Accurate and robust predictions for model order reduction via an adaptive, hybrid FOM/ROM approach

Sébastien Riffaud

Accurate and robust predictions for model order reduction via an adaptive, hybrid FOM/ROM approach

Sébastien Riffaud[a]

[a]*Sorbonne Université, Inria & CNRS, UMR 7598, Laboratoire Jacques-Louis Lions, 75012 Paris, France*

Abstract

In this paper, we introduce a hybrid approach that alternates between a high-fidelity model and a reduced-order model to speedup numerical simulations while maintaining accurate approximations. In particular, an error indicator based on a reduced-order approximation of the error is developed to determine when the reduced-order model is not sufficiently accurate and the high-fidelity model needs to be solved. Then, we propose an adaptive-extended version of the hybrid approach to update the reduced-order model with new solution snapshots. In particular, an interesting application concerns the acceleration of the offline stage of ROMs. The performance of the proposed method is finally evaluated on parametrized, time-dependent, nonlinear problems governed by the 1D Burgers' equation and 2D compressible Euler equations. The results demonstrate the accuracy and computational efficiency of the adaptive, hybrid approach with respect to the high-fidelity model.

Keywords: Reduced-order model, Proper orthogonal decomposition, Hyper-reduction, Error estimation, Adaptive model reduction

1. Introduction

Many applications in science and engineering require efficient numerical simulations, either due to runtime constraints in the case of large-scale computational models or due to the large number of simulations to run in the case of many-query problems. Projection-based reduced-order models (ROMs) [1, 2, 3, 4, 5, 6, 7, 8, 9] have been developed to dramatically reduce the computational cost associated with numerical solutions of parametric problems governed by partial differential equations (PDEs). They typically

consist of an offline stage in which training high-fidelity solutions are collected to define a low-dimensional approximation space. A popular dimensionality reduction technique for extracting the reduced basis that spans this approximation space is the proper orthogonal decomposition (POD) [10, 2]. Then, the low-dimensionality of the reduced basis is exploited during an online stage to enable fast or real-time predictions of new out-of-sample solutions.

In order to construct accurate ROMs over a wide range of input parameters, the standard approach consists in collecting a large number of high-fidelity solution snapshots at different training time-parameter instances. However, there are two major difficulties associated with this offline procedure. First, the number of high-fidelity simulations for sampling the solution manifold is limited in practice due to the expensive computational cost of the high-fidelity model (or high-dimensional model (HDM)). Second, if the training solution snapshots are too different from the predicted solution, the ROM may fail to deliver accurate approximations.

Several approaches have been proposed in the literature to address the aforementioned issues. In multifidelity methods [11, 12], the solutions of multilevel hierarchical models are combined to accelerate the estimation of statistics. The number of solution evaluations between the high- and low-fidelity models is balanced with respect to error and cost using an optimal model management. Alternatively, a hybrid HDM/ROM approach is proposed in [13, 14, 15]. The main idea is to switch between the HDM and ROM on the fly to achieve a prescribed level of accuracy. An *a posteriori* error estimation is employed to determine when the ROM is not sufficiently accurate and the HDM needs to be solved. Similarly, Zucatti et al. [16] introduced an adaptive, training-free ROM for convection-dominated problems based on hybrid snapshots. The latter are generated by combining space-local HDM and ROM solutions. An error strategy is employed to identify regions where the ROM is inaccurate and the HDM needs to be solved.

In this work, we focus on ROMs based on POD in the offline stage and on the least-squares Petrov-Galerkin (LSPG) projection [17, 18] equipped with the energy-conserving sampling and weighting (ECSW) hyper-reduction method [19, 20] in the online stage. In particular, we propose an approach [21] based on the principal directions of the reduced-order residual and reduced-order Jacobian matrix to improve the performance of the ECSW method. With this framework, we then present a hybrid HDM/ROM model to accelerate numerical simulations while maintaining accurate approximations. The main contribution concerns the introduction of an error indicator based

on a reduced-order approximation of the error. Compared to the hybrid model presented in [15], the error indicator proposed here can be employed for Galerkin or Petrov-Galerkin projection-based ROMs equipped with any hyper-reduction method. The computational efficiency of the resulting hybrid model depends on the accuracy of the underlying ROM to predict out-of-sample solutions. If the HDM is used in a large part of the computations, the hybrid model will be accurate but computationally expensive to solve. Conversely, if the ROM is sufficient to approximate the solution most of the time, this approach enables a significant reduction in the computational cost of numerical simulations. An interesting application of the hybrid model concerns the acceleration of the offline stage of ROMs. The main idea is to generate solution snapshots using the HDM only when the ROM is not sufficiently accurate. Moreover, instead of storing all solution snapshots, we propose to progressively enrich the ROM with the new solution snapshots in order to mitigate computational memory requirements. In this way, we also expect the ROM to become more robust for predicting out-of-sample solutions, which in turn allows to further accelerate the offline procedure.

The remainder of the paper is organized as follows. In Section 2, we introduce the HDM and ROM used in this work. Then, Section 3 describes the hybrid HDM/ROM approach and the underlying error indicator. In Section 4, we present an application of the adaptive, hybrid approach for accelerating the offline stage of ROMs. Section 5 demonstrates the performance of the proposed methods for parametrized, time-dependent, nonlinear problems. Finally, Section 6 draws some conclusions and perspectives.

2. Projection-based reduced-order model

2.1. High-dimensional model

Let the parameter domain $\mathcal{D}$ be a closed and bounded subset of the Euclidean space $\mathbb{R}^{N_p}$. In this work, we consider parametrized, time-dependent, nonlinear, semi-discrete problems of the form

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}(t;\boldsymbol{\theta}) = \mathbf{f}(\mathbf{u}(t;\boldsymbol{\theta}), t; \boldsymbol{\theta}) \\ \mathbf{u}(0;\boldsymbol{\theta}) = \mathbf{u}_0(\boldsymbol{\theta}), \end{cases}$$

where $t \geqslant 0$ denotes time, $\boldsymbol{\theta} \in \mathcal{D}$ denotes the input parameters, $\mathbf{u} : \mathbb{R}_+ \times \mathcal{D} \to \mathbb{R}^{N_{\mathrm{dof}}}$ denotes the solution, and $\mathbf{f} : \mathbb{R}^{N_{\mathrm{dof}}} \times \mathbb{R}_+ \times \mathcal{D} \to \mathbb{R}^{N_{\mathrm{dof}}}$ denotes the nonlinear function resulting from the semi-discretization of the PDE.

This semi-discrete problem is then integrated in time by an implicit scheme, leading to the nonlinear system of equations

$$\mathbf{r}_k(\mathbf{u}_k(\boldsymbol{\theta});\boldsymbol{\theta}) = \mathbf{0} \qquad \forall k \in \{1,\dots,N_t\}, \tag{1}$$

which is solved for $\mathbf{u}_k(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\mathrm{dof}}}$ by the Newton-Raphson method. Here, $\mathbf{u}_k(\boldsymbol{\theta})$ denotes the time-discrete counterpart of $\mathbf{u}(t_k;\boldsymbol{\theta})$, $t_k \geqslant 0$ denotes the k-th time-instance, and $\mathbf{r}_k : \mathbb{R}^{N_{\mathrm{dof}}} \times \mathcal{D} \to \mathbb{R}^{N_{\mathrm{dof}}}$ denotes the fully discrete approximation of the semi-discrete residual $\mathbf{r}(\mathbf{u}(t_k;\boldsymbol{\theta}),t_k;\boldsymbol{\theta}) = \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}(t_k;\boldsymbol{\theta}) - \mathbf{f}(\mathbf{u}(t_k;\boldsymbol{\theta}),t_k;\boldsymbol{\theta})$. In the following, this computational model will be referred to as the HDM.

2.2. Reduced-order model

In the ROM, the high-dimensional solution $\mathbf{u}_k(\boldsymbol{\theta})$ is approximated on a low-dimensional subspace to reduce the number of degrees of freedom:

$$\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}) = \mathbf{u}_{\mathrm{off}}(\boldsymbol{\theta}) + \mathbf{V}\mathbf{y}_k(\boldsymbol{\theta}),$$

where the offset $\mathbf{u}_{\mathrm{off}}(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\mathrm{dof}}}$ is set to the initial condition $\mathbf{u}_0(\boldsymbol{\theta})$, the reduced basis $\mathbf{V} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{v}}}$ is constructed by POD, the reduced coordinates $\mathbf{y}_k(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\mathrm{v}}}$ are determined by the LSPG projection equipped with the ECSW hyper-reduction method, and $N_{\mathrm{v}} \ll N_{\mathrm{dof}}$.

2.2.1. Proper orthogonal decomposition

During the offline stage, high-dimensional solution snapshots are collected at different training time-parameter instances $\{(t_1^{\mathrm{train}},\boldsymbol{\theta}_1^{\mathrm{train}}),\dots,(t_{N_{\mathrm{s}}}^{\mathrm{train}},\boldsymbol{\theta}_{N_{\mathrm{s}}}^{\mathrm{train}})\}$. Let the resulting snapshot database be $\mathbf{S}_{\mathrm{u}} = [\mathbf{s}_{\mathrm{u}_1},\dots,\mathbf{s}_{\mathrm{u}_{N_{\mathrm{s}}}}] \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s}}}$, where $\mathbf{s}_{\mathrm{u}_k} \in \mathbb{R}^{N_{\mathrm{dof}}}$ is the discrete approximation of $\mathbf{u}(t_k^{\mathrm{train}};\boldsymbol{\theta}_k^{\mathrm{train}})$. The reduced basis $\mathbf{V}$ is computed by POD to minimize the projection error of the snapshots onto this basis:

$$\begin{cases} \underset{\mathbf{V}\in\mathbb{R}^{N_{\mathrm{dof}}\times N_{\mathrm{v}}}}{\text{minimize}} & \left\|\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}})\right\|_F \\ \text{subject to} & \mathbf{V}^T\mathbf{V} = \mathbf{I}, \end{cases}$$

where $\mathbf{S}_{\mathrm{u}_{\mathrm{off}}} = [\mathbf{u}_{\mathrm{off}}(\boldsymbol{\theta}_1^{\mathrm{train}}),\dots,\mathbf{u}_{\mathrm{off}}(\boldsymbol{\theta}_{N_{\mathrm{s}}}^{\mathrm{train}})] \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s}}}$, and $\|\cdot\|_F$ stands for the Frobenius norm. Notably, the solution to this minimization problem is given by the Eckart-Young theorem [22]. Let $\widehat{\mathbf{S}}_{\mathrm{u}} = \boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Psi}^T$ be the truncated singular value decomposition (SVD) of $\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}$, where $\boldsymbol{\Phi} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{v}}}$ is

an orthonormal matrix, $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_{N_\mathrm{v}}) \in \mathbb{R}^{N_\mathrm{v} \times N_\mathrm{v}}$ is a diagonal matrix with non-negative real number on the diagonal, and $\mathbf{\Psi} \in \mathbb{R}^{N_\mathrm{s} \times N_\mathrm{v}}$ is an orthogonal matrix. The reduced basis corresponds to the N_v leading left singular vectors (i.e., $\mathbf{V} = \mathbf{\Phi}$), and the square of the projection error is equal to the sum of the neglected singular values squared:

$$\left\|\mathbf{S}_\mathrm{u} - \mathbf{S}_{\mathrm{u}_\mathrm{off}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_\mathrm{u} - \mathbf{S}_{\mathrm{u}_\mathrm{off}})\right\|_F^2 = \sum_{n=N_\mathrm{v}+1}^{\min(N_\mathrm{dof}, N_\mathrm{s})} \sigma_n^2.$$

