A reduced order model for domain decompositions with non–conforming interfaces

Elena Zappon · Andrea Manzoni · Paola Gervasio · Alfio Quarteroni

the date of receipt and acceptance should be inserted later

Abstract In this paper, we propose a reduced–order modeling strategy for two–way Dirichlet–Neumann parametric coupled problems solved with domain–decomposition (DD) sub–structuring methods. We split the original coupled differential problem into two sub–problems with Dirichlet and Neumann interface conditions, respectively. After discretization by, e.g., the finite element method, the full–order model (FOM) is solved by Dirichlet–Neumann iterations between the two sub–problems until interface convergence is reached. We then apply the reduced basis (RB) method to obtain a low–dimensional representation of the solution of each sub–problem. Furthermore, we apply the discrete empirical interpolation method (DEIM) at the interface level to achieve a fully reduced–order representation of the DD techniques implemented. To deal with non–conforming FE interface discretizations, we employ the INTERNODES method combined with the
interface DEIM reduction. The reduced–order model (ROM) is then solved by sub–iterating between the two reduced–order sub–problems until the convergence of the approximated high–fidelity interface solutions. The ROM scheme is numerically verified on both steady and unsteady coupled problems, in the case of non–conforming FE interfaces.

**Keywords** Two–way coupled problems, Dirichlet–Neumann coupling, Reduced order modeling, Discrete empirical interpolation method, Interface non–conformity, Domain–decomposition, Reduced basis method

**Mathematics Subject Classification (2010)** 65M99 · 65P99 · 68U01

1 Introduction

Reduced order modeling (ROM) techniques encompass a wide class of numerical methods able to solve differential problems several orders of magnitude faster than conventional, high–fidelity full order models (FOMs), achieving real–time decision–making and operational modeling in several contexts, ranging from fluid dynamics [2,9,8,27,44,54] to biomedical engineering [7,28,56].

In several cases, when dealing with complex – possibility multi–physics – problems, domain decomposition (DD) [63,62] techniques are necessary to (i) split the domain into two or more regions in which either the same or different methods are used to approximate the solution [20,32,57,64], to (ii) solve systems that stem from the assembly of independently generated meshes [34,51,55] or to (iii) frame coupled problems where the physical nature of the involved sub–models is very different [22]. In the last two cases, interface non–conformity issues can arise, and ad–hoc techniques, such as e.g. the MORTAR method [17,14,35] or the INTERNODES method [23,29,31] have to be implemented in order to ensure the correct exchange of information at the interface.

Domain decomposition schemes coupled with ROM techniques, especially the reduced basis (RB) method [12,58,59], have been first used in [46], where ROMs were applied only on small regions of the domains, for instance for PDEs with discontinuous solutions in those regions where the discontinuity occurs. Similar strategies have been implemented to tackle the numerical simulation of problems in fluid dynamics [3,39,50,58,63,55,66], aerospace engineering [43] and structural mechanics [42,21], as well as for the optimization of complex systems [4,3]. In order to reduce the global parametrized problem, other methods such as the reduced basis element methods (RBEM) [18,45], the reduced basis hybrid (RBHM) method [59], and the static condensation methods (SCRBEM) [25,38] have been developed. A common feature shared by several of these works is the application of a reduced order model in small parts of the domains, through an ad–hoc definition of a small set of basis functions in each subdomain able to preserve the solution continuity along the internal subdomain interfaces. An effort to include different interface conditions comes with the RDF method [40], where local basis functions are employed on each selected sub–problem and interface conditions are treated in high–fidelity form through the application of different techniques, e.g. Lagrangian multipliers or Fourier basis functions, for sub–iterating schemes and/or solving the coupled model in monolithic form, naturally imposing interface constraints. In [19] also Dirichlet and Neumann data on conforming interfaces are considered in reduced form, while the Dirichlet–Neumann DD iterations structure is preserved by the reduced algorithm, whereas an algebraic–splitting approach between internal and interface nodes is employed in the domain–decomposition least–squares Petrov–Galerkin (DD–LSPG) approach [37]. However, to the best of our knowledge, interface non–conformity has never been considered, given its intrinsic complexity.

In this work, we present a Dirichlet–Neumann DD–ROM relying on the RB method able to transfer interface data across non–conforming interfaces. In particular, we consider parametric second–order elliptic and/or parabolic problems solved with Dirichlet–Neumann sub–structuring DD algorithms. Therefore, we
split the problem domain into two non-overlapping subdomains with a common interface and we define two parametrized sub-problems with Dirichlet or Neumann interface conditions. RB methods are then applied at the sub-problems level to approximate both the sub-problems solution together with the Dirichlet or Neumann interface conditions.

To solve the coupled problem and compute the snapshots (for different parameter instances) required to train our ROM, we rely on the finite element (FE) method as a high-fidelity FOM. In particular, the FOM solution is sought by sub-iterating between the two sub-models solutions until convergence, being this latter is reached when the difference between the solution at both sides of the interface falls under a prescribed tolerance. In the presence of a non-conforming interface, FOM solutions can be sought through FE-based methods such as the MORTAR method or the INTERNODES. In our code, however, we surrogate the non-conforming coupling by solving for each selected parameter instance the coupled problems twice: indeed, we define two possible FE discretizations in the complete domain (each one being obtained by extending to the other subdomain the spatial discretization set in the first one) and compute the solution of each sub-problem using both discretizations, one for each simulation, featuring interface conformity. Then, we extract the solution of each of the sub-problems (in the discretization originally set for the corresponding subdomain), therefore obtaining two sets of solution snapshots as well as the corresponding Dirichlet and Neumann traces at the common interface that can be seen as the original model solution when interface nonconformity is considered (see Remark 3 and Fig. 1).

The RB method is then applied to define a low-dimensional representation of the solution in each subdomain, while the discrete empirical interpolation method (DEIM) is applied to both Dirichlet and Neumann interface data in order to achieve a fully reduced-order representation of the DD method adopted. In the online phase, a Galerkin projection is used to reduce the sub-problems’ dimension obtaining two reduced-order sub-problems. The solution of the coupled problem, for any new parameter instance, is then found by iterating between the solutions of the two reduced sub-problems until (a suitable norm of) the difference between the two solutions at the interface, once traced back at the high-fidelity level, falls below a prescribed tolerance. In this phase, we transfer Dirichlet and Neumann interface data by applying the INTERNODES method and by using the DEIM as well as a low-order piecewise constant interpolation. This approach, which extends the work presented in [67], is still modular, and allows us to achieve a complete reduction of the model at hand, which can be seen as a two-way coupled model, including interface non-conforming grid cases.

The organization of the paper is the following: in Section 2 we present the formulation and the high-fidelity discretization of the parametrized problem, considering both steady and unsteady cases. Section 3 is devoted to the reduced-order formulation of the two sub-problems, while Section 4 is dedicated to the interface Dirichlet and Neumann reduced formulation. In Section 5 the algorithm is numerically verified by means of two test cases dealing with second-order linear PDEs, considering both an elliptic and a parabolic problem. Section 6 then reports some final remarks and possible perspectives of this work.

2 Two-way coupled problem and its FE discretization

We introduce the parameter-dependent two-way coupled problem. Such a problem can arise from the application of splitting domain decomposition methods to single-physics or multi-physics models. For simplicity, let us consider the single-physics case – the same procedure can be considered for the multi-physics case – defined on an open bounded domain \( \Omega \subset \mathbb{R}^n \), \( n = 2, 3 \) with Lipschitz boundary \( \partial \Omega \), \( \partial \Omega_D \) and \( \partial \Omega_N \) denote the Dirichlet and the Neumann boundary, respectively, such that \( \partial \Omega_D \cup \partial \Omega_N = \partial \Omega \) and \( \partial \Omega_D \cap \partial \Omega_N = \emptyset \). Given the set of parameters \( \mu \in \mathcal{P}^d \subset \mathbb{R}^d, \ d \geq 1 \), we search for \( u(\mu) \) on \( \Omega \) such that
\[
\begin{aligned}
&\mathcal{L}(\mu)u(\mu) = f(\mu) \quad \text{in } \Omega \\
u(\mu) = g_D(\mu) \quad \text{on } \partial \Omega_D \\
\partial\mathcal{L}(\mu)u(\mu) = g_N(\mu) \quad \text{on } \partial \Omega_N,
\end{aligned}
\]

where \(\mathcal{L}(\mu)\) is a second–order elliptic operator, \(f(\mu)\), \(g_D(\mu)\) and \(g_N(\mu)\) are functions defined on \(\Omega, \partial \Omega_D\) and \(\partial \Omega_N\), respectively, and \(\partial\mathcal{L}(\mu)u(\mu)\) is the conormal derivative associated with the operator \(\mathcal{L}(\mu)\) on \(\partial \Omega\).

Moreover, different mesh sizes (tetrahedra) or quads (quadrilaterals or hexahedra), depending on the mesh size, a positive parameter \(h_i > 0\). For each \(i = 1, 2\), we denote \(\partial \Omega_i = \partial \Omega_i \cap \partial \Omega_D\) and \(\partial \Omega_i \cap \partial \Omega_N\). The two-domains formulation of model (1) reads (2):

\[
\begin{aligned}
&\mathcal{L}_i(\mu)u_i(\mu) = f(\mu) \quad \text{in } \Omega_i, \quad i = 1, 2 \\
u_i(\mu) = g_D(\mu) \quad \text{on } \partial \Omega_i \cap \partial \Omega_D \\
\partial\mathcal{L}_i(\mu)u_i(\mu) = g_N(\mu) \quad \text{on } \partial \Omega_i \cap \partial \Omega_N \\
\partial\mathcal{L}_i(\mu)u_i(\mu) + \partial\mathcal{L}_i(\mu)u_i(\mu) = 0 \quad \text{on } \Gamma_i,
\end{aligned}
\]

where \(\mathcal{L}_i\) is the differential operator acting on functions defined on the sub-domain \(\Omega_i\).

The Dirichlet-Neumann iterative scheme [15][21] is applied to solve problem (2). Therefore, starting from the initial guesses \(u_1^0(\mu)\) and \(u_2^0(\mu)\), for each \(k \geq 0\), we search for \(u_1^{k+1}(\mu)\) on \(\Omega_1\) and \(u_2^{k+1}(\mu)\) on \(\Omega_2\) such that

\[
\begin{aligned}
&\mathcal{L}_1(\mu)u_1^{k+1}(\mu) = f(\mu) \quad \text{in } \Omega_1 \\
u_1^{k+1}(\mu) = g_D(\mu) \quad \text{on } \Gamma \\
u_1^{k+1}(\mu) = g_D(\mu) \quad \text{on } \partial \Omega_1, \quad \text{D} \\
\partial\mathcal{L}_1(\mu)u_1^{k+1}(\mu) = g_N(\mu) \quad \text{on } \partial \Omega_1, \quad \text{N}
\end{aligned}
\]

and

\[
\begin{aligned}
&\mathcal{L}_2(\mu)u_2^{k+1}(\mu) = f(\mu) \quad \text{in } \Omega_2 \\
u_2^{k+1}(\mu) = g_D(\mu) \quad \text{on } \Gamma \\
u_2^{k+1}(\mu) = g_D(\mu) \quad \text{on } \partial \Omega_2, \quad \text{D} \\
\partial\mathcal{L}_2(\mu)u_2^{k+1}(\mu) = g_N(\mu) \quad \text{on } \partial \Omega_2, \quad \text{N}
\end{aligned}
\]

where \(\partial\mathcal{L}_i(\mu)u(\mu)\) is the conormal derivative associated with the operator \(\mathcal{L}_i(\mu)\) on \(\partial \Omega_i\). Moreover, a relaxation technique [21] is usually applied to ensure and accelerate the scheme convergence.

Then, we denote the problem and the corresponding solution on \(\Omega_1\) as slave model and slave solution, and the ones in \(\Omega_2\) as master model and master solution, respectively.

Defining \(V = H_0^1(\Omega)\), for each \(i = 1, 2\), we first define the local spaces

\[
V_i = \{v \in H_1^1(\Omega_i) \mid v = 0 \text{ on } \partial \Omega_1, \partial \Omega_D\} \quad \text{and} \quad V_i^0 = \{v \in V_i \mid v = 0 \text{ on } \partial \Omega_D\}.
\]

We consider two a–priori independent discretizations \(\mathcal{T}_h\) and \(\mathcal{T}_h\) on the domains \(\Omega_1\) and \(\Omega_2\) that can imply a mesh non–conformity at the interface. For instance, \(\mathcal{T}_h\) can be made of simplices (triangles or tetrahedra) or quads (quadrilaterals or hexahedra), depending on the mesh size, a positive parameter \(h_i > 0\). Moreover, different mesh sizes \(h_1\) and \(h_2\), or different polynomial degrees \(p_1\) or \(p_2\), can be selected. Then, we
denote the internal interfaces of $\Omega_1$ and $\Omega_2$ induced by $\mathcal{T}_{h_1}$ and $\mathcal{T}_{h_2}$ as $\Gamma_1$ and $\Gamma_2$, respectively: we talk of geometrical conformity if $\Gamma_1 = \Gamma_2$ and of geometrical non-conformity if $\Gamma_1 \neq \Gamma_2$. Finally, we assume that for any $T_{i,m} \in \mathcal{T}_{h_1}$, $\partial T_{i,m} \cap \partial \Omega$ fully belongs to $\partial \Omega_{1,D}$ or $\partial \Omega_{1,N}$. According to the test cases in Section 5 hereon we only consider quads elements.

