orthogonal subscales. *Computer methods in applied mechanics and engineering*, 191(39-40):4295–4321, 2002.
- [32] Ramon Codina. A stabilized finite element method for generalized stationary incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 190(20-21):2681–2706, 2001.
- [33] Ramon Codina and Javier Principe. Dynamic subscales in the finite element approximation of thermally coupled incompressible flows. *International journal for numerical methods in fluids*, 54(6-8):707–730, 2007.