In particular, the reduced basis dimension N_v is chosen here as the smallest integer such that the relative projection error is less than a user-defined tolerance $\varepsilon_\mathrm{pod} \in [0, 1]$:

$$\left\|\mathbf{S}_\mathrm{u} - \mathbf{S}_{\mathrm{u}_\mathrm{off}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_\mathrm{u} - \mathbf{S}_{\mathrm{u}_\mathrm{off}})\right\|_F \leqslant \varepsilon_\mathrm{pod} \left\|\mathbf{S}_\mathrm{u}\right\|_F.$$

Furthermore, the approximation of the different variables is decorrelated since they can have different orders of magnitude. For example, in Section 5.2, the variables $\{\rho, \rho u, \rho v, E\}$ are approximated on $\{\mathbf{V}_\rho, \mathbf{V}_{\rho u}, \mathbf{V}_{\rho v}, \mathbf{V}_E\}$ respectively, and the reduced basis is defined as

$$\mathbf{V} = \begin{bmatrix} \mathbf{V}_\rho & & & \\ & \mathbf{V}_{\rho u} & & \\ & & \mathbf{V}_{\rho v} & \\ & & & \mathbf{V}_E \end{bmatrix}.$$

2.2.2. Least-squares Petrov-Galerkin method

During the online stage, the reduced coordinates $\mathbf{y}_k(\boldsymbol{\theta})$ are determined by the residual-minimizing LSPG method:

$$\begin{cases} \underset{\mathbf{y}_k(\boldsymbol{\theta}) \in \mathbb{R}^{N_\mathrm{v}}}{\text{minimize}} \ \|\mathbf{r}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\|_2^2 & \forall k \in \{1, \ldots, N_t\} \\ \mathbf{y}_0(\boldsymbol{\theta}) = \mathbf{V}^T\big(\mathbf{u}_0(\boldsymbol{\theta}) - \mathbf{u}_\mathrm{off}(\boldsymbol{\theta})\big). \end{cases} \tag{2}$$

Notably, the first optimality condition leads to the Petrov-Galerkin projection:

$$\begin{cases} \left(\mathbf{J}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\mathbf{V}\right)^T \mathbf{r}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta}) = \mathbf{0} & \forall k \in \{1, \ldots, N_t\} \\ \mathbf{y}_0(\boldsymbol{\theta}) = \mathbf{V}^T\big(\mathbf{u}_0(\boldsymbol{\theta}) - \mathbf{u}_\mathrm{off}(\boldsymbol{\theta})\big), \end{cases}$$

where $\mathbf{J}_k(\mathbf{u}_k(\boldsymbol{\theta}); \boldsymbol{\theta}) \in \mathbb{R}^{N_\mathrm{dof} \times N_\mathrm{dof}}$ denotes the Jacobian of $\mathbf{r}_k(\mathbf{u}_k(\boldsymbol{\theta}); \boldsymbol{\theta})$ with respect to $\mathbf{u}_k$.

2.2.3. Energy-conserving sampling and weighting method

In this work, the nonlinear least-squares problem (2) is solved by the Gauss-Newton algorithm, which involves the projected reduced-order quantities

$$\overline{\mathbf{J}}_k(\boldsymbol{\theta})^T\overline{\mathbf{J}}_k(\boldsymbol{\theta}) \quad \text{and} \quad \overline{\mathbf{J}}_k(\boldsymbol{\theta})^T\overline{\mathbf{r}}_k(\boldsymbol{\theta}),$$

where $\overline{\mathbf{r}}_k(\boldsymbol{\theta}) = \mathbf{r}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta});\boldsymbol{\theta}) \in \mathbb{R}^{N_{\text{dof}}}$ and $\overline{\mathbf{J}}_k(\boldsymbol{\theta}) = \mathbf{J}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta});\boldsymbol{\theta})\mathbf{V} \in \mathbb{R}^{N_{\text{dof}}\times N_{\text{v}}}$. Unfortunately, as the residual is nonlinear with respect to the solution, the evaluation of these quantities typically scales with the size of the HDM, unless hyper-reduction techniques are employed. For this reason, we consider the ECSW method, in which the standard scalar product is approximated by the weighted sum:

$$\mathbf{a}^T\mathbf{b} \approx \sum_{i=1}^{N_{\text{dof}}} \omega_i a_i b_i,$$

where $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{N_{\text{dof}}}$, and only a few weights $\omega_i \geqslant 0$ are nonzero (i.e., $\|\boldsymbol{\omega}\|_0 \ll N_{\text{dof}}$, where $\|\cdot\|_0$ stands for the ℓ_0 pseudo-norm). In this way, the reduced-order quantities $\overline{\mathbf{r}}_k(\boldsymbol{\theta})$ and $\overline{\mathbf{J}}_k(\boldsymbol{\theta})$ need to be evaluated only at a few points of the spatial domain, namely those associated with nonzero weights, in order to speedup computations. This set of points will be referred to as the reduced mesh in the following, and N_{p} will denote the size of the reduced mesh.

The weights $\boldsymbol{\omega} = [\omega_1, \ldots, \omega_{N_{\text{dof}}}] \in \mathbb{R}^{N_{\text{dof}}}$ are determined during the offline stage to best approximate the exact scalar product. Given snapshots of the reduced-order residual $\mathbf{r}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta});\boldsymbol{\theta})$ and reduced-order Jacobian matrix $\mathbf{J}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta});\boldsymbol{\theta})\mathbf{V}$ at different time-parameter instances $(t_k, \boldsymbol{\theta}) \in \{(t_1^{\text{train}}, \boldsymbol{\theta}_1^{\text{train}}), \ldots, (t_{N_{\text{s}}}^{\text{train}}, \boldsymbol{\theta}_{N_{\text{s}}}^{\text{train}})\}$, the evaluation of the projected reduced-order quantities leads to an approximation problem of the form

$$\mathbf{C}\boldsymbol{\omega} \approx \mathbf{d}, \tag{3}$$

where $\mathbf{C} \in \mathbb{R}^{N_{\text{c}}\times N_{\text{dof}}}$ and $\mathbf{d} \in \mathbb{R}^{N_{\text{c}}}$ are defined in Appendix A . The weights $\boldsymbol{\omega}$ are then solution to the non-negative least-squares (NNLS) problem

$$\min_{\boldsymbol{\omega}\geqslant 0} \|\mathbf{C}\boldsymbol{\omega} - \mathbf{d}\|_2^2, \tag{4}$$

which is solved by the Matlab implementation of the Lawson-Hanson algorithm [23]. This algorithm promotes sparsity in the solution and terminates when the stopping criterion $\|\mathbf{C}\boldsymbol{\omega} - \mathbf{d}\|_2 \leqslant \varepsilon_{\text{ecsw}} \|\mathbf{d}\|_2$ is satisfied for a user-defined tolerance $\varepsilon_{\text{ecsw}} \in [0, 1]$.

Notably, we propose in Appendix A to construct $\mathbf{C}$ using the principal directions of the reduced-order residual and reduced-order Jacobian matrix.

Specifically, instead of constructing $\mathbf{C}$ directly from snapshots of the reduced-order quantities, we consider the POD basis $\mathbf{U} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{u}}}$ of these snapshots, and we define $\mathbf{C}^T = \mathbf{U}\mathbf{\Lambda}$, where $\mathbf{\Lambda} \in \mathbb{R}^{N_{\text{u}} \times N_{\text{u}}}$ is the diagonal matrix containing the singular values associated to $\mathbf{U}$. This modification allows to drastically reduce the size of the approximation problem (3) and, consequently, the runtime for solving the NNLS problem (4). In addition, the approximation problem (3) is fully described by the reduced basis $\mathbf{U}$ and the singular values $\mathbf{\Lambda}$, which will be a key point for updating the reduced mesh in Section 4.2.

Remark 1. *To compute snapshots of* $\mathbf{r}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})$ *and* $\mathbf{J}_k(\widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\mathbf{V}$*, the reduced-order solution* $\widetilde{\mathbf{u}}_k(\boldsymbol{\theta})$ *is replaced in practice by the projection of the high-dimensional solution* $\mathbf{u}_{\mathit{off}}(\boldsymbol{\theta}) + \mathbf{V}\mathbf{V}^T(\mathbf{u}_k(\boldsymbol{\theta}) - \mathbf{u}_{\mathit{off}}(\boldsymbol{\theta}))$ *in order to save the computation time associated with the evaluation of the ROM.*

3. Hybrid HDM/ROM model

Given the previous HDM and ROM, we now introduce the hybrid model, which aims at accelerating numerical simulations while maintaining accurate approximations. The main idea is to compute the solution using the HDM only when the ROM is not sufficiently accurate. Specifically, we want the relative space-time error to be smaller than a prescribed tolerance $\varepsilon_{\text{err}} \in [0, 1]$:

$$\sqrt{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta}) - \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2} \leqslant \varepsilon_{\text{err}} \sqrt{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta})\|_2^2}.$$

For this purpose, we impose at each time-step t_k that

$$\|\mathbf{u}_k(\boldsymbol{\theta}) - \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2 \leqslant \varepsilon_{\text{err}} \|\mathbf{u}_k(\boldsymbol{\theta})\|_2 ,$$

which is equivalent to bound the error $\mathbf{e}_k(\boldsymbol{\theta}) = \mathbf{u}_k(\boldsymbol{\theta}) - \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})$ as follows

$$\|\mathbf{e}_k(\boldsymbol{\theta})\|_2 \leqslant \varepsilon_{\text{err}} \|\mathbf{e}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2 . \tag{5}$$

However, computing the error $\mathbf{e}_k(\boldsymbol{\theta})$ is computationally prohibitive as it requires the evaluation of the high-dimensional solution $\mathbf{u}_k(\boldsymbol{\theta})$. For this reason, we employ an error estimation $\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) \approx \mathbf{e}_k(\boldsymbol{\theta})$. The latter is presented in Section 3.1, and the resulting hybrid model is described in Algorithm 1.

Algorithm 1 Hybrid model

Input: $\boldsymbol{\theta} \in \mathcal{D}$, and $\varepsilon_{\text{err}} \in [0, 1]$.
Output: $\{\widehat{\mathbf{u}}_1(\boldsymbol{\theta}), \ldots, \widehat{\mathbf{u}}_{N_t}(\boldsymbol{\theta})\} \in \mathbb{R}^{N_{\text{dof}}}$.

```
Initialize û_0(θ) = u_0(θ);
for k ∈ {1, ..., N_t} do
    Compute ũ_k(θ) by solving equation (2);
    Compute ẽ_k(θ) by solving equation (6);
    if ‖ẽ_k(θ)‖_2 ⩽ ε_err ‖ẽ_k(θ) + ũ_k(θ)‖_2 then
        Set û_k(θ) = ũ_k(θ);
    else
        Compute u_k(θ) by solving equation (1);
        Set û_k(θ) = u_k(θ);
    end
end
```

3.1. Error estimation

In this work, the error is estimated by a second ROM, which will be referred to as the error indicator in the following and which uses the same methods as the ROM presented in Section 2.2. Specifically, the error $\mathbf{e}_k(\boldsymbol{\theta})$ is approximated by

$$\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) = \mathbf{e}_{\text{off}}(\boldsymbol{\theta}) + \mathbf{W}\mathbf{z}_k(\boldsymbol{\theta}),$$

where the offset $\mathbf{e}_{\text{off}}(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\text{dof}}}$ is set to the initial error $\mathbf{e}_0(\boldsymbol{\theta})$, the reduced basis $\mathbf{W} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{w}}}$ is constructed by POD, and the reduced coordinates $\mathbf{z}_k(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\text{w}}}$ are determined by the LSPG projection equipped with the ECSW hyper-reduction method. Note that the initial error $\mathbf{e}_0(\boldsymbol{\theta})$ vanishes here since $\mathbf{u}_{\text{off}}(\boldsymbol{\theta}) = \mathbf{u}_0(\boldsymbol{\theta})$.