For each partition $\mathcal{T}_h$, we define the finite element approximation spaces as

$$X^p_{h_i} = \{ v \in C^0(\mathcal{T}_h) : v|_{T_{i,m}} \in \mathbb{P}_p, \forall T_{i,m} \in \mathcal{T}_{h_i} \},$$

in which $F_{i,m}$ is a smooth bijection that maps the reference quad $(-1,1)^n$ into the quad $T_{i,m}$, and $p_i$ are chosen integers. The finite-dimensional spaces to define the discrete formulation of the exploited problems will be

$$V_{h_i} = \{ v \in X^p_{h_i} : v|_{\partial \Omega_{1,D}} = 0 \}, \quad V^0_{h_i} = \{ v \in V_{h_i}, v|_{\Gamma} = 0 \}, \quad i = 1, 2,$$

while the spaces of traces on $\Gamma$ are

$$Y_{h_i} = \{ \lambda = v|_{\Gamma}, v \in X^p_{h_i} \} \quad \text{and} \quad A_{h_i} = \{ \lambda = v|_{\Gamma}, v \in V_{h_i} \}.$$

When the Dirichlet boundaries and the interface share common DoFs, the spaces $Y_{h_1}$ and $A_{h_1}$ (as well as $Y_{h_2}$ and $A_{h_2}$) are different and should be addressed separately, whereas when $\Gamma_1 \cap \Omega_{1,D} = \emptyset$, the two spaces $Y_{h_1}$ and $A_{h_1}$ (as well as $Y_{h_2}$ and $A_{h_2}$) coincide. Since the problem we are interested in falls in the first situation, from now on we assume that $A_{h_1} = Y_{h_1}$.

We also introduce two independent transfer operators able to exchange information between the independent grids on the interface $\Gamma$, namely

$$\Pi_{12} : Y_{h_2} \to Y_{h_1} \quad \text{and} \quad \Pi_{21} : Y_{h_1} \to Y_{h_2}.$$

In the non-conforming case, if $\Gamma_1$ and $\Gamma_2$ coincide, such operators could be the classical Lagrange interpolation operators, while $\Pi_{jk}$ are the identity operators when the meshes are conforming. Instead, if the meshes are non-conforming and $\Gamma_1 \neq \Gamma_2$, $\Pi_{12}$ and $\Pi_{21}$ could be, e.g., Rescaled Localized Radial Basis Function operators, as for the INTERNODES [23,30,31].

To exploit a FE–Galerkin approximation to set the high-fidelity FOM and get the algebraic formulation of problems [3] and [4], it is useful to consider local vectors and matrices. In particular, we define the following set of indices associated with the nodes $x_j$ of the mesh in $\Omega_i$:

$$I_{\mathcal{T}_i} = \{ j \in I_{\mathcal{T}_i} : x_j \in \mathcal{T}_i \}, \quad I_{\Gamma_i} = \{ j \in I_{\mathcal{T}_i} : x_j \in \partial \Omega_{1,D,i} \} \quad \text{and} \quad I_1 = \{ j \in I_{\mathcal{T}_1} : x_j \in \mathcal{T}_1 \setminus (\partial \Omega_{D,1} \cup \hat{\Gamma}_1) \},$$

being $N_i$ the cardinality of $I_{\mathcal{T}_i}$. Moreover, we denote by $\tilde{N}_i$ the cardinality of $I_i$. Note that the definitions of $I_1$ and $I_2$ are different; indeed, $I_1$ represents all the nodes in $\Omega_1$ minus the nodes on both the interface and the Dirichlet portion of the boundary of $\Omega_1$, whereas $I_2$ contains all the nodes in $\Omega_2$ minus only the nodes on the Dirichlet portion of the boundary of $\Omega_2$.

Then, for each $i = 1, 2$, we set the local stiffness matrices $A_i(\mu)$ so that

$$A_{i,i}(\mu) = A_i(I_i, I_i; \mu)$$

is the submatrix of $A_i(\mu)$ of the rows and columns of $A_i(\mu)$ whose indices belong to $I_i$, i.e. the internal nodes of $\Omega_i$ or those on $\partial \Omega_{i,N}$. Similarly, we can define $A_{i,\Gamma_i}(\mu) = A_i(I_{\Gamma_i}, I_{\Gamma_i}; \mu)$, $A_{i,\Gamma_i}(\mu) = A_i(I_{\Gamma_i}, I_{\Gamma_i}; \mu)$, $A_{\Gamma_1,i}(\mu) = A_i(I_{\Gamma_1}, I_i; \mu)$, and $A_{i,D}(\mu) = A_i(I_i, I_{D,i}; \mu)$. 

Moreover, if \( f_{N,i}(\mu) \) and \( u_{N,i}(\mu) \) are the right–hand side vector (including the Neumann data \( g_N(\mu) \)) and the vector of degrees of freedom of the approximated solution in \( \Omega_i \), respectively, we set

\[
f_i(\mu) = f_{N,i}(I_i; \mu), \quad f_{\Gamma_i}(\mu) = f_{N,i}(I_{\Gamma_i}; \mu),
\]
\[
u_i(\mu) = u_{N,i}(I_i; \mu), \quad u_{\Gamma_i}(\mu) = u_{N,i}(I_{\Gamma_i}; \mu).
\]

Then we can explicit the algebraic form of (3) as: for each \( i \),

\[
\begin{align*}
\begin{bmatrix} A_{1,1}(\mu) & A_{1,D}(\mu) \\
A_{D,1}(\mu) & A_{D,D}(\mu)
\end{bmatrix}
\begin{bmatrix} u^{k+1}_1(\mu) \\
A_{D,1}(\mu)
\end{bmatrix} &= f_1(\mu) - A_{1,D}(\mu)g_{D,1}(\mu) - A_{1,\Gamma_i}(\mu)u^{k+1}_{\Gamma_i}(\mu) \\
u^{k+1}_{\Gamma_i}(\mu) &= \mathbb{R}_{12}u^{k+1}_{\Gamma_i}(\mu).
\end{align*}
\]

(9)

where \( \mathbb{R}_{12} \) is the rectangular matrix associated with \( H_{12} \), and \( A_{D,1}(\mu) \) the vector whose elements are the evaluation of \( g_D(\mu) \) on the Dirichlet boundary nodes of \( \Omega_i \).

The algebraic formulation of problem (4) reads as: for each \( i \), find \( u^{k+1}_i(\mu) \) such that

\[
\begin{align*}
\begin{bmatrix} A_{2,2}(\mu) & A_{2,D}(\mu) \\
A_{D,2}(\mu) & A_{D,D}(\mu)
\end{bmatrix}
\begin{bmatrix} u^{k+1}_i(\mu) \\
u^{k+1}_i(\mu)
\end{bmatrix} &= f_2(\mu) + \mathbb{E}_2u^{k+1}_{\Gamma_i}(\mu) - A_{2,\Gamma_i}(\mu)g_{D,2}(\mu) \\
u^{k+1}_{\Gamma_i}(\mu) &= -\mathbb{M}_{\Gamma_i}\mathbb{R}_{21}M^{-1}_{\Gamma_i}u^{k+1}_{\Gamma_i}(\mu),
\end{align*}
\]

(10)

where, for \( i = 1, 2 \), \( \mathbb{M}_{\Gamma_i} \) is the interface mass matrix on \( \Gamma_i \), \( \mathbb{R}_{21} \) is the matrix associated with \( H_{21} \), \( A_{D,2}(\mu) \) is the vector whose elements are the evaluation of \( g_D(\mu) \) at the Dirichlet boundary nodes of \( \Omega_2 \),

\[
u^{k+1}_i(\mu) = \left( A_i(\mu)u^{k+1}_{N,i}(\mu) - f_{\Gamma_i}(\mu) \right)_{|\Gamma_i},
\]

(11)

is the residual at the interface \( \Gamma_i \), and

\[
\begin{bmatrix} u^{k+1}_{N,1}(\mu) \\
u^{k+1}_{\Gamma_1}(\mu) \\
u^{k+1}_{D,1}(\mu)
\end{bmatrix}
\]

\[
\begin{bmatrix} u^{k+1}_{N,2}(\mu) \\
u^{k+1}_{D,2}(\mu)
\end{bmatrix}
\]

are the complete solutions of the sub–problems (3) and (4), and, finally, \( \mathbb{E}_2 \in \mathbb{R}^{N_2 \times N_2} \) is the extension matrix from the interface to all the domain DoFs in \( I_2 \) such that \( (\mathbb{E}_2)_{ij} = 1 \) only if \( i \in I_2, j \in I_{\Gamma_2} \), otherwise it is null.

The meaning of (10) is, therefore, that of:

1. moving the residual vector \( r^{k+1}_{\Gamma_i}(\mu) \) from the dual space to the primal one by the product \( M^{-1}_{\Gamma_i}r^{k+1}_{\Gamma_i}(\mu) \);
2. interpolating on the primal space by applying \( \mathbb{R}_{12} \);
3. coming back to the dual space by applying the mass matrix \( M_{\Gamma_i} \).

This procedure is the one employed by the INTERNODES method [31]. Indeed, INTERNODES is a general–purpose method to deal with non–conforming discretizations of partial differential equations on 2D and 3D regions partitioned into two (or several) disjoint subdomains. Differently from Mortar methods, which are based on projection, the idea of INTERNODES consists in exchanging the information between the two subdomains by resorting to both two independent interpolation operators \( \mathbb{R}_{12} \) and \( \mathbb{R}_{21} \), and two local mass matrices at the interfaces \( \mathbb{M}_{\Gamma_i} \) and \( \mathbb{M}_{\Gamma_2} \). To transfer the trace \( u_{\Gamma_i}(\mu) \) of the solution from \( \Gamma_2 \) to \( \Gamma_1 \), only the interpolation operator \( \mathbb{R}_{12} \) is needed. On the contrary, for what concerns the balance of the residuals (which are the algebraic counterparts of the fluxes at the interfaces) the rule \( r^{k+1}_{\Gamma_2}(\mu) = -\mathbb{R}_{12}r^{k+1}_{\Gamma_1}(\mu) \) would lead to a suboptimal method (see, e.g., [11][13]). To recover the optimal convergence rate, INTERNODES employs
A reduced order model for domain decompositions with non–conforming interfaces

Instead of \( r_{k+1}^{k+1}(\mu) = -R_{12}r_{k+1}^{k+1}(\mu) \). Further details on the derivation of these systems, which are indeed quite standard in the DD literature, can be found, e.g., in [31,61].

The residual vector is the algebraic counterpart of an element of the dual space \( Y_h' \) of \( Y_h \) – see, e.g., [60, Chapter 3] – then we define by \( z_{\Gamma}(\mu) \) the element obtained from \( r_{\Gamma}(\mu) \) by solving

\[
M_{\Gamma}z_{\Gamma}(\mu) = r_{\Gamma}(\mu).
\]

(12)

Here \( z_{\Gamma}(\mu) \) is the algebraic counterpart of the Riesz’ element associated with the residual \( r_{\Gamma}(\mu) \). In other words, the interface mass matrix becomes the transfer matrix from the Lagrange basis to the dual one and vice–versa [16,31] and the array \( z_{\Gamma}(\mu) \) represents the residual in primal form.

Remark 1 When \( \partial \Gamma_i \cap \partial \Omega_{i,D} \neq \emptyset \), the residual \( r_{\Gamma} \) should be corrected to take into account the interpolation process on all the degrees of freedom of \( \Gamma_i \), including those on \( \partial \Gamma_i \) (see e.g. [31]). Even if the reduced technique presented in this paper will work in both cases, hereon we will consider only \( \partial \Gamma_i \cap \partial \Omega_{i,D} = \emptyset \).

Remark 2 The above formulation can be easily extended to time–dependent second–order parabolic PDE problems. In such cases, suitable numerical schemes have to be implemented to handle the time discretization, and Dirichlet–Neumann subdomains iterations must be applied for each time step of the approximated solution [61]. The application of our method to a time–dependent test case will be addressed in Section 5.

Remark 3 The master and the slave solution snapshots can be directly collected from the FOM computations by solving (9)–(11) when (i) conforming discretizations are considered in the two subdomains or (ii) when interpolation/projection methods are implemented to handle non–conforming grids, e.g. MORTAR methods or INTERNODES. However, since their implementation is not trivial, both MORTAR and INTERNODES methods are not easy to find in standard scientific libraries. Moreover, RB methods rely on snapshot data, whose generation can be performed with any desired numerical methods. For the sake of generality, here we assume to not have either INTERNODES or MORTAR method implemented and therefore compute the snapshots in an alternative – perhaps naive – way by solving for each parameter instance the FOM problem twice, i.e. with two different conforming discretizations. In particular, we define two possible FE grids in the global domain by setting on each first subdomain a chosen spatial discretization and extending it (conformingly) to the corresponding second one. Now, both coupled problems feature interface conformity and can be solved with Dirichlet–Neumann iterations. Then, we collect the snapshots of each sub–problem and the relative interface data in the discretization set in the non–conforming case for the corresponding subdomain, obtaining two sets of solution and their Dirichlet and Neumann data. Indeed, these snapshots can be seen as the model solution when interface non–conformity is considered (see Fig. 1 for a schematic sketch of the used procedure). In this paper, such a technique is used to collect the snapshots for the test cases reported in Section 5.

3 Master and slave reduced order problems

The strategy we propose here aims both to reduce the two sub–problems separately and to employ reduced techniques to also address the Dirichlet and Neumann interface conditions arising from the Dirichlet–Neumann...
subdomains iterations. In particular, this ROM technique is an extension of the one proposed in [67], and combines different RB methods, one set for each sub–problem, and the DEIM to treat both Dirichlet and Neumann interface conditions, thus defining independent reduced order representations of the involved quantities.