During the offline stage, solution snapshots of the ROM are collected to assemble the database $\mathbf{S}_{\text{e}} = [\mathbf{s}_{\text{e}_1}, \ldots, \mathbf{s}_{\text{e}_{N_{\text{s}}}}] \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{s}}}$, where $\mathbf{s}_{\text{e}_k}$ is the snapshot of the error $\mathbf{e}_k(\boldsymbol{\theta})$ at time-parameter instance $(t_k, \boldsymbol{\theta}) = (t_k^{\text{train}}, \boldsymbol{\theta}_k^{\text{train}})$. The reduced basis $\mathbf{W}$ is then computed by POD from $\mathbf{S}_{\text{e}}$ without offset (i.e., $\mathbf{S}_{\text{e}_{\text{off}}} = \mathbf{0}$).

The reduced coordinates $\mathbf{z}_k(\boldsymbol{\theta})$ are determined during the online stage by the LSPG method:

$$\begin{cases} \underset{\mathbf{z}_k(\boldsymbol{\theta}) \in \mathbb{R}^{N_{\text{w}}}}{\text{minimize}} \ \|\mathbf{r}_k(\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\|_2^2 & \forall k \in \{1, \ldots, N_t\} \\ \mathbf{z}_0(\boldsymbol{\theta}) = \mathbf{W}^T\big(\mathbf{e}_0(\boldsymbol{\theta}) - \mathbf{e}_{\text{off}}(\boldsymbol{\theta})\big), \end{cases} \tag{6}$$

where the reduced-order solution $\widetilde{\mathbf{u}}_k(\boldsymbol{\theta})$ is known. This nonlinear least-squares problem is then solved by the Gauss-Newton algorithm, and the ECSW method is employed to reduce the computational cost associated with the evaluation of the projected reduced-order quantities. Compared to the procedure described in Section 2.2.3, the only difference is that the approximation problem (3) is assembled using snapshots of the reduced-order residual $\mathbf{r}_k(\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})$ and reduced-order Jacobian matrix $\mathbf{J}_k(\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\mathbf{W}$.

Remark 2. *To compute snapshots of* $\mathbf{r}_k(\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})$ *and* $\mathbf{J}_k(\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta}); \boldsymbol{\theta})\mathbf{W}$*, the reduced-order error* $\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})$ *is replaced in practice by the projection of the error* $\mathbf{e}_{off}(\boldsymbol{\theta}) + \mathbf{W}\mathbf{W}^T(\mathbf{e}_k(\boldsymbol{\theta}) - \mathbf{e}_{off}(\boldsymbol{\theta}))$ *in order to save the computation time associated with the evaluation of the error indicator.*

Remark 3. *Replacing* $\mathbf{e}_k(\boldsymbol{\theta})$ *with* $\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})$ *enables the efficient evaluation of the error criterion* (5)*, since the latter scales with the size of the ROM and error indicator:*

$$
\begin{aligned}
\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2 &= \|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2^2 + 2\,\langle \widetilde{\mathbf{e}}_k(\boldsymbol{\theta}), \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\rangle_2 + \|\widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2\,, \\
\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2^2 &= \|\mathbf{e}_{off}(\boldsymbol{\theta})\|_2^2 + 2\,\left\langle \mathbf{W}^T\mathbf{e}_{off}(\boldsymbol{\theta}), \mathbf{z}_k(\boldsymbol{\theta})\right\rangle_2 + \|\mathbf{z}_k(\boldsymbol{\theta})\|_2^2\,, \\
\langle \widetilde{\mathbf{e}}_k(\boldsymbol{\theta}), \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\rangle_2 &= \langle \mathbf{e}_{off}(\boldsymbol{\theta}), \mathbf{u}_{off}(\boldsymbol{\theta})\rangle_2 + \left\langle \mathbf{V}^T\mathbf{e}_{off}(\boldsymbol{\theta}), \mathbf{y}_k(\boldsymbol{\theta})\right\rangle_2 \\
&\quad + \left\langle \mathbf{W}^T\mathbf{u}_{off}(\boldsymbol{\theta}), \mathbf{z}_k(\boldsymbol{\theta})\right\rangle_2 + \left\langle \mathbf{V}^T\mathbf{W}\mathbf{z}_k(\boldsymbol{\theta}), \mathbf{y}_k(\boldsymbol{\theta})\right\rangle_2\,, \\
\|\widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2 &= \|\mathbf{u}_{off}(\boldsymbol{\theta})\|_2^2 + 2\,\left\langle \mathbf{V}^T\mathbf{u}_{off}(\boldsymbol{\theta}), \mathbf{y}_k(\boldsymbol{\theta})\right\rangle_2 + \|\mathbf{y}_k(\boldsymbol{\theta})\|_2^2\,,
\end{aligned}
$$

where $\|\mathbf{u}_{off}(\boldsymbol{\theta})\|_2^2$, $\|\mathbf{e}_{off}(\boldsymbol{\theta})\|_2^2$, $\langle \mathbf{e}_{off}(\boldsymbol{\theta}), \mathbf{u}_{off}(\boldsymbol{\theta})\rangle_2$, $\mathbf{V}^T\mathbf{u}_{off}(\boldsymbol{\theta})$, $\mathbf{V}^T\mathbf{e}_{off}(\boldsymbol{\theta})$, $\mathbf{W}^T\mathbf{u}_{off}(\boldsymbol{\theta})$, $\mathbf{W}^T\mathbf{e}_{off}(\boldsymbol{\theta})$*, and* $\mathbf{V}^T\mathbf{W}$ *can be precomputed once for all.*

3.2. Computational complexity

According to Algorithm 1, the computational cost of the hybrid model is given by

$$
(c_{\text{rom}} + c_{\text{err}})N_t + c_{\text{hdm}}n_{\text{hdm}}, \tag{7}
$$

where c_{rom} denotes the cost for solving equation (2), c_{err} denotes the cost for solving equation (6), c_{hdm} denotes the cost for solving equation (1), and n_{hdm} denotes the number of time instances in which the HDM is employed to compute the solution. In particular, if the HDM is never used (i.e., $n_{\text{hdm}} = 0$), the computational cost of the hybrid model corresponds to the cost associated with the evaluation of the ROM and error indicator. Consequently, if the

ROM is sufficient to approximate the solution most of the time, the hybrid model enables a significant reduction in the computational cost of numerical simulations, since $c_{\text{rom}}, c_{\text{err}} \ll c_{\text{hdm}}$.

4. Acceleration of the offline stage of ROMs via an adaptive, hybrid approach

An interesting application of the hybrid model concerns the acceleration of the offline stage of ROMs. The main idea is to generate solution snapshots using the HDM only when the ROM is not sufficiently accurate. In this way, if the ROM is sufficient to approximate the solution snapshots most of the time, then this approach enables a significant reduction in the computational cost of the offline stage. Moreover, instead of storing all solution snapshots, the ROM of the hybrid model is progressively enriched with the new solution snapshots to mitigate computational memory requirements. Notably, an efficient procedure for updating the reduced basis $\mathbf{V}$ is described in Section 4.1. The same procedure is then applied to the reduced bases $\mathbf{U}$ and $\mathbf{W}$ in order to fully update the ROM and error indicator. The resulting adaptive-extended version of the hybrid model is presented in Algorithm 2.

Algorithm 2 Adaptive, hybrid approach

Input: $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_\theta}\} \in \mathcal{D}$.
Output: $\{\widehat{\mathbf{u}}_k(\boldsymbol{\theta}_1), \ldots, \widehat{\mathbf{u}}_k(\boldsymbol{\theta}_{N_\theta})\}_{k=1}^{N_t} \in \mathbb{R}^{N_{\text{dof}}}$.

1: **for** $\boldsymbol{\theta} \in \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_\theta}\}$ **do**
2: Compute $\{\widehat{\mathbf{u}}_1(\boldsymbol{\theta}), \ldots, \widehat{\mathbf{u}}_{N_t}(\boldsymbol{\theta})\}$ using Algorithm 1;
3: Set $\mathbf{S}_{\text{u}}^{\text{new}} = [\widehat{\mathbf{u}}_1(\boldsymbol{\theta}), \ldots, \widehat{\mathbf{u}}_{N_t}(\boldsymbol{\theta})]$ and $\mathbf{S}_{\text{u}_{\text{off}}}^{\text{new}} = [\mathbf{u}_0(\boldsymbol{\theta}), \ldots, \mathbf{u}_0(\boldsymbol{\theta})]$;
4: Update the reduced basis according to Section 4.1;
5: Update the reduced mesh according to Section 4.2;
6: Update the error indicator according to Section 4.3;
7: **end**

4.1. Update of the reduced basis

Let $\mathbf{V}_{\text{old}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{v}_{\text{old}}}}$ be the POD basis computed from the databases $\mathbf{S}_{\text{u}}^{\text{old}}, \mathbf{S}_{\text{u}_{\text{off}}}^{\text{old}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{s}_{\text{old}}}}$, and assume that new snapshots $\mathbf{S}_{\text{u}}^{\text{new}}, \mathbf{S}_{\text{u}_{\text{off}}}^{\text{new}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{s}_{\text{new}}}}$ become available. The objectif is to find the reduced basis $\mathbf{V}$ that verifies

$$\left\| \mathbf{S}_{\text{u}} - \mathbf{S}_{\text{u}_{\text{off}}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\text{u}} - \mathbf{S}_{\text{u}_{\text{off}}}) \right\|_F \leqslant \varepsilon_{\text{pod}} \left\| \mathbf{S}_{\text{u}} \right\|_F , \tag{8}$$

where $\mathbf{S}_{\mathrm{u}} = [\mathbf{S}_{\mathrm{u}}^{\mathrm{old}}, \mathbf{S}_{\mathrm{u}}^{\mathrm{new}}] \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s}}}$, $\mathbf{S}_{\mathrm{u_{off}}} = [\mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{old}}, \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{new}}] \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s}}}$, and $N_{\mathrm{s}} = N_{\mathrm{s_{old}}} + N_{\mathrm{s_{new}}}$. For this purpose, we consider the approach described in [24], which relies on the compressed snapshot matrix

$$\widehat{\mathbf{S}} = \begin{bmatrix} \widehat{\mathbf{S}}_{\mathrm{old}} & \widehat{\mathbf{S}}_{\mathrm{new}} \end{bmatrix} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s}}}.$$

Here, $\widehat{\mathbf{S}}_{\mathrm{old}} = \boldsymbol{\Phi}_{\mathrm{old}} \boldsymbol{\Sigma}_{\mathrm{old}} \boldsymbol{\Psi}_{\mathrm{old}}^T \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s_{old}}}}$ (resp. $\widehat{\mathbf{S}}_{\mathrm{new}} = \boldsymbol{\Phi}_{\mathrm{new}} \boldsymbol{\Sigma}_{\mathrm{new}} \boldsymbol{\Psi}_{\mathrm{new}}^T \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\mathrm{s_{new}}}}$) denotes the truncated SVD of $\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{old}}$ (resp. $\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{new}}$), where $\boldsymbol{\Phi}_{\mathrm{old}} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\hat{\mathrm{v}}_{\mathrm{old}}}}$ (resp. $\boldsymbol{\Phi}_{\mathrm{new}} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\hat{\mathrm{v}}_{\mathrm{new}}}}$) is unitary, $\boldsymbol{\Sigma}_{\mathrm{old}} \in \mathbb{R}^{N_{\hat{\mathrm{v}}_{\mathrm{old}}} \times N_{\hat{\mathrm{v}}_{\mathrm{old}}}}$ (resp. $\boldsymbol{\Sigma}_{\mathrm{new}} \in \mathbb{R}^{N_{\hat{\mathrm{v}}_{\mathrm{new}}} \times N_{\hat{\mathrm{v}}_{\mathrm{new}}}}$) is diagonal, $\boldsymbol{\Psi}_{\mathrm{old}} \in \mathbb{R}^{N_{\mathrm{s_{old}}} \times N_{\hat{\mathrm{v}}_{\mathrm{old}}}}$ (resp. $\boldsymbol{\Psi}_{\mathrm{new}} \in \mathbb{R}^{N_{\mathrm{s_{new}}} \times N_{\hat{\mathrm{v}}_{\mathrm{new}}}}$) is unitary, and the rank $N_{\hat{\mathrm{v}}_{\mathrm{old}}}$ (resp. $N_{\hat{\mathrm{v}}_{\mathrm{new}}}$) is chosen so that the relative error is less than a user-defined tolerance $\varepsilon_1 \in [0, 1]$:

$$\left\| (\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{old}}) - \widehat{\mathbf{S}}_{\mathrm{old}} \right\|_F \leqslant \varepsilon_1 \left\| \mathbf{S}_{\mathrm{u}}^{\mathrm{old}} \right\|_F \tag{9a}$$

$$\left(\text{resp.} \quad \left\| (\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{new}}) - \widehat{\mathbf{S}}_{\mathrm{new}} \right\|_F \leqslant \varepsilon_1 \left\| \mathbf{S}_{\mathrm{u}}^{\mathrm{new}} \right\|_F \right). \tag{9b}$$

The reduced basis $\mathbf{V}$ is then computed by POD from the compressed snapshot matrix $\widehat{\mathbf{S}}$ using the user-defined tolerance $\varepsilon_2 \in [0, 1]$:

$$\left\| \widehat{\mathbf{S}} - \mathbf{V}\mathbf{V}^T \widehat{\mathbf{S}} \right\|_F \leqslant \varepsilon_2 \left\| \mathbf{S}_{\mathrm{u}} \right\|_F . \tag{10}$$

The advantage of this approach is twofold. On the one hand, only the low-rank representation of $\widehat{\mathbf{S}}$ needs to be stored in memory, rather than the full matrices $(\mathbf{S}_{\mathrm{u}}, \mathbf{S}_{\mathrm{u_{off}}})$, which would have been computationally prohibitive. On the other hand, the resulting reduced basis $\mathbf{V}$ verifies

$$\left\| \mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u_{off}}} - \mathbf{V}\mathbf{V}^T (\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u_{off}}}) \right\|_F \leqslant \sqrt{\varepsilon_1^2 + \varepsilon_2^2} \left\| \mathbf{S}_{\mathrm{u}} \right\|_F \tag{11}$$

according to Proposition 1, and we set $\varepsilon_1 = \varepsilon_2 = \frac{\varepsilon_{\mathrm{pod}}}{\sqrt{2}}$ in the following to ensure that equation (8) holds.

Moreover, we propose in this work to combine this approach with the fast SVD update of Brand [25] in order to reduce the computational cost of updating the reduced basis. The main idea is to progressively update the low-rank representation of $\widehat{\mathbf{S}}$ with the new solution snapshots, instead of computing the SVD from scratch. To this end, let $\widehat{\mathbf{S}} = \widehat{\boldsymbol{\Phi}} \widehat{\boldsymbol{\Sigma}} \widehat{\boldsymbol{\Psi}}$ be the SVD of $\widehat{\mathbf{S}}$. We introduce the extended reduced basis $\widehat{\mathbf{V}} \in \mathbb{R}^{N_{\mathrm{dof}} \times N_{\hat{\mathrm{v}}}}$ defined as

the left singular vectors of $\widehat{\mathbf{S}}$ (i.e., $\widehat{\mathbf{V}} = \widehat{\mathbf{\Phi}}$), where the first N_{v} columns of $\widehat{\mathbf{V}}$ correspond to the reduced basis $\mathbf{V}$, while the $N_{\hat{\mathrm{v}}} - N_{\mathrm{v}}$ additional vectors allow to ensure that equation (8) holds without the need to store the snapshot databases $(\mathbf{S}_{\mathrm{u}}, \mathbf{S}_{\mathrm{u_{off}}})$. In Section 2.2.1, the extended reduced basis $\widehat{\mathbf{V}}$ and associated singular values $\widehat{\mathbf{\Sigma}}$ are initialized from the truncated SVD of $\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u_{off}}}$ using the tolerance ε_1. Now, assume that $\widehat{\mathbf{V}}_{\mathrm{old}}$ and $\widehat{\mathbf{\Sigma}}_{\mathrm{old}}$ are known and new snapshots $(\mathbf{S}_{\mathrm{u}}^{\mathrm{new}}, \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{new}})$ become available. Algorithm 3 presents the update of $(\widehat{\mathbf{V}}, \widehat{\mathbf{\Sigma}})$ and, as a result, of $(\mathbf{V}, \mathbf{\Sigma})$. In Line 1, $\widehat{\mathbf{V}}_{\mathrm{new}}$ and $\widehat{\mathbf{\Sigma}}_{\mathrm{new}}$ are computed from the truncated SVD of $\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u_{off}}}^{\mathrm{new}}$ using the tolerance ε_1. Next, $\widehat{\mathbf{V}}$ and $\widehat{\mathbf{\Sigma}}$ are efficiently updated in Lines 2-6 using the fast SVD update of Brand. If the matrix $\widehat{\mathbf{V}}$ is not unitary due to numerical error, then a re-orthogonalization step can be performed in Line 7. Finally, the first N_{v} columns (resp. rows and columns) of $\widehat{\mathbf{V}}$ (resp. $\widehat{\mathbf{\Sigma}}$) are extracted in Line 8-11 to obtain $\mathbf{V}$ (resp. $\mathbf{\Sigma}$). In this way, the right singular vectors of $\widehat{\mathbf{S}}$ do not need to be computed or stored, and only the left singular vectors $\widehat{\mathbf{V}}$ and singular values $\widehat{\mathbf{\Sigma}}$ are updated to save computational cost. In addition, the computational complexity of Algorithm 3 is dominated by the SVD of a $N_{\mathrm{dof}} \times N_{\mathrm{s_{new}}}$ matrix and the QR factorization of a $N_{\mathrm{dof}} \times N_{\hat{\mathrm{v}}}$ matrix, which can be efficiently computed using standard algorithms in sequential or parallel.

Algorithm 3 Update of the reduced basis

Input: $\widehat{\mathbf{V}}_{\text{old}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\hat{\text{v}}}}$, $\widehat{\mathbf{\Sigma}}_{\text{old}} \in \mathbb{R}^{N_{\hat{\text{v}}} \times N_{\hat{\text{v}}}}$, $\mathbf{S}_{\text{u}}^{\text{new}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{s}_{\text{new}}}}$, $\mathbf{S}_{\text{u}_{\text{off}}}^{\text{new}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{s}_{\text{new}}}}$, $\varepsilon_1 \in [0,1]$, $\varepsilon_2 \in [0,1]$, and $C_{\left\|\mathbf{S}_{\text{u}}^{\text{old}}\right\|_F^2} \geqslant 0$.

Output: $\mathbf{V} \in \mathbb{R}^{N_{\text{dof}} \times N_{\text{v}}}$, $\mathbf{\Sigma} \in \mathbb{R}^{N_{\text{v}} \times N_{\text{v}}}$, $\widehat{\mathbf{V}} \in \mathbb{R}^{N_{\text{dof}} \times \widehat{N}_{\text{v}}}$, $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{\widehat{N}_{\text{v}} \times \widehat{N}_{\text{v}}}$, and $C_{\|\mathbf{S}_{\text{u}}\|_F^2} \geqslant 0$.

1: Compute $\widehat{\mathbf{V}}_{\text{new}}$ and $\widehat{\mathbf{\Sigma}}_{\text{new}}$ by POD from $(\mathbf{S}_{\text{u}}^{\text{new}}, \mathbf{S}_{\text{u}_{\text{off}}}^{\text{new}})$ using the tolerance ε_1;
2: Set $\mathbf{M} = \widehat{\mathbf{V}}_{\text{new}} - \widehat{\mathbf{V}}_{\text{old}} \widehat{\mathbf{V}}_{\text{old}}^T \widehat{\mathbf{V}}_{\text{new}}$;
3: Compute the QR-decomposition of $\mathbf{M} = \mathbf{Q}_{\text{M}} \mathbf{R}_{\text{M}}$;
4: Set $\mathbf{K} = \begin{bmatrix} \widehat{\mathbf{\Sigma}}_{\text{old}} & \widehat{\mathbf{V}}_{\text{old}}^T \widehat{\mathbf{V}}_{\text{new}} \widehat{\mathbf{\Sigma}}_{\text{new}} \\ \mathbf{0} & \mathbf{R}_{\text{M}} \widehat{\mathbf{\Sigma}}_{\text{new}} \end{bmatrix}$;
5: Compute the SVD of $\mathbf{K} = \mathbf{\Phi}_{\text{K}} \mathbf{\Sigma}_{\text{K}} \mathbf{\Psi}_{\text{K}}^T$;
6: Set $\widehat{\mathbf{V}} = \begin{bmatrix} \widehat{\mathbf{V}}_{\text{old}} & \mathbf{Q}_{\text{M}} \end{bmatrix} \mathbf{\Phi}_{\text{K}}$ and $\widehat{\mathbf{\Sigma}} = \mathbf{\Sigma}_{\text{K}}$;
7: *Optional:* orthonormalize $\widehat{\mathbf{V}}$ and modify $\widehat{\mathbf{\Sigma}}$ using Algorithm 4;
8: Set $C_{\|\mathbf{S}_{\text{u}}\|_F^2} = C_{\left\|\mathbf{S}_{\text{u}}^{\text{old}}\right\|_F^2} + \|\mathbf{S}_{\text{u}}^{\text{new}}\|_F^2$;
9: Find the smallest integer N_{v} such that $\sum_{i=N_{\text{v}}+1}^{N_{\hat{\text{v}}}} \widehat{\sigma}_i^2 \leqslant \varepsilon_2^2 C_{\|\mathbf{S}_{\text{u}}\|_F^2}$;
10: Set $\mathbf{V}$ as the first N_{v} columns of $\widehat{\mathbf{V}}$;
11: Set $\mathbf{\Sigma}$ as the first N_{v} rows and columns of $\widehat{\mathbf{\Sigma}}$.

Algorithm 4 Orthonormalization process

Input: $\widehat{\mathbf{V}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\hat{\text{v}}}}$, and $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{N_{\hat{\text{v}}} \times N_{\hat{\text{v}}}}$.

Output: $\widehat{\mathbf{V}}_{\text{orth}} \in \mathbb{R}^{N_{\text{dof}} \times N_{\hat{\text{v}}}}$, and $\widehat{\mathbf{\Sigma}}_{\text{orth}} \in \mathbb{R}^{N_{\hat{\text{v}}} \times N_{\hat{\text{v}}}}$.

1: Compute the QR-decomposition of $\widehat{\mathbf{V}} = \mathbf{Q}_{\text{V}} \mathbf{R}_{\text{V}}$;
2: Set $\mathbf{N} = \mathbf{R}_{\text{V}} \widehat{\mathbf{\Sigma}}$;
3: Compute the SVD of $\mathbf{N} = \mathbf{\Phi}_{\text{N}} \mathbf{\Sigma}_{\text{N}} \mathbf{\Psi}_{\text{N}}^T$;
4: Set $\widehat{\mathbf{V}}_{\text{orth}} = \mathbf{Q}_{\text{V}} \mathbf{\Phi}_{\text{N}}$ and $\widehat{\mathbf{\Sigma}}_{\text{orth}} = \mathbf{\Sigma}_{\text{N}}$.

4.2. Update of the reduced mesh

Given new snapshots of the reduced-order quantities, the extended reduced basis $\widehat{\mathbf{U}}_{\text{old}}$ and associated singular values $\widehat{\mathbf{\Lambda}}_{\text{old}}$ are updated in the same way as $\widehat{\mathbf{V}}_{\text{old}}$ and $\widehat{\mathbf{\Sigma}}_{\text{old}}$ using Algorithm 3 in order to obtain $(\widehat{\mathbf{U}}, \widehat{\mathbf{\Lambda}})$ and therefore $(\mathbf{U}, \mathbf{\Lambda})$. The weights $\boldsymbol{\omega}$ are then solution to the NNLS problem (4) defined by $(\mathbf{U}, \mathbf{\Lambda})$ as explained in Section 2.2.3.