We first approximate the FOM solution of the master and slave models by means of a POD–derived small number of basis functions defined in the corresponding subdomain \( \Omega_i \). Moreover, employing the DEIM, we identify a suitable set of basis functions for the master and slave interface snapshots, and we use them to transfer Dirichlet and Neumann data across conforming or non–conforming interface grids. Lastly, considering the same Dirichlet–Neumann iteration scheme of the high–fidelity FOM, we iterate between the reduced solutions of the two sub–problems by imposing the continuity of both the interface solutions and fluxes at each iteration (see Fig. 2).

The reduced forms of the master and the slave problems are described in this section, while we derive the procedure to reduce parameter–dependent Dirichlet and Neumann interface conditions in Section 4.

We define the reduced–order version of problems (9) and (10) relying on a POD–Galerkin approach [60]. Therefore, in the offline stage, we collect the set of snapshots solving the sub–FOMs for a suitable set of parameter values. In particular, we choose as snapshots the FOM slave and master solutions at convergence of the sub–iterations, i.e.

\[
S_1 = \{ u_1(\mu^k), \mu^k \in P_{\text{train}}^d \}
\]

and

\[
S_2 = \{ u_2(\mu^k), \mu^k \in P_{\text{train}}^d \},
\]

respectively. A sampling of the parameter space is here done considering a Latin hypercube sampling (LHS) method [41,52].

Remark 4 From now on, the index \( k \) is omitted when quantities at the convergence of the Dirichlet–Neumann FOM iterations are considered.

The POD technique is applied to each set of snapshots \( S_1 \) and \( S_2 \) and a corresponding set of reduced basis functions is computed and stored for the approximation of the solution on each subdomain. We denote by \( n_i \) the cardinality of the set of reduced basis functions. Defining \( V_i \in \mathbb{R}^{N_i \times n_i}, n_i \ll N_i \), for \( i = 1, 2 \), the matrices whose columns yield the obtained basis functions, the ROM seeks an approximation of the FOM solutions under the form

\[
u_1(\mu) \approx V_1 u_{n,1}(\mu)
\]

and

\[
u_2(\mu) \approx V_2 u_{n,2}(\mu).
\]
Fig. 2: Schematic representation of the reduced order Dirichlet–Neumann domain decomposition algorithm.

Projecting problems (9) and (10) onto the reduced spaces defined by $V_i$, starting from an initial guess $u_{0,i}^n(\mu)$ and $u_{0,2}^n(\mu)$, in the online phase, for each $k \geq 0$, we search for the reduced solutions $u_{n,1}^{k+1}(\mu) \in \mathbb{R}^{n_1}$ and $u_{n,2}^{k+1}(\mu) \in \mathbb{R}^{n_2}$ such that

$$
\begin{align*}
A_{n,1}(\mu)u_{n,1}^{k+1}(\mu) &= f_{n,1}(\mu) - V_i^T A_{1,D}(\mu)g_{D,1}(\mu) - V_i^T A_{1,\Gamma_1}(\mu)u_{\Gamma_1}^{k+1}(\mu) \\
u_{\Gamma_1}^{k+1}(\mu) &= R_{12}u_{\Gamma_2}^{k+1}(\mu)
\end{align*}
$$

and

$$
\begin{align*}
A_{n,2}(\mu)u_{n,2}^{k+1}(\mu) &= f_{n,2}(\mu) + V_i^T E_2r_{\Gamma_2}^{k+1}(\mu) - V_i^T A_{2,D}(\mu)g_{D,2}(\mu) \\
r_{\Gamma_2}^{k+1}(\mu) &= -M_{\Gamma_2}^{-1}M_{\Gamma_1}^{-1}r_{\Gamma_1}^{k+1}(\mu),
\end{align*}
$$

where

$$
A_{n,i}(\mu) = V_i^TA_{i,\cdot}(\mu)V_i, \quad f_{n,i}(\mu) = V_i^Tf_i(\mu), \quad i = 1, 2.
$$

Notice that the second equations in problems (15) and (16) (the interface equations) are defined in the FOM space, whereas the first ones are in the ROM space. The reduced version of the interface equations will be derived in the following section.
Remark 5 For simplicity, in this paper, we only consider the case of linear PDE problems. In the case of non-linear problems, the presence of non-linear terms in the master and slave formulations can be handled through suitable hyper-reduction techniques like, e.g., the DEIM \cite{10,19,18}. The ROM approach would be still modular, requiring an ad-hoc reduction of each sub-problem, whereas the processing of interface data would not be affected by the operator non-linearity, and would be treated with the approach shown in the next section.

Remark 6 Time-dependent problems can be reduced with RB methods considering the time variable as an additional parameter of the model. Indeed, in such a case the FOM solutions at each time step of the simulation are collected in the snapshots set, and the reduced basis allows to approximate the time-dependent solution of each sub-problem involving vectors of (reduced) degrees of freedom $u_{n,1}$, $u_{n,2}$ that are also time-dependent (see Section 5).

4 Parametric interface data reduction

Dealing with interface conditions, especially when using non-conforming grids, requires special care. Since the sub-problems (15) and (16) are parameter-dependent, the interface data naturally inherit the parameter dependency and the DEIM \cite{10,19,18,26,33,47,53} can be applied to both reduce the dimension of such data and to transfer the information across the interface grids.

Indeed, using DEIM requires: (i) to compute a set of basis functions for the quantity of interest employing POD, (ii) to use a greedy algorithm to identify a small number of DoFs to compute the weights for the corresponding basis functions (instead of the weights used in a simple POD). The nodes corresponding to such DoFs define the so-called reduced mesh, effectively determining a relation between the FE grids and the reduced space. Therefore, when considering interface data reduction, a small number of interface nodes can be selected through the DEIM to describe the complete vector of parametrized interface data. In the conforming case, DEIM can be used directly on the quantity of interest, i.e., the interface solution in the case of Dirichlet data, or the interface residual in the case of Neumann data. For non-conforming interface grids, the primal form of the residuals must be employed to properly treat the Neumann terms \cite{55}.

Even if using the DEIM at the interface between the two subdomains could seem a rather involved approach compared to more classic interpolation algorithms – like, e.g., piecewise-constant interpolation – DEIM offers two main advantages. First, in this context, DEIM also aims at approximating interface data that could in principle depend on parameters unrelated to the problem solution. Therefore, DEIM adjusts to variations in the interface data given to changes in the interface parametrization, independently of the model solution. Second, DEIM can be applied to larger domains with a substantially higher number of DoFs at the interface, compared to the one employed in our test cases, only involving a small subset of the interface DoFs, thus significantly accelerating the overall interpolation process.

In Subsection 4.1 and 4.2, we define the reduction of the Dirichlet and Neumann interface conditions, respectively, when non-conforming interface grids are considered. These new ROM interface conditions are then used to replace the interface equations of problems (15) and (16).

4.1 Parameter-dependent Dirichlet data

The parametric interpolation method of the Dirichlet data used in this work is similar to the one introduced in \cite{67}. Such a technique relies on the DEIM and can be applied in the case of both conforming and non-conforming interface grids.
First, in the offline phase, we collect from the slave domain $\Omega_1$ the interface snapshots, i.e., we extract the interface (Dirichlet) degrees of freedom obtained for different instances of the parameter vector from the FOM computation. Notice that, as for the solution reduction, we only select the interface DoFs at the convergence of the FOM Dirichlet–Neumann iterations, namely

$$S_D = \{ u_{I_1}(\mu), \, \mu \in \mathcal{P}_{\text{train}} \}.$$ 

Let us denote by $N_{1,A}$ the number of FOM DoFs on $\Gamma_1$. A low-dimensional representation of the interface DoFs can then be computed by determining a set of $M_1 \ll N_{1,A}$ POD basis functions from $S_D$ that we store in the matrix $\Phi_D \in \mathbb{R}^{N_{1,A} \times M_1}$, with the purpose of getting $u_{\Gamma_1}(\mu) \approx \Phi_D u_{1,M_1}(\mu)$,

where $u_{1,M_1}(\mu)$ is a vector of $M_1$ coefficients. Furthermore, with a greedy algorithm \[47\], we select iteratively $M_1$ indices in $\{1, \ldots, N_{1,A}\}$, by minimizing the interpolation error over the interface snapshots set $S_D$, according to the maximum norm. The set of such indices is denoted by

$$I_{1,D} \subset \{1, \ldots, N_{1,A}\}, \quad \text{with cardinality } |I_{1,D}| = M_1. \quad (17)$$

The points corresponding to the indices of $I_{1,D}$ are usually referred to as magic points on $\Gamma_1$, and are used to impose the Dirichlet interface conditions on $\Gamma_1$ for the reduced online problem. Let us denote by $u_{\Gamma_1,I_{1,D}}(\mu)$ the vector of the FOM DoFs on $\Gamma_1$.

In the online phase, at each Dirichlet–Neumann iteration $k$, we ask that the reduced interface vector $u_{1,M_1}^{k+1}(\mu)$ satisfies the relation

$$\Phi_{D_{I_{1,D}}} u_{1,M_1}^{k+1}(\mu) = u_{I_{1,D}}^{k+1}(\mu),$$

where $\Phi_{D_{I_{1,D}}} \in \mathbb{R}^{M_1 \times M_1}$ is the sub–matrix of $\Phi_D$ containing the $I_{1,D}$ rows (see e.g. \[18\] for the well–posedness of the above procedure). The FOM interface DoFs on $\Gamma_1$ can be then approximated as

$$u_{\Gamma_1}^{k+1}(\mu) \approx \Phi_D \Phi_{D_{I_{1,D}}}^{-1} u_{I_{1,D}}^{k+1}(\mu). \quad (18)$$

Now we replace $u_{I_{1,D}}^{k+1}(\mu)$ with the values of the master solution $u^k_{N,2}$ at the points on $\Gamma_2$ corresponding to the magic points on $\Gamma_1$. Thus, given the position $p_1$ of the magic point corresponding to the index $i_{1,D} \in I_{1,D}$ in Cartesian coordinates, we search for the corresponding node in the master interface $\Gamma_2$, i.e. for the point $p_2 \in \Gamma_2$ such that

$$p_2 = \arg \min_{p_j \in \text{nodes}_{\Gamma_2}} (\text{dist}(p_1 - p_j)),$$

where dist represents the Euclidean distance.

**Remark 7** When the solution of the above minimization problem is not unique, we choose as $p_2$ the last node found by the algorithm that satisfies such a relation.

Then, we can define the set $I_{2,D}$ of the indices on the master grid corresponding to the indices in $I_{1,D}$, i.e.

$$I_{2,D} = \{ i_{2,D}(i_{1,D}) \}_{i_{1,D} \in I_{1,D}}.$$

Notice that $I_{2,D}$ is computed in the offline phase.
Finally, in the online phase, we replace the FOM interface values at the magic points on $\Gamma_1$ with the values of the master solution at the corresponding points on $\Gamma_2$, i.e.,

$$
\left( u_{\Gamma_1}^{k+1}(\mu) \right)_{i_1,D} = \left( u_{\Gamma_2}^k(\mu) \right)_{i_2,D(i_1,D)}; \quad i_1,D \in I_{1,D}.
$$

(19)

More briefly, we write $u_{\Gamma_1}^{k+1}(\mu) = u_{\Gamma_2}^k(\mu)$, so that (18) becomes

$$
u_{\Gamma_1}^{k+1}(\mu) \approx \Phi_D \Phi_D^{-1} u_{\Gamma_2}^k(\mu).
$$

(19)

Remark 8 The substitution (19) can be interpreted as a low–order interpolation process: first, we build the piecewise constant function $\tilde{u}_{\Gamma_2}^k(\mu)$ (the orange one in Fig. 3) that interpolates the values of $u_{\Gamma_2}^k(\mu)$ at the magic points on $\Gamma_2$ (the blue dots); then the values $\tilde{u}_{\Gamma_1}^{k+1}(\mu)$ at the corresponding points of $\Gamma_1$ (the red symbols) are obtained by evaluating the function $\tilde{u}_{\Gamma_2}^k(\mu)$ at such points.

Fig. 3: Geometrical interpretation of the interface reduction with the DEIM. The blue points represent the magic points on $\Gamma_2$ and the values of $u_{\Gamma_2}^k(\mu)$ at these magic points, the orange lines represent the piecewise constant interpolating function $\tilde{u}_{\Gamma_2}^k(\mu)$, while the red crosses are the points on $\Gamma_1$ corresponding to the magic points of $\Gamma_2$ and the values of $u_{\Gamma_1}^{k+1}$ at these points.

Note that $u_{\Gamma_2}^k(\mu)$ refers to the approximation of the FOM solution of the master problem that must, therefore, be computed from the ROM solution $u_{n,2}^k(\mu)$ during the online phase. However, only a part of the approximated FOM master solution is needed, i.e., the one at the magic points on $\Gamma_2$. Therefore, one can store the FOM solution at the interface only at the magic points by pre–multiplying the ROM master solution by those rows of $V_2$ corresponding to the magic points. To this end, we introduce a restriction matrix $U_{12} \in \mathbb{R}^{M_1 \times N_2}$ whose only entries different from zero (and equal to one) are those whose column index belongs to $I_{2,D}$, so that

$$
\left( u_{\Gamma_2}^k(\mu) \right)_{i_2,D} = U_{12} \mathbf{v}_{n,2}^k(\mu).
$$

Therefore, the action of the operator $R_{12}$ in (15) (in both conforming and non–conforming cases), can be summarized by

$$
R_{12} u_{\Gamma_2}^k(\mu) = \Phi_{D} \Phi_{D}^{-1} u_{\Gamma_2}^k(\mu)
$$

and the interface condition in (15) becomes

$$
u_{\Gamma_1}^{k+1}(\mu) = \Phi_{D} \Phi_{D}^{-1} u_{\Gamma_2}^k(\mu).
$$
In this way, the last term of (15) can be approximated as

$$V_T^1 A_{1,1}(\mu) u_{I_1}^{k+1}(\mu) = V_T^1 A_{1,1}(\mu) \|_{\mathcal{R}_{12}} u_{I_2}^{k+1}(\mu) = V_T^1 A_{1,1}(\mu) \Phi_D \Phi_D^{-1}_{I_2,I_1} U_{12} V_{2n,2}(\mu),$$

where the matrix $\Phi_D \Phi_D^{-1}_{I_2,I_1} U_{12} V_{2}$ does not depend on the solution and can be pre-computed and stored in offline phase. Note that if $A$ is parameter independent or depends affinely on $\mu$, in the online phase we can pre-compute and store the entire product $V_T^1 A_{1,1}(\mu) \Phi_D \Phi_D^{-1}_{I_2,I_1} U_{12} V_{2}$.