4.3. Update of the error indicator

The error indicator is finally updated in the same way as the ROM. Specifically, the reduced basis $\mathbf{W}$ is updated from new snapshots of the error using the procedure presented in Section 4.1, and the reduced mesh is updated from new snapshots of the reduced-order quantities using the procedure described in Section 4.2.

5. Numerical experiments

The performance of the proposed methods is assessed for parametrized, time-dependent, nonlinear problems based on the 1D Burger's equation and 2D compressible Euler equations. The first experiment seeks to evaluate the robustness and accuracy of the hybrid model, and more specifically, its ability to control the estimated error by switching between the HDM and ROM on the fly, while the second experiment aims at evaluating the reduction in the computational cost of the offline stage provided by the adaptive, hybrid approach. In the following, the accuracy of the hybrid model (resp. ROM) is assessed using the relative space-time error:

$$\sqrt{\frac{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta}) - \widehat{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2}{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta})\|_2^2}} \quad \left(\text{resp.} \quad \sqrt{\frac{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta}) - \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2^2}{\sum_{k=1}^{N_t} \|\mathbf{u}_k(\boldsymbol{\theta})\|_2^2}} \right),$$

and the reduction in computational cost is evaluated using the relative runtime:

$$\frac{\text{Runtime of the hybrid model}}{\text{Runtime of the HDM}} \quad \left(\text{resp.} \quad \frac{\text{Runtime of the ROM}}{\text{Runtime of the HDM}} \right).$$

5.1. One-dimensional Burgers' equation

The first experiment focuses on the numerical simulation of a single hump in 1D. The fluid velocity $u(x, t; \boldsymbol{\theta}) \in \mathbb{R}$ is governed by the Burgers' equation:

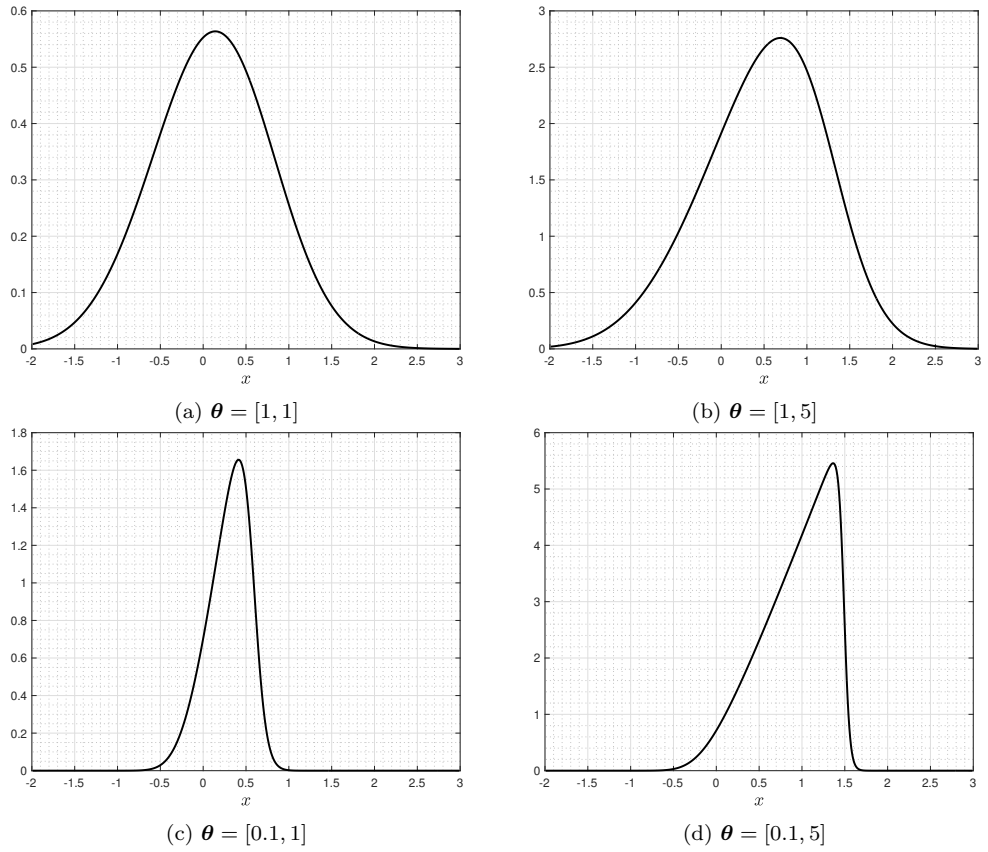
$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2},$$

where $x \in [-2, 3]$, $t \in [0, 0.2]$, and the input parameter $\boldsymbol{\theta} = [\nu, A]$ corresponds to the diffusion coefficient $\nu \in [0.1, 1]$ and a constant $A \in [1, 5]$. The initial and Dirichlet boundary conditions are supplied by the exact solution

$$u(x, t; \boldsymbol{\theta}) = \sqrt{\frac{\nu}{t+\epsilon}} \left[\frac{\left(e^{\frac{A}{2\nu}} - 1\right) e^{\frac{-x^2}{4\nu(t+\epsilon)}}}{\sqrt{\pi} + \frac{\sqrt{\pi}}{2}\left(e^{\frac{A}{2\nu}} - 1\right) \operatorname{erfc}(\frac{x}{\sqrt{4\nu(t+\epsilon)}})} \right],$$

where $\operatorname{erfc}(x)$ is the complimentary error function, and $\epsilon = 0.05$.

This equation is then discretized in the HDM by a second-order finite difference method. The time-interval is divided into $N_t = 200$ uniform subintervals using the fixed time-step size $\Delta t = 10^{-3}$, and the spatial domain is equally partitioned by 500 interior grid points, leading to $N_{\text{dof}} = 500$ degrees of freedom (DOFs). In Figure 1, we show solution snapshots computed by the HDM for different parameters $\boldsymbol{\theta}$.



(a) $\boldsymbol{\theta} = [1, 1]$ (b) $\boldsymbol{\theta} = [1, 5]$

(c) $\boldsymbol{\theta} = [0.1, 1]$ (d) $\boldsymbol{\theta} = [0.1, 5]$

Figure 1: High-dimensional solution at final time for different parameters $\boldsymbol{\theta}$

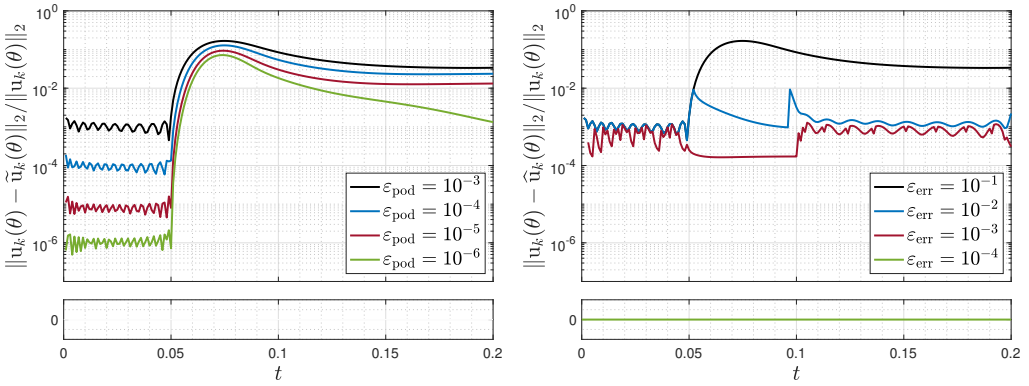
5.1.1. Performance of the hybrid model

The performance of the hybrid model is evaluated for predicting the solution corresponding to $\boldsymbol{\theta} = [0.1, 5]$. Specifically, solution snapshots are collected in the time-interval $[0, 0.05] \cup [0.1, 0.2]$, and the hybrid model must therefore predict the solution for $t \in (0.05, 0.1)$. Compared to a standard prediction test in time, where solution snapshots are collected at the beginning of the simulation (i.e., for $t \in [0, 0.05]$) and the hybrid model has to compute the solution for a larger time-interval (i.e., $[0, 0.1]$), solution snapshots are also collected in the time-interval $[0.1, 0.2]$. The objective is to show the ability of the hybrid model to maintain accurate approximations by switching from the ROM to the HDM when the error becomes too large (i.e. when $t \approx 0.05$) and also to check that the hybrid model is able to switch from the HDM to the ROM when the ROM becomes sufficient again to approximate

the solution (i.e., when $t \approx 0.1$).

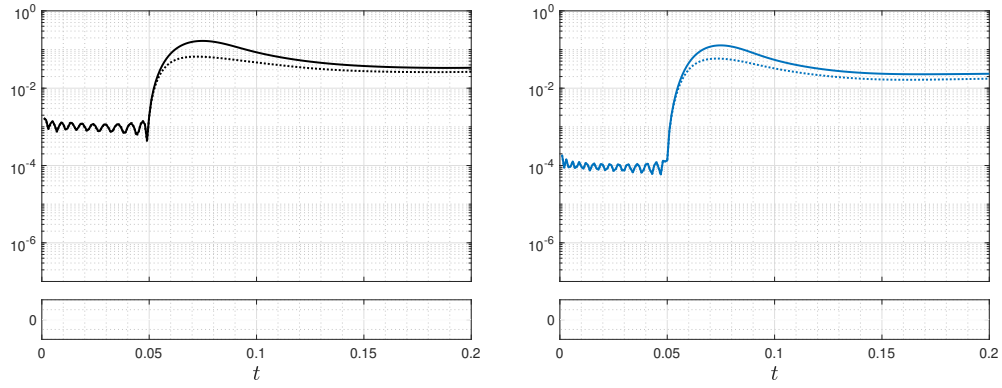
In Figure 2a, we compare the accuracy of four ROMs corresponding to different tolerances $\varepsilon_{\text{pod}} \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$. For constructing these ROMs, solution snapshots are collected every time-step, leading to $N_{\text{s}} = 150$ snapshots, and the hyper-reduction tolerance is chosen as $\varepsilon_{\text{ecsw}} = 10^{-6}$ to provide sufficient accuracy. The reader can see that the error increases for $t > 0.05$ since no snapshots were collected. Moreover, this error remains large even for small value of ε_{pod}, which means that the corresponding solution cannot be accurately approximated using only the snapshots available.

We now consider the hybrid model, where the underlying ROM is built using the tolerances $\varepsilon_{\text{pod}} = 10^{-3}$ and $\varepsilon_{\text{ecsw}} = 10^{-6}$, leading to $N_{\text{v}} = 19$ basis vectors and $N_{\text{p}} = 62$ points. In the error indicator, the size of the reduced basis and reduced mesh is the same than that of the ROM for simplicity. Figure 2b presents the accuracy of the hybrid model for different tolerances $\varepsilon_{\text{err}} \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. Compared to the previous ROMs, the solution is accurately approximated for every time-steps since the relative error remains generally smaller than the prescribed tolerance ε_{err}. The relative error is sometimes larger than ε_{err} because the error indicator underestimates the true error, as illustrated in Figure 3. Moreover, Figure 4 shows the evolution of the trade-off between the accuracy and computational cost of the hybrid model as a function of ε_{err}. As expected, when ε_{err} decreases, the error tends to decrease while the runtime increases, since the high-dimension system (1) is solved in a larger part of the computations. In particular, the hybrid model allows a transition from the HDM to the ROM corresponding to $\varepsilon_{\text{pod}} = 10^{-3}$ by varying the tolerance ε_{err} from 10^{-4} to 10^{-1}, and the runtime of the hybrid model is consistent with the computational cost given in equation (7). In addition, for $\varepsilon_{\text{err}} = 10^{-2}$, the high-dimension system (1) is solved only in the time-interval $(0.05, 0.1)$, which enables a speedup factor of 1.58 with respect to the HDM, while maintaining accurate approximations, as illustrated in Figure 5.