Note that even if the interface snapshots stored in $\Phi_D$ contain the DoFs on $\Gamma_2$ at the convergence of the Dirichlet–Neumann iterations, the reduced coupled model is solved by iterating between the reduced master and slave models. For this reason, the index $k$ appears on the quantities computed online.

**Remark 9** As for the FOM computation, an initial guess of the Dirichlet interface conditions must be considered, but only at those points on $I_2$ corresponding to the magic points on $I_1$. Therefore, for $k=0$, the approximated FOM solution $U_{12} V_{2} u_{n,2}^0$ can be replaced with the FOM initial guess $u_0^{0,2}$.\[\text{Remark 10} \] Note that if the coupled problem is unsteady, to take into account the time variations of the solution, the interface data at the convergence of the Dirichlet–Neumann iterations for each time instant $n=1,\ldots,N_t$ must be collected in the set of snapshots. Once the POD basis has been computed on this set of snapshots, the interpolation of the Dirichlet data can be performed as in the steady case.

### 4.2 Parameter–dependent Neumann data

The DEIM used to interpolate the parametric Dirichlet interface conditions can be also applied to the parametric Neumann interface conditions; as before, let us detail the case of a steady problem. If the interface grids are conforming, for each $k \geq 0$,\[r_{I_2}^{k+1}(\mu) = -r_{I_1}^{k+1}(\mu)\]

so that the DEIM can be used on the interface residual. Instead, in the non–conforming case, the interface mass matrices are involved (see Section 2). However, recalling definition (12) of $z_{I_2}(\mu)$, i.e. the algebraic counterpart of the Riesz’ element associated with the residual, the second equation of (16) can be replaced by

$$z_{I_2}^{k+1}(\mu) = -R_{21} z_{I_1}^{k+1}(\mu).$$

Therefore, the vector $z_{I_2}^{k+1}$ is the quantity to be reduced and reconstructed using the DEIM.

For the sake of generality, in this subsection we derive the Neumann data approximation for the non–conforming case, however, the same procedure also holds for the conforming case. Starting from (11)–(12), for each $\mu$ we compute the interface residual snapshots,

$$S_N = \{ z_{I_2}^\ell(\mu), \mu \in \mathcal{P}_{\text{train}} \};$$

we remind that these snapshots are saved at the convergence of the FOM Dirichlet–Neumann iterations.

Applying POD, a set of $M_2$ basis functions is selected and stored in $\Phi_N \in \mathbb{R}^{N_2,2 \times M_2}$, being $M_2 \ll N_2,2$ (recall that $N_2,2$ is the number of mesh points on $I_2$). Given a generic $\mu$, the vector $z_{I_2}$ is, therefore, approximated as

$$z_{I_2}(\mu) = \Phi_N z_{2,M_2}(\mu),$$
where $z_{2,M_2}$ is a vector of $M_2$ coefficients. Then, $M_2$ magic points on $I_2$ are selected through a greedy algorithm and their indices in the master grid numbering are collected in the set

$$\mathcal{I}_{2,N} \subset \{1, \ldots, N_{2,A}\}, \quad \text{with cardinality } |\mathcal{I}_{2,N}| = M_2.$$ 

In the online phase, at each Dirichlet–Neumann iteration $k$, we need to find $z_{2,M_2}^{k+1}(\mu)$ such that

$$\Phi_{N|z_{2,N}} z_{2,M_2}^{k+1}(\mu) = \Phi_{T_2^k|z_{2,N}}^{k+1}(\mu),$$

where $\Phi_{N|z_{2,N}} \in \mathbb{R}^{M_2 \times M_2}$ is the restriction of $\Phi_N$ to the indices associated with the magic points identified by $\mathcal{I}_{2,N}$, while $\Phi_{T_2^k|z_{2,N}}^{k+1}(\mu)$ is the restriction of $z_{T_2^k}^{k+1}(\mu)$ at such magic points. Then, we can approximate

$$z_{T_2^k}^{k+1}(\mu) \approx \Phi_N \Phi_{N|z_{2,N}}^{-1} \Phi_{T_2^k|z_{2,N}}^{k+1}(\mu).$$

Now, $\Phi_{T_2^k|z_{2,N}}^{k+1}(\mu)$ is replaced by the values of $z_{T_2^k}^{k+1}(\mu)$ extracted at the points on $I_1$ corresponding to the magic points on $I_2$. As done in the previous subsection, to find such points on $I_1$, we first need to select the set of indices $\mathcal{I}_{1,N}$ on the slave interface $I_1$ corresponding to the magic points identified by $\mathcal{I}_{2,N}$. Therefore, denoting by $p_2$ the node in cartesian coordinates corresponding to the index $i_{2,N}$ in $\mathcal{I}_{2,N}$, we search for

$$p_1 = \arg \min_{p_1 \in \text{nodes}_{I_1}} (\text{dist}(p_2 - p_1)),
$$

where $\text{dist}(\cdot)$ denotes the Euclidean distance. Then, denoting by $i_{1,N}(i_{2,N})$ the index of $p_1$ in the numbering of the slave grid, we define the set of indices in the slave grid corresponding to the nodes defined by the indices in $\mathcal{I}_{2,N}$, i.e.

$$\mathcal{I}_{1,N} = \{i_{1,N}(i_{2,N})\}_{i_{2,N} \in \mathcal{I}_{2,N}}.$$

Thus, in the online phase, we impose that

$$(z_{T_2^k}^{k+1}(\mu))_{i_{2,N}} = (z_{T_1^k}^{k+1}(\mu))_{i_{1,N}(i_{2,N})}, \quad \text{with } i_{2,N} \in \mathcal{I}_{2,N}$$

or more briefly $z_{T_2^k|z_{2,N}}^{k+1}(\mu) = z_{T_1^k|z_{1,N}}^{k+1}(\mu)$, so that

$$z_{T_2^k}^{k+1}(\mu) \approx \mathbb{R}_{21} z_{T_1^k}^{k+1}(\mu) \approx \Phi_N \Phi_{N|z_{2,N}}^{-1} \Phi_{T_2^k|z_{2,N}}^{k+1}(\mu)$$

and, similarly to the Dirichlet interpolation, the operator $\mathbb{R}_{21}$ here is represented by

$$\mathbb{R}_{21} z_{T_2^k}^{k+1}(\mu) = \Phi_N \Phi_{N|z_{2,N}}^{-1} z_{T_2^k|z_{2,N}}^{k+1}(\mu).$$

Finally, recalling formula \cite{12}, we can recover the (Neumann) interface condition in the second equation of \cite{16} as

$$\Phi_{T_2^k}^{k+1}(\mu) \approx -M_{T_2^k} \Phi_N \Phi_{N|z_{2,N}}^{-1} U_{21} M_{T_1^k}^{-1} U_{21}^T \Phi_{T_2^k|z_{2,N}}^{k+1}(\mu),$$

with $U_{21} \in \mathbb{R}^{M_2 \times N_{1,A}}$ the restriction matrix whose entries different from zero and equal to 1 are only those with column index in $\mathcal{I}_{1,D}$. Note that the matrix $\Phi_{T_2^k}^{k+1}(\mu)$ does not depend on $\mu$ and can be therefore computed and stored offline.
Moreover, recalling the definition (11) of the interface residual, we can recover the dependency of such vectors on the reduced slave solution, computing \( r_{Γ_1;ζ_1,N}^{k+1} \) as the vector of dimension \( M_1 \) whose only non-zero entries are those at the magic points, i.e.,

\[
\begin{bmatrix}
\mathbf{A}_1(μ) & \mathbf{V}_1 \mathbf{u}_{n+1,1}(μ) & \mathbf{g}_{D,1}(μ)
\end{bmatrix} - f_{N,1}(μ)
\]  \quad . \tag{23}
\]

We notice that the magic points on \( Γ_1 \) found during the Dirichlet phase do not necessarily coincide with the corresponding points on \( Γ_2 \) found during the Neumann phase, i.e, \( I_{1,D} \) and \( I_{1,N} \) can differ, as well as \( I_{2,D} \) and \( I_{2,N} \) can do.

We summarize the coupled problem training procedure in Algorithms 1–2 including the interface DEIM reduction of both Dirichlet and Neumann processing, while the complete reduction of the two–way coupled model can be found in Algorithms 3–4 concerning the offline training and the online query of the ROM.

We remark that when the interface grids are conforming, a perfect matching between the corresponding nodes on the master and slave interface is found. However, this does not happen in the non–conforming case, where interpolation errors arise for both the Dirichlet and Neumann interface approximations, especially when the grids differ substantially. Considering the numerical tests of Section 5, to minimize such errors a possible remedy is to consider a finer discretization on the master domain than in the slave one since the Dirichlet approximation seems to suffer more from the interface difference than the Neumann one. Another remedy consists in employing more accurate interpolation operators, like Lagrange interpolation when the interfaces are geometrically conforming, or Radial Basis functions interpolation in the presence of geometrical non–conforming interfaces; this will be the subject of future work.

Moreover, given the smaller number of DoFs in the slave interface than in the master one – considering a coarser discretization in the slave domains, as done above – we check the continuity of the interface solution using the \( ℓ_2 \) norm of the difference between the approximated solutions (expressed in a high–fidelity format) on the DoFs of the slave interface, i.e.

\[
\| \mathbf{u}_{Γ_1}^{k+1}(μ) - \mathbf{R}_{12} \mathbf{u}_{Γ_2}^{k+1}(μ) \|_{ℓ_2} < ϵ,
\]  \quad \tag{24}

where \( \mathbf{R}_{12} \mathbf{u}_{Γ_2}^{k+1}(μ) = \mathbf{Φ}_{D,D} \mathbf{Φ}_{D,Γ_1}^{-1} \mathbf{U}_{12} \mathbf{V}_2 \mathbf{u}_{Γ_2,2}^{k+1}(μ) \) effectively represents the interpolation of the master solution on the slave interface grids by means of the DEIM (see Subsection 4.1).

**Remark 11** If the coupled problem is unsteady, similarly to the case of parameter–dependent Dirichlet data, to take into account the time variations of the solution, the interface data at the convergence of the sub–iterations for each time instant \( n = 1, \ldots, N_t \) must be collected in the set of snapshots. Once the POD basis has been computed on this set of snapshots, the interpolation of the Neumann data can be performed as in the steady case.

**Remark 12** The nearest neighbor interpolation adopted in exchanging the information across the interfaces in formulas (20) and (22) could be replaced by more sophisticated interpolation techniques, such as, e.g., that based on Rescaled Localized Radial Basis Function (RBF). In such a case, if we denote by \( P_{12} \) and \( P_{21} \) the RBF interpolation matrices (see [9][24]) of size \( (M_1 \times N_{Γ_1}) \) and \( (M_2 \times N_{Γ_2}) \), respectively, and define a matrix \( \mathbf{U}_{Γ_2,2} \) of size \( (N_{Γ_2} \times N_2) \), used to extract the solution of the master model on \( Γ_2 \), then equation (20) for the interpolation of the Dirichlet data on the interfaces can be reformulated as:

\[
\mathbf{V}_1^T \mathbf{A}_{1,Γ_1}(μ) \mathbf{u}_{Γ_1}^{k+1}(μ) = \mathbf{V}_1^T \mathbf{A}_{1,Γ_1}(μ) \mathbf{Φ}_{D,D} \mathbf{Φ}_{D,Γ_1}^{-1} \mathbf{P}_{12} \mathbf{U}_{Γ_2,2} \mathbf{V}_2 \mathbf{u}_{Γ_2,2}^{k}(μ).
\]
Similarly, for the interpolation of the residuals, equation (22) can be reformulated as:
\[ r^{k+1}_{I_2}(\mu) \approx -M_I \Phi_N \Phi^{-1}_{D_1 I_2, N} P_{21} M^{-1}_{I_1} U_{21} r^{k+1}_{I_1, N}(\mu). \]
Notice that it is sufficient to compute the matrix products appearing in the right–hand sides of the above formulas only once during the offline phase. At least for the coupled problems exploited in Section 5, we experienced that, keeping fixed all the other approximations (such as the FOM finite element spaces and the POD/DEIM techniques), the interpolation approach across the interfaces affects very mildly the convergence rate of the ROM Dirichlet–Neumann method, as well as the error of the ROM solution with respect to the FOM one. In Appendix B we report some numerical results obtained with RBF interpolation that corroborate this remark.

**Algorithm 1** ROM training procedure – Snapshots computation

1: procedure [SNAPSHOTS arrays] = Snapshots(FOM arrays, \(\mathcal{P}_{\text{train}}^d\), tol)
2: Dirichlet and Neumann data snapshots
3: Given the set of \(\mathcal{P}_{\text{train}}^d \subset \mathcal{P}^d\) parameters:
4: for \(\mu \in \mathcal{P}_{\text{train}}^d\) do
5: \(u_1, u_2 \leftarrow\) solve the coupled problem (9)–(11) with Dirichlet–Neumann iterations, the convergence is achieved when \(\|u^{k+1}_I - R_{12} u^{k+1}_{I_2}\|_2 < \text{tol}\)
6: \(u_{I_1} \leftarrow\) extract the slave interface solution;
7: \(z_{I_2} \leftarrow\) extract the master primal residual.
8: \(S_1 = [S_1, u_1];\)
9: \(S_2 = [S_2, u_2];\)
10: \(S_D = [S_D, u_{I_1}];\)
11: \(S_N = [S_N, z_{I_2}];\)
12: end for
13: end procedure

5 Numerical results

In this section, we present numerical results obtained solving (i) a steady problem, namely a Dirichlet boundary value problem for a linear diffusion–reaction equation, and (ii) a time–dependent problem, namely an initial–boundary value problem for the heat equation with Neumann boundary conditions. In particular, we aim to investigate the performances of the proposed algorithm by comparing FOM and ROM results in terms of both efficiency and accuracy.

The mathematical models and numerical methods presented in this section have been implemented in C++ and Python languages and based on lifeX [https://lifex.gitlab.io][1], an in–house high–performance C++ FE library mainly focused on cardiac applications based on deal.II FE core [3] [https://www.dealii.org]. Both online and offline stages of the simulations have been performed in serial on a notebook with an Intel Core i7–10710U processor and 16GB of RAM.

In what follows, spatial domains are discretized employing Q1-FEM. We would like to point out that the geometric error induced by the approximation of curve surfaces by planes is of order \(h^2\) and is totally
Algorithm 2 Interface DEIM training procedure – ROM arrays

1: procedure [ROM arrays] = ROM arrays($S_D$, $S_N$, $\epsilon_{tol_D}$, $\epsilon_{tol_N}$)
2: DEIM reduced–order arrays:
3: $\Phi_D \leftarrow$ POD($S_D$, $\epsilon_{tol_D}$); $I_{1,D} \leftarrow$ DEIM–indices($\Phi_D$); 
4: $\Phi_N \leftarrow$ POD($S_N$, $\epsilon_{tol_N}$); $I_{2,N} \leftarrow$ DEIM–indices($\Phi_N$); 
5: Dirichlet magic points:
6: for $i_{1,D} \in I_{1,D}$ do
7: $p_1 \leftarrow$ get Cartesian coordinates of $i_{1,D}$ node;
8: $p_2 = \arg\min_{p_2 \in \text{nodes}_{\Gamma_2}} (\text{dist}(p_1 - p_2)) \leftarrow$ search in $\Gamma_2$ the nearest node to $p_1 \in \text{nodes}_{\Gamma_1}$;
9: $i_{2,D} \leftarrow$ get the Dirichlet index for $p_2$;
10: $I_{2,D} = [I_{2,D}, i_{2,D}]$;
11: end for
12: Neumann magic points:
13: for $i_{2,N} \in I_{2,N}$ do
14: $p_2 \leftarrow$ get Cartesian coordinates of $i_{2,N}$ node;
15: $p_1 = \arg\min_{p_1 \in \text{nodes}_{\Gamma_1}} (\text{dist}(p_2 - p_1)) \leftarrow$ search in $\Gamma_1$ the nearest node to $p_2 \in \text{nodes}_{\Gamma_2}$;
16: $i_{1,N} \leftarrow$ get the Neumann index for $p_1$;
17: $I_{1,N} = [I_{1,N}, i_{1,N}]$;
18: end for
19: end procedure

Algorithm 3 Complete ROM training procedure

1: procedure [ROM arrays] = Offline training(FOM arrays, $\mathcal{P}^d_{\text{train}}, \epsilon_{tol_1}, \epsilon_{tol_2}, \epsilon_{tol_D}, \epsilon_{tol_N}, \text{tol}$)
2: Solution, Dirichlet, and Neumann data snapshots
3: [$S_1$, $S_2$, $S_D$, $S_N$] = SNAPSHOTS(FOM arrays, $\mathcal{P}^d_{\text{train}}$, tol);
4: POD reduced–order arrays:
5: $\mathcal{V}_1 \leftarrow$ POD($S_1$, $\epsilon_{tol_1}$);
6: $\mathcal{V}_2 \leftarrow$ POD($S_2$, $\epsilon_{tol_2}$);
7: $\{A_{n,1}, f_{n,1}\} \leftarrow$ Galerkin projection of the FOM slave arrays onto $\mathcal{V}_1$;
8: $\{A_{n,2}, f_{n,2}\} \leftarrow$ Galerkin projection of the FOM master arrays onto $\mathcal{V}_2$;
9: DEIM reduced–order arrays:
10: $[\Phi_D, \Phi_N, I_{1,D}, I_{2,N}] = \text{ROM ARRAYS}(S_D, S_N, \epsilon_{tol_D}, \epsilon_{tol_N})$;
11: $\Phi_D^T \Phi_D^{-1}$ $U_1 \mathcal{V}_2 \leftarrow$ save matrix product for the slave trace term;
12: $\mathcal{V}_2^T E_2 M_2^T \Phi_N^T \Phi_N^{-1}$ $U_21 \mathcal{M}_1^{-1} \mathcal{U}_1^T$ $\mathcal{U}_21 \leftarrow$ save matrix product for the master residual term;
13: end procedure

consistent with the approximation properties of Q1-FEM. The gaps and overlaps on the interface are again of order $h^2$ and they do not downgrade the FOM approximation error. Moreover, the ROM procedure is totally independent of the FE element used, since the reduced basis is not related to spatial properties of the mesh.
Algorithm 4 ROM query

1: procedure $[\mathbf{u}_{N,1},\mathbf{u}_{N,2}] = \text{ONLINE QUERY}(\text{ROM arrays, FOM arrays, } \mathbf{\mu}, \text{tol})$
2: given the index $k \geq 0$ and the initial guess $\mathbf{u}_2^0$
3: while $\left(\|\mathbf{u}_{k+1}^{\Gamma_1} - \mathbf{R}_{1;2}^{k}\mathbf{u}_2^0(\mathbf{\mu})\|_2 > \text{tol}\right)$ do
4: $\mathbf{u}_{k+1;1;2,D}^{\Gamma_1}(\mathbf{\mu}) \leftarrow$ extract Dirichlet magic points values on $\Gamma_2$;
5: $\mathbf{u}_{k+1;1,D}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ assemble the trace term;
6: $\mathbf{u}_{k+1;1}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ solve the slave problem (15);
7: $\mathbf{u}_{k+1;2}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ solve the master problem (16);
8: $\mathbf{u}_{k+1}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ extract the slave interface residual on the magic points $\mathcal{I}_{1,N}$;
9: $-\mathbf{r}_{k+1;1;2,N}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ assemble the interface residual term;
10: $\mathbf{r}_{k+1;1;2,N}^{\Gamma_2}(\mathbf{\mu}) \leftarrow$ relax the master trace on $\Gamma_2$ (see e.g. (26));
11: $k \leftarrow k + 1$
12: end while
13: Recover $\mathbf{u}_{N,1}$ and $\mathbf{u}_{N,2}$, using [13] and [14], respectively.
14: end procedure

5.1 Test#1. Steady case: diffusion–reaction equation

In this first test case, we solve the following boundary value problem for a diffusion–reaction equation:

\[
\begin{align*}
-\nabla \cdot (\alpha \nabla u) + \beta u &= f \quad \text{in } \Omega \\
u &= g_D \quad \text{on } \partial \Omega_D
\end{align*}
\]  

(25)

where the domain $\Omega$ is a hollow spheroid (see Fig. 4) with inner and outer radius equal to $0.5m$ and $3.0m$, respectively, and parameters vector $\mathbf{\mu} = (\alpha, \beta)$. The interface $\Gamma$ is here represented by a sphere of radius $1.5m$. Moreover, $f(x,y,z) = \pi^2 4y^2 \sin(\pi y) \exp(z-1)$, while

\[
g_D = \begin{cases} 
0.01 & \text{on the internal sphere} \\
0 & \text{on the external sphere.}
\end{cases}
\]

We split the domain into two hollow spheroids with a common interface (see Fig. 4), where $\Omega_1$ is the inner and $\Omega_2$ is the outer spheroid, and apply the Dirichlet–Neumann iterative scheme so that, given $\lambda_0^2$, for each $k \geq 0$, we solve the two following sub–problems

\[
\begin{align*}
-\nabla \cdot (\alpha \nabla u_1^{k+1}) + \beta u_1^{k+1} &= f \quad \text{in } \Omega_1 \\
u_1^{k+1} &= 0.01 \quad \text{on } \partial \Omega_1,D \\
u_1^{k+1} &= \lambda_2^1 \quad \text{on } \Gamma_1
\end{align*}
\]

and

\[
\begin{align*}
-\nabla \cdot (\alpha \nabla u_2^{k+1}) + \beta u_2^{k+1} &= f \quad \text{in } \Omega_2 \\
u_2^{k+1} &= 0 \quad \text{on } \partial \Omega_2,D \\
\alpha \nabla u_2^{k+1} \cdot \mathbf{n}_2 &= \alpha \nabla u_1^{k+1} \cdot \mathbf{n}_2 \quad \text{on } \Gamma_2,
\end{align*}
\]
while

\[ \lambda_{k+1}^2 = \omega u_{2_1/2_2}^k + (1 - \omega)\lambda_k^2. \]  

(26)

The acceleration parameter \( \omega = 0.25 \) is fixed among the iterations, for both FOM and ROM problems, whereas we choose a tolerance \( \epsilon = 10^{-10} \) to check at each iteration \( k \) the continuity of the solutions at the subdomains interface, according to (24).

As FOM dimension, we consider \( N = 3474 \) in the slave domain \( \Omega_1 \) and \( N = 26146 \) in the master domain \( \Omega_2 \). We choose to vary the parameters \( \alpha \) and \( \beta \) in \([1, 10]\) using an LHS to compute the solution snapshots. We select \( N_{\text{train}} = 150 \) to get a sufficiently rich snapshot set to build the reduced bases, while additional \( N_{\text{test}} = 20 \) values of the parameter vectors are chosen to test the method.

As highlighted in Section 2, Remark 3, for each set of parameters we solve twice the high-fidelity FOM (once with a conforming mesh on \( \Omega \) designed starting from the mesh size in \( \Omega_1 \), and the other time with another conforming mesh on \( \Omega \) designed starting from the mesh size in \( \Omega_2 \)) and once the ROM. Computational relative errors are evaluated considering the difference between the FOM and ROM solutions in the \( H^1(\Omega_i) \) norm, normalized with the \( H^1(\Omega_i) \) norm of the FOM solutions, \( i.e. \)

\[ \frac{\|u_{\text{FOM}}(\mu) - u_{\text{ROM}}(\mu)\|_{H^1(\Omega_i)}}{\|u_{\text{FOM}}(\mu)\|_{H^1(\Omega_i)}}, \quad i = 1, 2. \]  

(27)

Remark 13 The FOM solutions are computed on conforming meshes, whereas the ROM solutions are computed on non-conforming ones. Consequently, FOM and ROM solutions inherently differ. This discrepancy, which should be related to the distinction between the FOM computed on conforming meshes (as conducted in this work) and non-conforming ones, should become apparent when performing a comprehensive convergence study of the RB method proposed here. However, this topic exceeds the scope of this work and will be the focus of future research endeavors.

Fig. 4 and 5 show three FOM and ROM solutions and the absolute error on the slave and the master domains, respectively, while in Fig. 7 we plot the \( H^1(\Omega_i) \)-norm relative errors for different dimensions of the ROM, \( i.e. \) for different values of the parameters \( n_1, n_2, M_1 \) and \( M_2 \) defined in Sections 3 and 4. After some tests, we chose to consider the same number of basis functions to get a fixed accuracy of the solution for the Dirichlet and Neumann data \( i.e. M_1 = M_2 \), while we treat independently the number of basis functions \( n_1 \)
\( \alpha = 2.35, \beta = 9.55 \quad \alpha = 6.85, \beta = 3.25 \quad \alpha = 4.15, \beta = 5.05 \)

Fig. 5: Test#1. Slave solution FOM (top), ROM (center) solutions, and absolute error (bottom) for three different vectors of testing parameters.

and \( n_2 \) used for the master and slave reduction. Then, Fig. 7 shows an increase, even if small, in the solution accuracy when the number of basis functions of one of the reduced quantities involved in the procedure is increased, as expected from RB theory. Major effect on the approximation accuracy of either master and slave solutions can be observed when \( M_1 \) and \( M_2 \) are varied, i.e. depending on the level of accuracy required on the interface data approximation (see also Fig. 9, left). A more complete analysis of the effect of varying the number of basis functions for each involved reduced quantity can be found in Appendix A.

Table 1 compares the number of DoFs and basis functions employed in each FOM and ROM simulation. On average, a FOM solution is found after 33 iterations for the coarser discretization and 32 iterations for the finer one, while a ROM solution is found after 42 iterations. Since, however, the number of iterations to reach convergence of the interface solutions depends on the model parameter instance, we investigate the overall iterations trends of the ROM model versus the FOM ones by computing the average ratio between the number of iterations required by the ROM, and the number of iterations required by the corresponding FOM, with either coarse and fine discretizations. Fig. 8 (top row) shows that increasing the number of basis functions to approximate the interface data, the number of iterations to reach the solution convergence decreases, whereas increasing the number of basis functions for the slave solutions, the number of iterations increases. The iterations ratio is overall not influenced by the master hyper-parameters. The same behavior can be also observed by plotting the iterations ratio against the approximation error in Fig. 9.