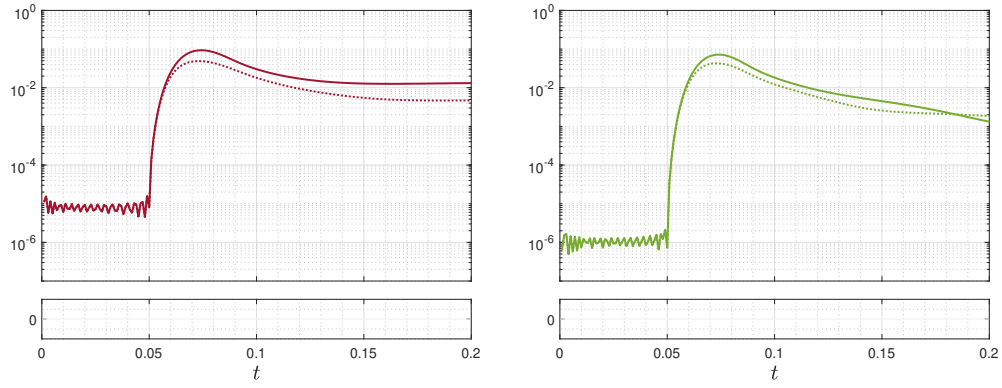
(a) ROM (b) Hybrid model

Figure 2: Accuracy of the ROM and hybrid model over time. On the left, we plot the relative error of ROMs constructed using different tolerances ε_{pod}, while the relative error of the hybrid model for different tolerances ε_{err} is plotted on the right



(a) $\varepsilon_{\text{pod}} = 10^{-3}$

(b) $\varepsilon_{\text{pod}} = 10^{-4}$



(c) $\varepsilon_{\text{pod}} = 10^{-5}$

(d) $\varepsilon_{\text{pod}} = 10^{-6}$

Figure 3: Time-history of the relative error $\|\mathbf{e}_k(\boldsymbol{\theta})\|_2 \,/\, \|\mathbf{u}_k(\boldsymbol{\theta})\|_2$ (solid line) and its estimation $\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2 \,/\, \|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2$ (dotted line) by the error indicator

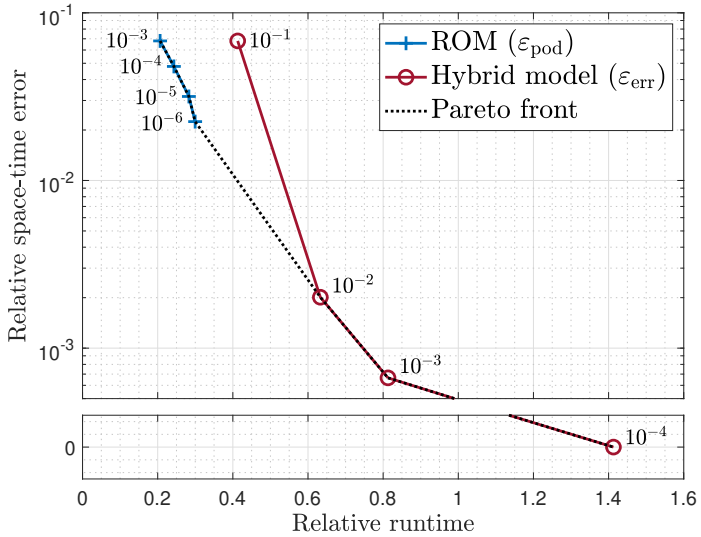
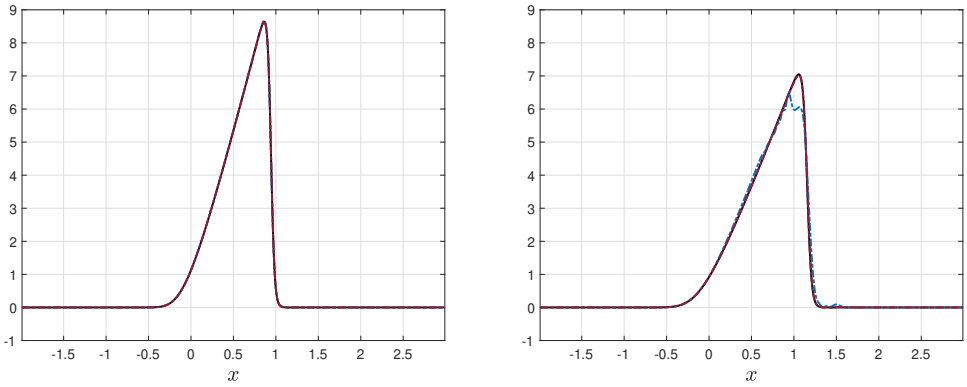
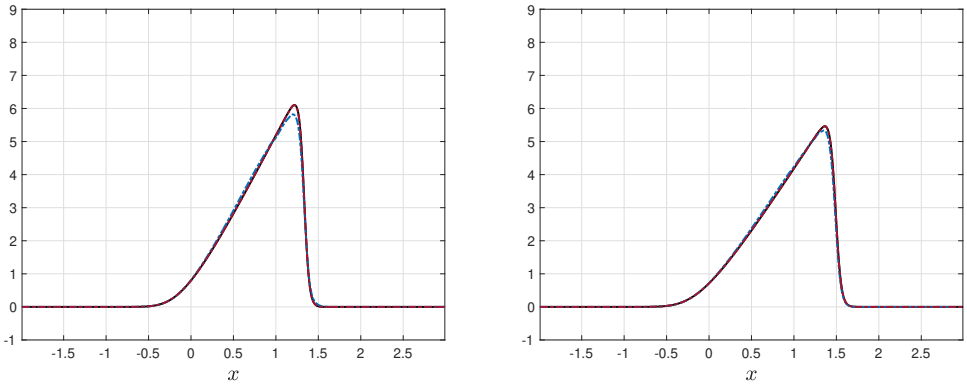


Figure 4: Performance of the hybrid model (resp. ROM) as a function of ε_{err} (resp. ε_{pod})



(a) $t = 0.05$

(b) $t = 0.1$



(c) $t = 0.15$

(d) $t = 0.2$

Figure 5: Solution snapshots of the HDM (black), ROM for $\varepsilon_{\text{pod}} = 10^{-3}$ (blue), and hybrid model for $\varepsilon_{\text{pod}} = 10^{-3}$ and $\varepsilon_{\text{err}} = 10^{-2}$ (red)

5.1.2. Accuracy of the error indicator

The accuracy of the proposed error indicator is further illustrated for predicting the error corresponding to different out-of-sample parameters $\boldsymbol{\theta}$. To this end, the ROM and associated error indicator are constructed using snapshots collected every time-step at the 5 training parameter instances shown in Figure 6, leading to $N_{\text{s}} = 200 \times 5 = 1,000$ snapshots. The reduced basis is then built using the tolerance $\varepsilon_{\text{pod}} = 10^{-3}$, which yields to $N_{\text{v}} = 32$ vectors, and the reduced mesh is extracted using the tolerance $\varepsilon_{\text{ecsw}} = 10^{-6}$, leading to $N_{\text{p}} = 105$ points. In the error indicator, the size of the reduced basis and reduced mesh is the same than that of the ROM to provide sufficient accuracy.

Figure 7 presents the relative error of the resulting ROM and its esti-

mation by the error indicator for the 20 out-of-sample parameters shown in Figure 6. In particular, there are two risks associated with error estimation. First, if the relative error $\|\mathbf{e}_k(\boldsymbol{\theta})\|_2 / \|\mathbf{u}_k(\boldsymbol{\theta})\|_2$ is smaller than the error tolerance ε_{err}, but the error indicator overestimates the relative error, and the estimated relative error $\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2 / \|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta}) + \widetilde{\mathbf{u}}_k(\boldsymbol{\theta})\|_2$ is larger than the error tolerance, then the hybrid model will employ the HDM whereas the ROM was sufficient in term of accuracy, resulting in unnecessary additional computation time. Second, if the relative error is larger than the error tolerance, but the error indicator underestimates the relative error, and the estimated relative error is smaller than the error tolerance, then the HDM will not be used while the ROM is not sufficiently accurate, which can lead to inaccurate solutions. However, despite the coarse training grid used for constructing the error indicator, the estimated error $\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2$ accurately approximates the true error $\|\mathbf{e}_k(\boldsymbol{\theta})\|_2$ for the different out-of-sample parameters. Furthermore, the predictions of the error indicator can be improved by enriching the database $\mathbf{S}_{\text{e}}$ with new snapshots of the error and by increasing the dimension of the reduced basis $\mathbf{W}$.

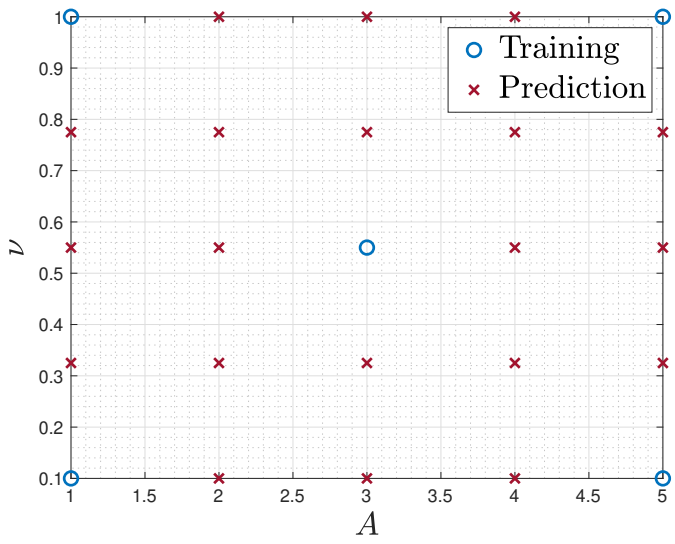


Figure 6: Sampling of the parameter domain $\mathcal{D}$. The training parameters are plotted in blue, while the out-of-sample parameters are plotted in red