Remark 14 The computational expense of the ROM should be compared with that of a FOM utilizing non-conforming meshes in both domains. It is important to note that the FOM computations are performed on conforming meshes – with the finest discretization employed in the slave domain and the coarsest
discretization in the master domain of the ROM simulations, respectively. Moreover, the FOM calculations do not incorporate interface interpolation methods. As a result, comparing the computational times of FOM and ROM simulations would not yield a fair assessment. However, the appropriate FOM simulation for comparison with the ROM should possess a computational expense that falls between the two FOM cases under consideration.

The same behaviour can be observed in terms of computational costs (see the bottom row of Fig. 8), where we show that the speed up gained by employing the ROM instead of the FOM scheme increases when $M_1$ and $M_2$ increase, whereas it is overall the same independently of $n_1$ and $n_2$. Of the complete ROM simulation, about 20% of the computational cost is required to set up and assemble the parameter–independent matrices and vectors, whereas the remaining 80% is devoted to the ROM iterations. 7% of the costs of each ROM iteration depends instead on the extraction and application of Dirichlet interface data, 33% on extraction, computation, and application of the Neumann interface data, 32% on the assembling and solution of slave
Fig. 7: Test#1. $H^1(\Omega)$ mean relative error ($z$-axis) over the solution for $N_{\text{test}} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave and the master solution $n_1$ and $n_2$, and the interface data $M_1$ and $M_2$ ($x$- and $y$-axis). On the top row, we fix the number of basis functions of the master problem to 5 (on the left) and to 7 for the slave problem (on the right), while on the bottom we fix the number of basis functions equal to 6 for the interface data representation.

ROM, 25% on the assembling and solution of the master ROM, and 3% on the computation of the interface residual. Specifically, for an instance of the parameters, with a prescribed accuracy of the solution of $10^{-5}$, the computational costs of solving problem (25) with a coarse (fine) discretization of both subdomains is of about 3.39s (37.16s), whereas the ROM simulation requires an online time of about 1.55s, and a total time of 1h and 45min for training the model. The ROM is thus about 2.5 times faster than the FOM employing the coarser discretization, corresponding to a CPU time reduction of about 61%, while we are able to achieve a speedup of about 24 times compared to the FOM employing the finer discretization, corresponding to a reduction of 96% of the computational costs.
Fig. 8: Test#1. Top row: the ratio between the number of iterations obtained with ROM and FOM schemes versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). Bottom row: the ratio between the FOM and ROM computational time versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). The FOM simulation refers to the finer discretization.

|                  | Master solution | Slave solution | Master interface | Slave interface |
|------------------|-----------------|----------------|------------------|-----------------|
| FOM - coarse mesh| #DoFs 3474      | #DoFs 3474     | 386              | 386             |
| FOM - fine mesh  | #DoFs 26146     | #DoFs 26146    | 1538             | 1538            |
| ROM              | #Basis 7        | #Basis 5       | 6                | 6               |
|                  | #DoFs 26146     | #DoFs 3474     | 1538             | 386             |

Table 1: Test#1. High fidelity and reduced order model dimensions of subdomains and interface discretization, as well as the number of basis functions required to achieve an approximation error of $10^{-5}$. 
5.2 Test#2. Steady case: diffusion reaction equation with parametrized sources

Employing the same modeling framework of Subsection 5.1, we increase the level of complexity of the model by modifying the right-hand side function $f$ as

$$f(x,y,z) = \begin{cases} 
\gamma_1 \left( \sin \left( \frac{\pi}{2} x^2 \right) + xy \right) & \text{in } \Omega_1, \\
\gamma_2 e^{-\left( x-1 \right)^2 + \left( y-1 \right)^2 + \left( z-1 \right)^2} & \text{in } \Omega_2, 
\end{cases}$$

where $\gamma_1$ and $\gamma_2$ represent two varying parameters, defined in the interval $[0, 15]$. This $f$ thus represents two different physical sources in $\Omega_1$ and $\Omega_2$, that increase the interface solution mismatch, as well as the gradient of the interface solution, if compared to the previous test case. Model solutions for different instances of $\mu = [\alpha, \beta, \gamma_1, \gamma_2]$ on either slave or master subdomains can be found in Fig. 10 and 11.

Employing an LHS to randomly select the parameter values, we define $N_{\text{train}} = 250$ to train the ROM, and $N_{\text{test}} = 20$ to test the online procedure. Fig. 12 shows the variation of the approximation error as a function of the number of basis functions employed to compute one of the three reduced quantities involved. Fixing two of the reduced order hyper-parameters (between $n_1$, $n_2$, or $M_1 = M_2$), the decrease of the approximation error follows the growth of the third hyper-parameter involved, as expected from the ROM theory. However, differently from the test case in Subsection 5.1, this test shows a similar dependency on the accuracy of either the slave solution, master solution, or interface data, as can be observed from Fig. 14.

To solve the model with the coarse and the fine discretization in both $\Omega_1$ and $\Omega_2$, on average 34 and 33 iterations are required, respectively, whereas the ROM algorithm needs 45 iterations to reach interface convergence (see Table 1 for the number of DoFs employed in each subdomain). The iterations trend of the ROM vs the FOM solutions (see Fig. 13) indicates that, increasing $M_1$ and $M_2$, the number of iterations required by the ROM algorithm increases, which can be explained by the complex interface solution to be approximated. By increasing $n_1$, instead, the cost of the iterations decreases. The same behaviour can be observed also in Fig. 14 by comparing the approximation errors and the iterations ratio, as well as looking at the computational time as a function of the ROM hyper-parameters (see Fig. 13 bottom row) where, by increasing $M_1$ and $M_2$, the ROM speed up compared to the FOM decreases, while the overall time gained by employing the ROM is overall the same independently on $n_1$ and $n_2$. Imposing a tolerance of $10^{-5}$ to
 approximate all slave and master solutions, and interface data, about 18.1s are needed by the ROM algorithm to compute the solution, 26.8s for the FOM computation with the coarse discretization, and 225.33s for the FOM computation with the fine discretization. Therefore, the CPU time is reduced by about 34% compared to the coarse discretization, and 92% compared to the fine discretization.

5.3 Test#3. Unsteady case: time–dependent heat equation

We now apply the proposed technique to a time–dependent problem. In particular, we consider the following initial–boundary value problem for the heat equation

\[
\begin{align*}
\frac{\partial u}{\partial t} - \nabla \cdot (\alpha \nabla u) &= f & \text{in } \Omega \times (0, T) \\
\alpha \nabla u \cdot n &= 0 & \text{on } \partial \Omega \times (0, T) \\
u &= 0 & \text{in } \Omega \times \{t = 0\}
\end{align*}
\]

being \( \Omega = (-0.5, 1.5) \times (-0.5, 0.5) \times (-0.5, 0.5) \) (see Figure 15) and

\[
f = \begin{cases} 
1 & \text{if } x < 0 \land 0.2 < t < 0.5 \\
0 & \text{otherwise}. 
\end{cases}
\]

The problem is parametrized through the coefficient \( \alpha \) and – indirectly – through the time variable \( t \).
To apply the reduced technique presented in this work, we split $\Omega$ into two cubes with a common plane $x = 0.5$ that represents the interface $\Gamma$ (see Fig. 15). The application of the Dirichlet–Neumann iterative scheme leads therefore to the two following sub–problems: given $\lambda_2^k$ on $\Gamma_2$, for each $k \geq 0$, solve until convergence of the interface solution

\[
\begin{align*}
\frac{\partial u_1^{k+1}}{\partial t} - \nabla \cdot (\alpha \nabla u_1^{k+1}) &= f & \text{in } \Omega_1 \times (0, T) \\
\alpha \nabla u_1^{k+1} \cdot n_1 &= 0 & \text{on } \partial \Omega_1 \times (0, T) \\
u_1^{k+1} &= \lambda_2^k & \text{on } \Gamma_1 \times (0, T) \\
u_1^{k+1} &= 0 & \text{in } \Omega_1 \times \{t = 0\}
\end{align*}
\]
A reduced order model for domain decompositions with non–conforming interfaces

Fig. 12: Test#2. $H^1(\Omega_i)$ mean relative error (z–axis) over the solution for $N_{\text{test}} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave and the master solution $n_1$ and $n_2$, and the interface data $M_1$ and $M_2$ ($x$– and $y$–axis). On the top row, we fix the number of basis functions of the master problem to 8 (on the left) and to 10 for the slave problem (on the right), while on the bottom we fix to 8 the number of basis function for the Dirichlet and Neumann interface data representation.

and

$$\begin{align*}
\frac{\partial u_{k+1}^2}{\partial t} - \nabla \cdot (\alpha \nabla u_{k+1}^2) &= f \quad \text{in} \quad \Omega_2 \times (0, T) \\
\alpha \nabla u_{k+1}^2 \cdot n_2 &= 0 \quad \text{on} \quad \partial \Omega_{2,N} \times (0, T) \\
\alpha \nabla u_{k+1}^2 \cdot n_2 &= -\alpha \nabla u_{k+1}^2 \cdot n_1 \quad \text{on} \quad I_2 \times (0, T) \\
u_{k+1}^2 &= 0 \quad \text{in} \quad \Omega_2 \times \{t = 0\},
\end{align*}$$

while

$$\lambda_{k+1}^2 = \omega u_{2,1}^k + (1 - \omega)\lambda^k_2.$$ 

As for the steady test case, a fixed point acceleration strategy with parameter $\omega = 0.25$ is applied to accelerate the convergence of the Dirichlet–Neumann scheme for both FOM and ROM computations.

Two different discretizations are employed, corresponding to a FOM dimension equal to $N_1 = N_2 = 35937$ for the first discretization, and $N_1 = N_2 = 274625$ for the second discretization. The ROM is then solved with $N_2 = 274625$ DoFs for the slave domain and $N_1 = 35937$ DoFs for the master domain. We define $t \in [0, 1]$. 
Fig. 13: Test#2. Top row: the ratio between the number of iterations obtained with ROM and FOM schemes versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). Bottom row: the ratio between the FOM and ROM computational time versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). The FOM simulation referred to fine discretization.

and we use the first order backward Euler scheme with $\Delta t = 10^{-2}s$ to handle time discretization. Moreover, the diffusivity coefficient $\alpha$ is varied between 0.5 and 5, employing also, in this case, the LHS strategy to select the snapshots to be computed for the ROM training. In particular, we select $N_S = N_t N_{\text{train}}$ snapshots to train the ROM, corresponding to $N_{\text{train}}$ =10 complete simulations in time, being $N_t = 100$ the number of time steps in a model simulation. Then, $N_{\text{test}} = 2$ simulations of 100 time–steps each are considered to test the ROM. The tolerance to stop the Dirichlet–Neumann iteration is $10^{-10}$, which corresponds to 18 iterations to find the FOM solution with the coarser discretization, 19 iterations for the FOM solution with the finer discretizations, and 17 iterations to find the ROM solution for each time step.

We compute the $H^1(\Omega_i)$ relative error (27) between the ROM and the FOM solutions with the coarser mesh (see Table 2) for the slave problem, and between the ROM and the FOM solutions with the coarser mesh (see Table 2) for the master problem. In Figs. 16 and 17 we draw the ROM and FOM solutions and the absolute errors for three different time instants and some selected values of $\alpha$, while in Fig. 18 we plot the $H^1(\Omega_i)$ relative error for different ROM dimensions. Note that whenever we fix the number of basis functions used to approximate the master solution, we vary the number of basis functions for both the slave solution and the interface data. Similarly, when we fix the number of basis functions for either the slave solution or
Fig. 14: Test #2. Ratio between the number of iterations obtained with ROM and FOM schemes versus the achieved approximation error depending on the number of basis functions employed for the interface data (left) and the slave solution (right).

Fig. 15: Test #3. \( \Omega_1 \) (left) and \( \Omega_2 \) (center) and \( \Omega \) (right). In green the interface \( \Gamma \).

the Dirichlet–Neumann interface data, we vary the number of basis functions for the other sets. Both the slave and the master approximation errors show the expected behaviour: they decrease whenever we increase the employed number of basis functions. Different from the steady test cases (see Subsections 5.1 and 5.2), the approximations error is here equally dependent on the hyper–parameters \( n_1 \), \( n_2 \) and \( M_1 \) and \( M_2 \), as can also be observed in Fig. 20.

Finally, in Table 2 we compare the number of DoFs employed to solve the FOM and ROM. The time average iterations varying the ROM interface hyper–parameters or the \( n_1 \) can be found in Fig. 19 (top row). Such graphs show that increasing both \( M_1 \) (and \( M_2 \) and \( n_1 \)), the number of iterations required by the ROM to reach convergence of the interface solutions increases. A corresponding decrease in speed up can be seen in the bottom row of Fig. 19. When the approximation tolerance of \( 10^{-5} \) is selected for each reduced quantity with the coarsest (finest) discretization, the FOM solution requires 37\,min and 18\,s (7\,h and 52\,min) of computations, whereas only 7\,min and 45\,s are needed to solve the ROM. As before, our reduced technique shows high computational efficiency, effectively achieving a speedup of 5 times with respect to the FOM solution with the coarsest discretization and of about 60.9 times compared to the finest discretization, corresponding to a reduction of about 80% and 98% of the computational costs, respectively. These results refer to a complete simulation of 100 time–steps, including some repeated tasks such as the assembling of the right–hand side, according to the time discretization scheme implemented.
Fig. 16: Test#3. Slave solution FOM (top), ROM (center) solutions and absolute error (bottom) for $\alpha = 2.75$ and three different time instants.

Table 2: Test#3. High fidelity and reduced order model dimensions of subdomains and interface discretization, as well as the number of basis functions required to achieve an approximation error of $10^{-5}$.

|                | Master solution | Slave solution | Master interface | Slave interface |
|----------------|-----------------|----------------|------------------|-----------------|
| FOM – coarse mesh | #DoFs 35937     | #DoFs 35937    | 1089             | 1089            |
| FOM – fine mesh  | #DoFs 274625    | #DoFs 274625   | 4225             | 4225            |
| ROM             | #Basis 10       | #Basis 10      | 10               | 10              |
|                 | #DoFs 274625    | #DoFs 35937    | 4225             | 1089            |

6 Conclusion

In this work, we have introduced a reduced order modeling technique based on RB methods to decrease the computational costs entailed by the solution of two–way coupled problems employing Dirichlet–Neumann iterative schemes. The modularity of the procedure ensures the efficiency of the ROM through different treatments of the master, slave, and interface data reduction. Indeed, the master and the slave modes can be reduced using appropriate RB strategies, while the Dirichlet and the Neumann interface data can be handled through the DEIM inside the INTERNODES method, highlighting the possibility of using such reduced scheme to transfer the interface data between the conforming and non–conforming interface grids.
The proposed algorithm can also be applied, in principle, to more general multi-physics problems that are solved through Dirichlet–Neumann domain-decomposition iterative schemes.

The numerical test cases show that our ROM is very cheap in the online phase, outperforming the online CPU time of the FOM when either fine or coarse conforming meshes in the two domains are considered. Indeed, a decrease of 98% of the CPU time can be achieved in the case of a time-dependent parabolic PDE problem (see Subsection 5.3) with respect to a finite element FOM built over a fine discretization, and of 85% with respect to a finite element FOM built over a coarser one. An analysis of the iteration cost as a function of the reduced order hyper-parameters is also carried out.

This paper represents (to our knowledge) the first attempt toward the achievement of a fully-reduced RB interpolation-based numerical scheme for the two-way coupled model.

Future developments concern the use of the INTERNODES method on the FOM offline phase and the use of interpolation operators in the ROM online phase (see Remark 8) that are more accurate than those used in this paper. We are confident in this way to improve the accuracy as well as the efficiency (of the offline phase) of our approach. Finally, future work will be the derivation of an error estimate for the proposed method.

Considering the results proposed in the last section, we expect to be able to apply the presented method to more complex and relevant physics-based coupled problems, e.g. cardiac electrophysiological and fluid-structure interaction problems. Moreover, having been able to correctly reduce Dirichlet and Neumann interface data, the final extension of this work would be the application of our technique to Robin-like interface conditions, too. All these aspects will be the focus of future papers.
Fig. 18: Test#3. $H^1(\Omega_i)$ mean relative error ($z$–axis) over the solution for $N_iN_{test} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave and the master solution, and the interface data ($x$– and $y$–axis). On the top row, we fix the number of basis functions of the master problem to 10 (on the left) and to 10 for the slave problem (on the right), while on the bottom we fix 8 the number of basis functions for the interface data representation.

References

1. Africa, P., Piersanti, R., Fedele, M., Dede, L., Quarteroni, A.: lifex–heart module: a high-performance simulator for the cardiac function Package 1: Fiber generation. arXiv preprint arXiv:2201.03303 (2022)
2. Amsallem, D., Cortial, J., Farhat, C.: Toward real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information. AIAA Journal 48, 2029–2037 (2010)
3. Antil, H., Heinkenschloss, M., Hoppe, R.: Domain decomposition and balanced truncation model reduction for shape optimization of the Stokes system. Optimization Methods and Software 26(4-5), 643–669 (2011). DOI 10.1080/10556781003767904
4. Antil, H., Heinkenschloss, M., Hoppe, R., Sorensen, D.: Domain decomposition and model reduction for the numerical solution of PDE constrained optimization problems with localized optimization variables. Computing and Visualization in Science 13, 249–264 (2010)
5. Arndt, D., Bangerth, W., Blais, B., Fehling, M., Gassmöller, R., Heister, T., Heltai, L., Köcher, U., Kronbichler, M., Maier, M., Munch, P., Pelteret, J., Proell, S., Simon, K., Turcksin, B., Wells, D., Zhang, J.: The deal.II library, version 9.3. Journal of Numerical Mathematics (2021). URL https://dealii.org/deal93-preprint.pdf
6. Baiges, J., Codina, R., Idelsohn, S.: A domain decomposition strategy for reduced order models. application to the incompressible Navier–Stokes equations. Computer Methods in Applied Mechanics and Engineering 267, 23–42 (2013). DOI https://doi.org/10.1016/j.cma.2013.08.001
7. Ballarin, F., Faggiano, E., Ippolito, S., Manzoni, A., Quarteroni, A., Rozza, G., Scrofani, R.: Fast simulations of patient-specific haemodynamics of coronary artery bypass grafts based on a POD-Galerkin method and a vascular shape parametrisation. Journal of Computational Physics 315, 609–626 (2016). DOI 10.1016/j.jcp.2016.03.065
Fig. 19: *Test#3*. Top row: the ratio between the number of iterations obtained with ROM and FOM schemes versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). Bottom row: the ratio between the FOM and ROM computational time versus the number of basis functions employed to approximate the interface data (left) and the slave solution (right). The FOM simulation referred to fine discretization.

8. Ballarin, F., Rozza, G.: POD-Galerkin monolithic reduced order models for parametrized fluid-structure interaction problems: POD-Galerkin monolithic ROM for parametrized FSI problems. International Journal for Numerical Methods in Fluids 82 (2016). DOI 10.1002/fld.4252
9. Ballarin, F., Rozza, G., Maday, Y.: Reduced-order semi-implicit schemes for fluid-structure interaction problems. In: Model Reduction of Parametrized Systems, pp. 149–167. Springer (2017)
10. Barrault, M., Maday, Y., Nguyen, N.C., Patera, A.T.: An ‘empirical interpolation’ method: application to efficient reduced-basis discretization of partial differential equations. C. R. Math. Acad. Sci. Paris 339(9), 667–672 (2004)
11. Bègue, C., Bernardi, C., Debit, N., Maday, Y., Karniadakis, G., Mavriplis, C., Patera, A.: Nonconforming spectral element-finite element approximations for partial differential equations. In: Proceedings of the Eighth International Conference on Computing Methods in Applied Sciences and Engineering, vol. 75, pp. 109–125 (1989)
12. Benner, P., Ohlberger, M., Cohen, A., Willcox, K.: Model reduction and approximation: Theory and algorithms. Society for Industrial and Applied Mathematics, Philadelphia (2017). DOI 10.1137/1.9781611974829
13. Bernardi, C., Maday, Y., Patera, A.: A new nonconforming approach to domain decomposition: the mortar element method. In: Nonlinear partial differential equations and their applications. Collège de France Seminar, Vol. XI (Paris, 1989–1991), Pitman Res. Notes Math. Ser., vol. 299, pp. 13–51. Longman Sci. Tech., Harlow (1994)
14. Bernardi, C., Maday, Y., Rapetti, F.: Basics and some applications of the mortar element method. GAMM-Mitteilungen 28 (2005). DOI 10.1002/gamm.201490020
15. Bjørstad, P.E., B., S.C., Halpern, L., Kim, H., Kornhuber, R., Rahman, T., Widlund, O.: Domain Decomposition Methods in Science and Engineering XXIV. Lecture Notes in Computational Science and Engineering, Springer, Cham (2018)
Fig. 20: Test#3. Ratio between the number of iterations obtained with ROM and FOM schemes versus the achieved approximation error depending on the number of basis functions employed for the interface data (left) and the slave solution (right).

16. Brauchli, H., Oden, J.: Conjugate approximation functions in finite-element analysis. Quart. Appl. Math. 29, 65–90 (1971).

17. Chan, T., Smith, B., Zou, J.: Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids. Numerische Mathematik 73, 149–167 (1996). DOI 10.1007/s002110050189

18. Chaturantabut, S., Sorensen, D.: Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing 32(5), 2737–2764 (2010). DOI 10.1137/090766498

19. Chaturantabut, S., Sorensen, D.C.: Discrete empirical interpolation for nonlinear model reduction. In: Proceedings of the 48th IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference, pp. 4316–4321 (2009). DOI 10.1109/CDC.2009.5400045

20. Confalonieri, F., Corigliano, A., Dossi, M., Gornati, M.: A domain decomposition technique applied to the solution of the coupled electro-mechanical problem. International Journal for Numerical Methods in Engineering 93(2), 137–159 (2013). DOI https://doi.org/10.1002/nme.4375

21. Corigliano, A., Dossi, M., Mariani, S.: Model order reduction and domain decomposition strategies for the solution of the dynamic elastic–plastic structural problem. Computer Methods in Applied Mechanics and Engineering 290, 127–155 (2015). DOI https://doi.org/10.1016/j.cma.2015.02.021

22. de Boer, A., van Zuijlen, A., Bijl, H.: Review of coupling methods for non-matching meshes. Computer Methods in Applied Mechanics and Engineering 196(8), 1515–1525 (2007). DOI https://doi.org/10.1016/j.cma.2006.03.017

23. Deparis, S., Forti, D., Gervasio, P., Quarteroni, A.: INTERNODES: an accurate interpolation-based method for coupling the galerkin solutions of PDEs on subdomains featuring non-conforming interfaces. Computers & Fluids 141 (2016). DOI 10.1016/j.compfluid.2016.03.033

24. Deparis, S., Forti, D., Quarteroni, A.: A fluid-structure interaction algorithm using radial basis function interpolation between non-conforming interfaces, pp. 439–450. Modeling and Simulation in Science, Engineering and Technology. Springer (2016)

25. Eftang, J., Patera, A.: A port-reduced static condensation reduced basis element method for large component-synthesized structures: approximation and a posteriori error estimation. Advanced Modeling and Simulation in Engineering Sciences 1, 3 (2014). DOI 10.1186/2213-7467-1-3

26. Farhat, C., Grimberg, S., Manzoni, A., Quarteroni, A.: Computational bottlenecks for PROMs: precomputation and hyperreduction. In: P. Benner, S. Gritvet-Talocia, A. Quarteroni, G. Rozza, W. Schilders, L. Silveira (eds.) Model Order Reduction, vol. 2, Snapshot-Based Methods and Algorithms, pp. 181–244. De Gruyter (2020)

27. Forti, D., Rozza, G.: Efficient geometrical parametrisation techniques of interfaces for reduced-order modelling: application to fluid-structure interaction coupling problems. International Journal of Computational Fluid Dynamics 28(3-4), 158–169 (2014). DOI 10.1080/10618562.2014.932352

28. Fresca, S., Manzoni, A., Dedè, L., Quarteroni, A.: POD-enhanced deep learning-based reduced order models for the real-time simulation of cardiac electrophysiology in the left atrium. Frontiers in Physiology 12 (2021). DOI 10.3389/fphys.2021.679076

29. Gervasio, P., Quarteroni, A.: Analysis of the INTERNODES method for non-conforming discretizations of elliptic equations. Computer Methods in Applied Mechanics and Engineering 334, 138–166 (2018). DOI https://doi.org/10.1016/j.cma.2018.02.004
30. Gervasio, P., Quarteroni, A.: INTERNODES for heterogeneous couplings. In: P.E. Bjørstad, S. Brenner, L. Halpern, H. Kim, R. Kornhuber, T. Rahman, O. Widlund (eds.) Domain Decomposition Methods in Science and Engineering XXIV, pp. 59–71. Springer International Publishing, Cham (2018)

31. Gervasio, P., Quarteroni, A.: The INTERNODES method for non-conformant discretizations of PDEs. Communications on Applied Mathematics and Computation 1, 361–401 (2019). DOI 10.1007/s12967-019-00029-1

32. Glowinski, R., Dinh, Q., Periaux, J.: Domain decomposition methods for nonlinear problems in fluid dynamics. Computer Methods in Applied Mechanics and Engineering 301, 27–109 (1983). DOI https://doi.org/10.1016/0045-7825(83)90045-2. URL https://www.sciencedirect.com/science/article/pii/0045782583900452

33. Grepl, M., Maday, Y., Nguyen, N., Patera, A.: Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. ESAIM: Mathematical Modelling and Numerical Analysis 41 (2007). DOI 10.1051/m2an:2007031

34. Hartmann, S., Oliver, J., Weyler, R., Caute, J., Hernández, J.: A contact domain method for large deformation frictional contact problems. part 2: Numerical aspects. Computer Methods in Applied Mechanics and Engineering 198(33), 2607–2634 (2009). DOI https://doi.org/10.1016/j.cma.2009.03.009

35. Hesch, C., Gil, A., Arranz Carreño, A., Bonet, J., Betsch, P.: A mortar approach for fluid-structure interaction problems: Immersed strategies for deformable and rigid bodies. Computer Methods in Applied Mechanics and Engineering 278 (2014). DOI 10.1016/j.cma.2014.06.004

36. Hesthaven, J., Rozza, G., Stamm, B.: Certified Reduced Basis Methods for Parametrized Partial Differential Equations. Springer Nature (2016). DOI 10.1007/978-3-319-22470-1

37. Hoang, C., Choi, Y., Carlberg, K.: Domain-decomposition least-squares Petrov–Galerkin (DD–LSPG) nonlinear model reduction. Computer Methods in Applied Mechanics and Engineering 384, 113997 (2021). DOI https://doi.org/10.1016/j.cma.2021.113997. URL https://www.sciencedirect.com/science/article/pii/S0045782521003285