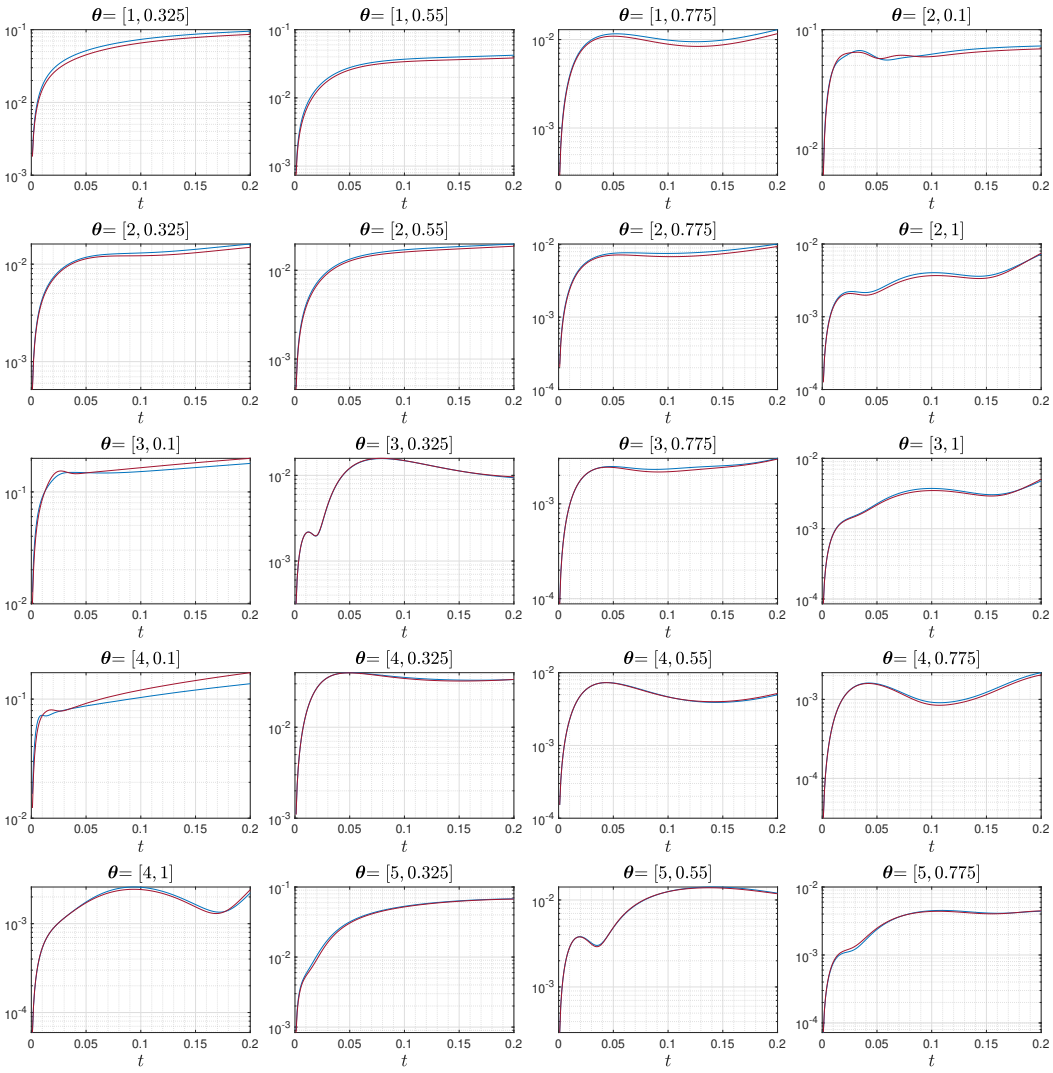


Figure 7: Time-history of the error norm $\|\mathbf{e}_k(\boldsymbol{\theta})\|_2$ (in blue) and its estimation $\|\widetilde{\mathbf{e}}_k(\boldsymbol{\theta})\|_2$ (in red) for different out-of-sample parameters $\boldsymbol{\theta}$

5.2. Two-dimensional compressible Euler equations

The second experiment concerns the numerical simulation of subsonic and transonic flows over a NACA 0012 airfoil in 2D. The input parameter $\boldsymbol{\theta} = [M_\infty, \alpha]$ corresponds to the free-stream Mach number $M_\infty \in [0.3, 0.8]$ and the angle of attack $\alpha \in [0, 8]$. The spatial domain is shown in Figure 8, and we consider the time-interval $[0, 10]$. The fluid density $\rho(\boldsymbol{x}, t; \boldsymbol{\theta}) \geq 0$, velocity $\boldsymbol{u}(\boldsymbol{x}, t; \boldsymbol{\theta}) = [u(\boldsymbol{x}, t; \boldsymbol{\theta}), v(\boldsymbol{x}, t; \boldsymbol{\theta})] \in \mathbb{R}^2$, and total energy $E(\boldsymbol{x}, t; \boldsymbol{\theta}) \geq 0$ are

governed by the compressible Euler equations:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial \rho u}{\partial x} + \dfrac{\partial \rho v}{\partial y} = 0 \\ \dfrac{\partial \rho u}{\partial t} + \dfrac{\partial (\rho u^2 + p)}{\partial x} + \dfrac{\partial \rho uv}{\partial y} = 0 \\ \dfrac{\partial \rho v}{\partial t} + \dfrac{\partial \rho uv}{\partial x} + \dfrac{\partial (\rho v^2 + p)}{\partial y} = 0 \\ \dfrac{\partial E}{\partial t} + \dfrac{\partial u(E+p)}{\partial x} + \dfrac{\partial v(E+p)}{\partial y} = 0, \end{cases}$$

where $p(\boldsymbol{x}, t; \boldsymbol{\theta}) \geq 0$ is the fluid pressure, given by the equation of state

$$p = (\gamma - 1)\left(E - \rho \frac{u^2 + v^2}{2}\right)$$

with γ, the specific heat ratio, taken as $\gamma = 1.4$ in the following. The initial condition is a uniform flow at Mach M_∞:

$$\rho_0 = \gamma, \quad u_0 = M_\infty, \quad v_0 = 0, \quad p_0 = 1. \tag{12}$$

The inflow and outflow boundary conditions are set to be a uniform flow at Mach M_∞ (12), and slip boundary conditions are applied at the airfoil surface.

The HDM is constructed using a first-order finite volume method equipped with the local Lax-Friedrichs flux in space and the implicit Euler scheme in time. The time-interval is divided into $N_t = 1,000$ uniform subintervals using the fixed time-step size $\Delta t = 10^{-2}$, and the spatial domain is partitioned by 1,718 vertices and 2,300 triangular cells, leading to $N_{\text{dof}} = 4 \times 2,300 = 9,200$ DOFs. In Figure 9, we show snapshots of the Mach number solution computed by the HDM for different parameters $\boldsymbol{\theta}$.

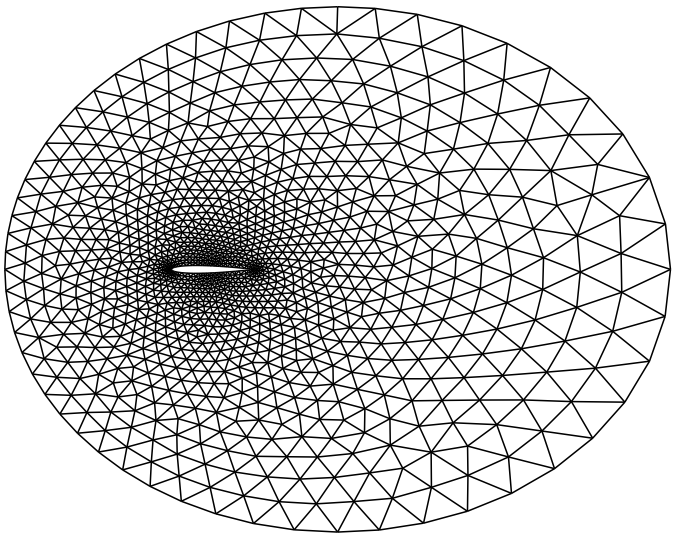
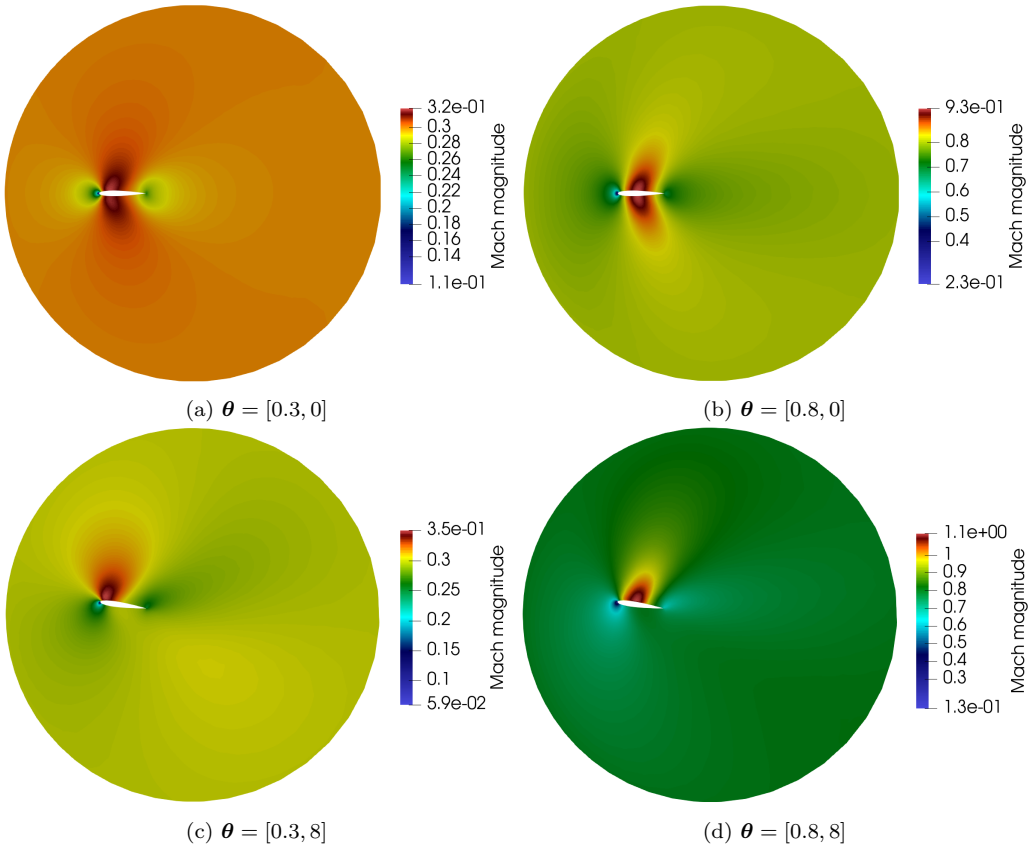


Figure 8: Computational domain in the reference configuration (i.e., $\alpha = 0$). To account for the variation in the angle of attack, the mesh is rotated through an angle α about the trailing edge



(a) $\boldsymbol{\theta} = [0.3, 0]$ (b) $\boldsymbol{\theta} = [0.8, 0]$

(c) $\boldsymbol{\theta} = [0.3, 8]$ (d) $\boldsymbol{\theta} = [0.8, 8]$

Figure 9: High-dimensional solution at final time for different parameters $\boldsymbol{\theta}$

5.2.1. Performance of the adaptive, hybrid approach

The reduction in the computational cost of the offline stage provided by the adaptive, hybrid model is assessed for sampling the solution at the 25 training parameter instances shown in Figure 10. To this end, we first compare the runtime of the hybrid model with that of the classical approach, where solution snapshots are computed using only the HDM. For initializing the hybrid model, solution snapshots are collected every 4 time-steps using the HDM at the 5 training parameter instances shown in Figure 10, leading to $N_\mathrm{s} = 250 \times 5 = 1,250$ snapshots. The reduced basis is then built using the tolerance $\varepsilon_\mathrm{pod} = 5 \times 10^{-3}$ and the reduced mesh is extracted using the tolerance $\varepsilon_\mathrm{ecsw} = 10^{-6}$. In the error indicator, the size of the reduced basis and reduced mesh is taken as $(2N_\mathrm{v}, 2N_\mathrm{p})$ to provide sufficient accuracy. Furthermore, the error indicator is employed in the following to select iteratively

the solution to be sampled by identifying the training parameter associated with the maximum error.

When the classical approach is used for sampling solution snapshots, the offline stage takes 2.39 hours and results in $N_{\mathrm{v}} = 14$ basis vectors and $N_{\mathrm{p}} = 204$ points. When the hybrid model is employed, the offline stage lasts 44.53 minutes, which corresponds to 28.7 minutes for sampling the 5 initial parameters and 15.83 minutes for sampling the 20 other training parameters. Moreover, the dimension of the resulting reduced basis is $N_{\mathrm{v}} = 16$ and the size of the reduced mesh is $N_{\mathrm{p}} = 260$. The hybrid model therefore provides a speedup factor of 3.22, and the dimension of the ROMs $(N_{\mathrm{v}}, N_{\mathrm{p}})$ obtained using the adaptive approach is slightly higher than that obtained using the classical approach as shown in Figure 11.