38. Huynh, P., Bao, D., Knezevic, D., Patera, A.: A static condensation reduced basis element method : approximation and a posteriori error estimation. ESAIM: Mathematical Modelling and Numerical Analysis 47(1), 213–251 (2013). DOI 10.1051/m2an/2012022

39. Iapichino, L., Quarteroni, A., Rozza, G.: A reduced basis hybrid method for the coupling of parametrized domains represented by fluidic networks. Computer Methods in Applied Mechanics and Engineering 221-222, 63–82 (2012). DOI https://doi.org/10.1016/j.cma.2012.02.005

40. Iapichino, L., Quarteroni, A., Rozza, G.: Reduced basis method and domain decomposition for elliptic problems in networks and complex parametrized geometries. Computers & Mathematics with Applications 71(1), 408–430 (2016). DOI https://doi.org/10.1016/j.camwa.2015.12.001

41. Iman, R., Helton, J.: An investigation of uncertainty and sensitivity analysis techniques for computer-models. Risk Analysis 8, 71 – 90 (2006). DOI 10.1111/j.1539-6924.1988.tb01155.x

42. Kerfriden, P., Goury, O., Rabczuk, T., Bordas, S.: A partitioned model order reduction approach to rationalise computational expenses in nonlinear fracture mechanics. Computer Methods in Applied Mechanics and Engineering 256, 169–188 (2013). DOI https://doi.org/10.1016/j.cma.2012.12.004

43. Legresley, P., Alonso, J.: Application of proper orthogonal decomposition (POD) to design decomposition methods / (2006)

44. Lorenzi, S., Cammi, A., Luzzi, L., Rozza, G.: POD-Galerkin method for finite volume approximation of Navier-Stokes and RANS equations. Computer Methods in Applied Mechanics and Engineering 311 (2016). DOI 10.1016/j.cma.2016.08.006

45. Lovgreen, A.E., Maday, Y., Rønquist, E.M.: A reduced basis element method for complex flow systems (2006)

46. Lucia, D., King, P., Beran, P.: Reduced order modeling of a two-dimensional flow with moving shocks. Computers & Fluids 32(7), 917–938 (2003). DOI https://doi.org/10.1016/S0045-7930(02)00035-X

47. Maday, Y., Nguyen, N., Patera, A., Pau, G.: A general multipurpose interpolation procedure: The magic points. Communications on Pure and Applied Analysis 8 (2009). DOI 10.3934/cpaa.2009.8.383

48. Mckay, M., Beckman, R., Conover, W.: latin hypercube sampling. Journal of Quality Technology 21, 197–205 (1989). DOI 10.1080/00224089.1989.11980063

49. Maier, I., Haasdonk, B.: A Dirichlet–Neumann reduced basis method for homogeneous domain decomposition problems. Computer Methods in Applied Mechanics and Engineering 198(24–25), 2219–2233 (2009). DOI 10.1016/j.cma.2009.02.016

50. Martin, A., Haasdonk, B.: Reduced basis approximation and a-posteriori error estimation for the coupled Stokes-Darcy system. Advances in Computational Mathematics 45(1), 331–357 (2019)

51. McGee, W., Padmanabhan, S.: Non–Conforming Finite Element Methods for Nonmatching Grids in Three Dimensions, Lect. Notes Comput. Sci. Eng., vol. 40, pp. 327–334. Springer, Berlin, Heidelberg (2005). DOI 10.1007/3-540-26825-1_32

52. Mays, M., Greenspan, D., Verwer, J.: A reduced basis element method: Application to a thermal fin problem. SIAM Journal on Scientific Computing 23(1), 240–258 (2001). DOI 10.1137/S0036139999358283

53. Negri, F., Manzoni, A., Amsallem, D.: Efficient model reduction of parametrized systems by matrix discrete empirical interpolation. Journal of Computational Physics 303, 431–454 (2015). DOI 10.1016/j.jcp.2015.09.046

54. Noack, B., Stankiewicz, W., Morzyński, M., Schmid, P.: Recursive dynamic mode decomposition of transient and post-transient wake flows. Journal of Fluid Mechanics 809, 845–872 (2016). DOI 10.1017/jfm.2016.678
Statements and Declarations

This research has been funded partly by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 740132, iHEART “An Integrated Heart Model for the simulation of the cardiac function”, P.I. Prof. A. Quarteroni) and partly by the Italian Ministry of University and Research (MIUR) within the PRIN (Research projects of relevant national interest 2017 “Modeling the heart across the scales: from cardiac cells to the whole organ” Grant Registration number 2017AXL54F).

This manuscript has no associated data.

Appendix A  A detailed error and computational costs analysis for the steady test case

We report here a detailed analysis of the model performances for the test case Test#1. described in Subsection 5.1, including a study of the effect of the ROM hyper–parameters on the approximation error, number of iterations before solution convergence, and overall computational costs. A simplified analysis for the other test cases, which confirm the results reported here, is given in Subsections 5.3.
involved, and a decreasing approximation error can be achieved every time that the number of basis functions for one between the reduced slave solution, master solution, or interface data, is increased. The major influence on the approximation of both master and slave solution comes from the reduction operated on in the interface data: it can be observed, indeed, a higher reduction of the approximation error when \( M_1 \) and \( M_2 \) are increased, whereas the error decrease is much slower when either \( n_1 \) or \( n_2 \) are increased. A more expensive and careful approximation of the interface data is therefore to be preferred over an expensive approximation of the slave and master solutions.

![Graphs comparing mean relative error over the slave and master solutions for different numbers of basis functions and basis sets.](image)

**Fig. 21:** Test\#1. \( H^1(\Omega_i) \) mean relative error (z-axis) over the slave (top row) and master (bottom row) solution for \( N_{test} = 20 \) different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the master solution and \( n_2 \), and the interface data \( M_1 \) and \( M_2 \) (x- and y-axis), and fixing the number of basis functions employed to approximate the slave solution.

The computational costs of the ROM are investigated by means of the number of iterations needed by the scheme to reach the solution interface convergence, as well as by the effective CPU time. Since both quantities depend on parameter instance, we compute the ratio between the iterations number of the ROM and FOM computations, as well as the ratio between the FOM and ROM computational times. The last ratio is able to describe the speed-up achieved by employing the ROM. Fig. 24 shows the variation of such iterations ratio depending on \( n_1, n_2, M_1 \) and \( M_2 \), comparing the results obtained with both the coarse and the fine discretization. The graphs show a dependency of the iterations number from the number of basis functions employed to compute the slave solution and interface data. Specifically, a greater approximation accuracy (and a number of basis functions) of the slave solution increases the number of iterations required to achieve the solution convergence, whereas a higher approximation accuracy of the interface data decreases the number of iterations. The same effect can be observed also in Fig. 25 where we compare the iterations ratio with the approximation error. Varying the number of basis functions to compute the master solution has instead a minor effect on the number of iterations.

We report the CPU time ratio compared to the basis functions number in Fig. 26, for both coarse and fine FOM discretizations, showing that independently of the number of basis functions imposed for any reduced quantity, we achieve very similar computational speed up.
Fig. 22: Test#1. $H^1(\Omega_i)$ mean relative error (z-axis) over the slave (top row) and master (bottom row) solution for $N_{test} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave and the master solution $n_1$ and $n_2$ ($x$- and $y$-axis), and fixing the number of basis functions employed to approximate interface data $M_1$ and $M_2$.

Fig. 23: Test#1. $H^1(\Omega_i)$ mean relative error (z-axis) over the slave (top row) and master (bottom row) solution for $N_{test} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave solution $n_1$ and the interface data $M_1$ and $M_2$ ($x$- and $y$-axis), and fixing the number of basis functions employed to approximate the master solution.

Appendix B Preliminary results employing Radial Basis Functions to interpolate data across the domain interfaces

In this appendix, we present a very preliminary investigation of the effects of utilizing a more advanced interpolation technique compared to the nearest neighbor approach outlined and utilized throughout the rest
Fig. 24: Test#1. Ratio between the number of iterations obtained with ROM and FOM schemes versus the number of basis functions employed to approximate the slave solutions (first row), the interface data (second row), or the master solution (third row), either employing the coarse discretization (left) or the fine discretization (right) for the FOM computation.

of the paper, to interpolate in the ROM algorithm the Dirichlet–Neumann data across the non-conforming interfaces. Specifically, in accordance with Remark 12 of Section 4, we have implemented Radial Basis Function (RBF) interpolation to interpolate the model solution between $\Gamma_2$ and $\Gamma_1$ (i.e., the Dirichlet interface data), as well as the residual vector between $\Gamma_1$ and $\Gamma_2$ (i.e., the Neumann interface data) at the subdomain interfaces.
Fig. 25: Test#1. Iterations ratio versus approximation error depending on the number of basis functions employed to approximate the slave solutions (first row), the interface data (second row), or the master solution (third row), either employing the coarse discretization (left) or the fine discretization (right) for the FOM computation.

We subsequently apply this RBF scheme to Test#2 in Subsection 5.2 and a partial application to the test case Test#3 in Subsection 5.3.
Fig. 26: Test#1. Ratio between the CPU time of FOM and ROM versus the number of basis functions employed to approximate the slave solutions (first row), the interface data (second row), or the master solution (third row), either employing the coarse discretization (left) or the fine discretization (right) for the FOM computation.

B.1 Test#2 - steady case: diffusion reaction equation with parametrized source

In this subsection we compute a comprehensive set of approximated ROM solutions of problem (25) with source term (28), by employing varying numbers of basis functions for the master, slave, and interface data,
and employing the RBF method to interpolate reduced order Dirichlet and Neumann data across the domain interface. This approach allows us to achieve different levels of approximation accuracy for each quantity depending on the parameters choice, reflecting the methodology outlined in Section 5.

For the sake of fairness, we employing the same parameters – encompassing both physical and reduced basis parameters (such as $N_{\text{train}}$ and $N_{\text{test}}$ dimensions) - of Test#2, presenting the same error analysis procedure as depicted in Fig. 12.

In Fig. 27 we therefore report the approximation errors obtained by varying the basis functions for two quantities between the master solution, slave solution, or interface data reduction, while keeping the basis functions for the third quantity constant. Specifically, the number of basis functions held constant in each graph are those required by the POD/DEIM algorithm to attain an approximation accuracy for the respective quantity of $10^{-5}$. These graphs aim at providing a comprehensive overview of the algorithm’s behavior.

![Fig. 27: Test#2. $H_1(\Omega_i)$ mean relative error (z-axis) over the solution for $N_{\text{test}} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the slave and the master solution $n_1$ and $n_2$, and the interface data $M_1$ and $M_2$ ($x$- and $y$-axis). On the top row, we fix the number of basis functions of the master problem to 8 (on the left) and to 10 for the slave problem (on the right), while on the bottom we fix the number of basis functions equal to 8 for the interface data representation.](image)

We observe that although the overall reduction in computational error appears to be more stable and consistent compared to the test presented in Subsection 5.2 (which utilized nearest neighbor interpolation), the overall values of the approximation errors are of a similar magnitude to those discussed in the paper.
Furthermore, in terms of the number of Dirichlet–Neumann iterations required for the algorithms to reach convergence, we have not observed any differences compared to the case of nearest neighbor interpolation. We believe this can be largely due to the number of FOM Dirichlet–Neumann iterations, and not to the interpolation method employed.

B.2 Test#3 - unsteady case: time–dependent heat equation

In this subsection we instead investigate the effect of the RBF interpolation on the ROM algorithm on test case #3 presented in Subsection 5.3.

Considering the results obtained in Subsection [B.1] and recognizing that the interpolation specifically targets the interface data, thereby impacting mostly the effectiveness of the DEIM, we proceed to investigate the influence of the RBF methods, by varying the prescribed approximation accuracy only for the interface data. Therefore, differently from previous tests, here we present error graphs obtained by varying the number of basis functions employed to approximate the interface quantities, while always prescribing an accuracy of $10^{-5}$ for both the slave and master solutions. The results are illustrated in Fig. 28, where we depict the approximation error on the solution in both the slave and master domains. We observed a remarkably similar

![Graph 1](image1.png) ![Graph 2](image2.png)

Fig. 28: Test#3. $H_1(\Omega_i)$ mean relative error over the solution for $N_{test} = 20$ different instances of the parameters between the FOM and ROM solutions varying the number of basis functions used to represent the interface data $M_1$ and $M_2$ when either nearest neighbor (P0) or Radial Basis Functions (RBF) interpolation is used to exchange the data across the interfaces. The number of basis functions for master and slave solutions is fixed to 10 and 8, respectively, to prescribe an approximation accuracy of $10^{-5}$.

behavior in the reduction of the approximation error when employing both the nearest neighbor approach and the RBF interpolation. It is noteworthy that the RBF proves to be more effective in approximating the master solution, while the nearest neighbor approach yields better results for the slave model. Nevertheless, these distinctions are quite minor, whereas the computational error magnitude remains consistent for each set of selected basis functions, regardless of the interpolation method employed.
Taking into account the results presented in this appendix, we observe a very similar overall outcome when using either the RBF or the nearest neighbor approach. While one might anticipate the RBF to exhibit a superior convergence rate, several factors could contribute to these findings: (i) the interface non-conformity may not be pronounced enough to favor the RBF over the simpler P0 method, (ii) the test cases considered are straightforward and do not experience significant issues with the basic P0 interpolation, (iii) the snapshots used for constructing the RBF do not incorporate interface interpolation. Each of these points requires separate investigation through a comprehensive convergence analysis, which will be the focus of future work.