Finally, Figure 12 presents the performance of the ROMs resulting from the classical and adaptive approaches. The reader can see that the accuracy of these ROMs is also almost the same for predicting the solution at the 16 out-of-sample parameters shown in Figure 10. Furthermore, the computational cost of the ROM resulting from the adaptive approach is slightly higher than that resulting from the classical approach, because the dimension $(N_{\mathrm{v}}, N_{\mathrm{p}})$ of the latter is slightly lower.

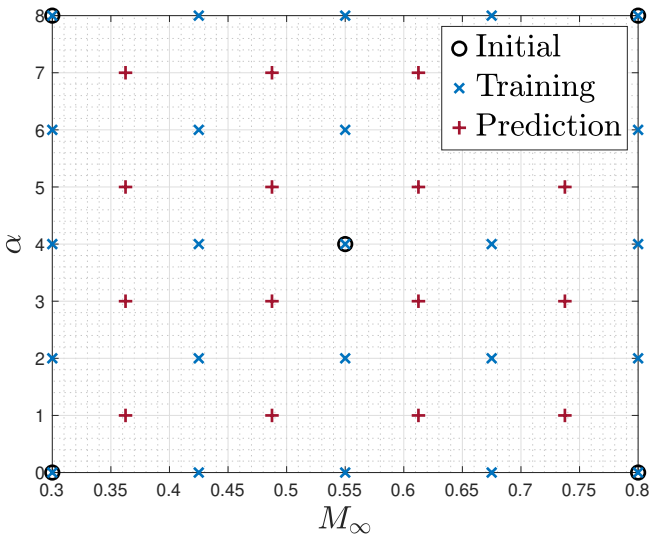
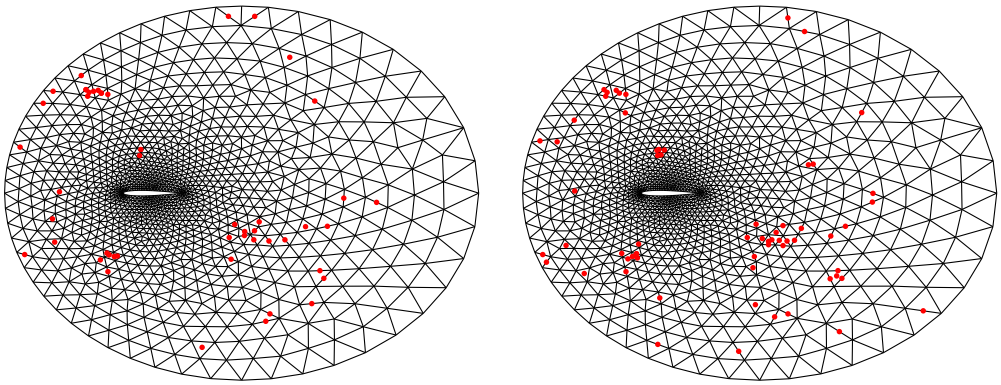


Figure 10: Sampling of the parameter domain $\mathcal{D}$. The parameters used to initialize the hybrid model are plotted in black, the training parameters are plotted in blue, and the out-of-sample parameters are plotted in red



(a) Classical approach (b) Adaptive approach

Figure 11: Comparaison of the reduced meshes obtained using the classical and adaptive approaches

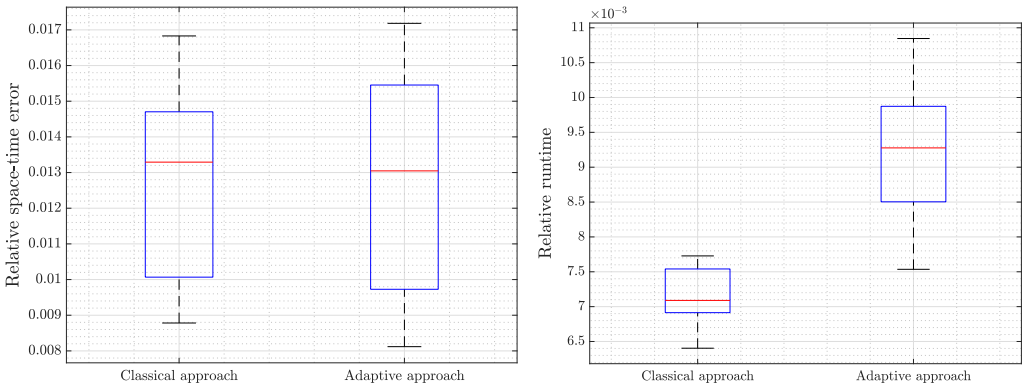


Figure 12: Performance of the ROMs resulting from the classical and adaptive approaches

6. Conclusion

In this paper, we have proposed an approach based on the principal directions of the reduced-order residual and reduced-order Jacobian matrix to improve the performance of the ECSW method. Then, we have presented a hybrid HDM/ROM strategy to accelerate numerical simulations while maintaining accurate approximations. In particular, we have developed an error indicator based on a reduced-order approximation of the error to determine when the ROM is not sufficiently accurate and the HDM needs to be solved. Finally, we have proposed an adaptive-extended version of the hybrid model to update the ROM with new solution snapshots.

The performance of the proposed methods has been evaluated for parametrized, time-dependent, nonlinear problems based on the 1D Burger's equation and 2D compressible Euler equations. The results demonstrated the accuracy and robustness of the hybrid model, capable of controlling the estimated error by switching between the HDM and ROM on the fly, and the significant computational speedup factor delivered by the adaptive, hybrid approach to accelerate the offline stage of ROMs.

In perspective, the performance of the hybrid model could be further improved by using ROMs based on nonlinear manifolds [26, 27, 28]. In addition, the computational cost associated with the HDM solution could also be reduced by considering hybrid snapshots. The latter are generated by employing the HDM in regions where the ROM is not sufficiently accurate, and space-local ROMs [29, 30, 31] elsewhere. These topics will be addressed in future work.

Appendix A. Construction of the approximation problem (3)

The objective of this section is to describe the construction of the approximation problem (3). Let the training time-parameter instances be $\{(t_1^{\text{train}}, \boldsymbol{\theta}_1^{\text{train}}), \ldots, (t_{N_\text{s}}^{\text{train}}, \boldsymbol{\theta}_{N_\text{s}}^{\text{train}})\}$. We start to present different strategies for constructing $\mathbf{C}$ and consequently $\mathbf{d}$, since $\mathbf{d} = \mathbf{C}\mathbf{1}$.

Residual. The evaluation of the projected reduced-order residual $\overline{\mathbf{J}}_k(\boldsymbol{\theta})^T \overline{\mathbf{r}}_k(\boldsymbol{\theta})$ leads to the approximation problem:

$$\mathbf{C}_1 \boldsymbol{\omega} \approx \mathbf{d}_1, \tag{A.1}$$

where

$$\mathbf{C}_1 = \begin{bmatrix} \mathbf{C}_1^{(1)} \\ \vdots \\ \mathbf{C}_1^{(N_\text{s})} \end{bmatrix} \in \mathbb{R}^{(N_\text{s} N_\text{v}) \times N_\text{dof}} \quad \text{and} \quad \mathbf{C}_1^{(k)} \in \mathbb{R}^{N_\text{v} \times N_\text{dof}}$$

with $\left(\mathbf{C}_1^{(k)}\right)_{i,j} = \left(\overline{\mathbf{J}}_k(\boldsymbol{\theta})\right)_{j,i} \left(\overline{\mathbf{r}}_k(\boldsymbol{\theta})\right)_j$ and $(t_k, \boldsymbol{\theta}) = (t_k^{\text{train}}, \boldsymbol{\theta}_k^{\text{train}})$.

Jacobian. In the same way, the evaluation of the projected reduced-order Jacobian matrix $\overline{\mathbf{J}}_k(\boldsymbol{\theta})^T \overline{\mathbf{J}}_k(\boldsymbol{\theta})$ leads to the approximation problem:

$$\mathbf{C}_2 \boldsymbol{\omega} \approx \mathbf{d}_2, \tag{A.2}$$

where

$$\mathbf{C}_2 = \begin{bmatrix} \mathbf{C}_2^{(1)} \\ \vdots \\ \mathbf{C}_2^{(N_\mathrm{s})} \end{bmatrix} \in \mathbb{R}^{(N_\mathrm{s}N_\mathrm{v}(N_\mathrm{v}+1)/2)\times N_\mathrm{dof}} \quad \text{and} \quad \mathbf{C}_2^{(k)} \in \mathbb{R}^{(N_\mathrm{v}(N_\mathrm{v}+1)/2)\times N_\mathrm{dof}}$$

with $\left(\mathbf{C}_2^{(k)}\right)_{i_1+i_2(i_2-1)/2,j} = \left(\overline{\mathbf{J}}_k(\boldsymbol{\theta})\right)_{j,i_1} \left(\overline{\mathbf{J}}_k(\boldsymbol{\theta})\right)_{j,i_2}$ and $(t_k, \boldsymbol{\theta}) = (t_k^{\mathrm{train}}, \boldsymbol{\theta}_k^{\mathrm{train}})$ for $1 \leq i_1 \leq i_2 \leq N_\mathrm{v}$. Note that the symmetry of the problem has been exploited to remove duplicate rows in $\mathbf{C}_2$.

Residual+Jacobian. Combining the contributions of (A.1) and (A.2) leads to the approximation problem:

$$\mathbf{C}_3\boldsymbol{\omega} \approx \mathbf{d}_3, \tag{A.3}$$

where

$$\mathbf{C}_3 = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \end{bmatrix} \in \mathbb{R}^{(N_\mathrm{s}N_\mathrm{v}(N_\mathrm{v}+3)/2)\times N_\mathrm{dof}}.$$

Principal directions. In practice, the number of rows in $\mathbf{C}_3$ can quickly become prohibitive for solving the NNLS problem (4). In order to drastically reduce the number of rows, we replace $\mathbf{C}_3$ by its principal directions. Let $\mathbf{C}_3 \approx \boldsymbol{\Phi}_\mathrm{C}\boldsymbol{\Sigma}_\mathrm{C}\boldsymbol{\Psi}_\mathrm{C}^T$ be the truncated SVD of $\mathbf{C}_3$, where $\boldsymbol{\Phi}_\mathrm{C} \in \mathbb{R}^{(N_\mathrm{s}N_\mathrm{v}(N_\mathrm{v}+3)/2)\times N_\mathrm{c}}$ is unitary, $\boldsymbol{\Sigma}_\mathrm{C} \in \mathbb{R}^{N_\mathrm{c}\times N_\mathrm{c}}$ is diagonal, $\boldsymbol{\Psi}_\mathrm{C} \in \mathbb{R}^{N_\mathrm{dof}\times N_\mathrm{c}}$ is unitary, and the rank N_c is defined as the smallest integer such that the relative approximation error is less than the tolerance ε_pod (i.e., $\left\|\mathbf{C}_3 - \boldsymbol{\Phi}_\mathrm{C}\boldsymbol{\Sigma}_\mathrm{C}\boldsymbol{\Psi}_\mathrm{C}^T\right\|_F \leqslant \varepsilon_\mathrm{pod} \left\|\mathbf{C}_3\right\|_F$). The approximation problem (A.3) becomes

$$\mathbf{C}_4\boldsymbol{\omega} \approx \mathbf{d}_4, \tag{A.4}$$

where

$$\mathbf{C}_4 = \boldsymbol{\Psi}_\mathrm{C}^T \in \mathbb{R}^{N_\mathrm{c}\times N_\mathrm{dof}} \quad \text{and} \quad N_\mathrm{c} \ll N_\mathrm{s}N_\mathrm{v}(N_\mathrm{v}+3)/2.$$

Weighted principal directions. In the approximation problem (A.4), the principal directions contribute equally to the approximation error, which may not be representative of the original approximation problem (A.3). For this reason, we introduce a weighting of the principal directions to approximately recover the original problem (A.3):

$$\begin{aligned} \left\|\mathbf{C}_3\boldsymbol{\omega} - \mathbf{d}_3\right\|_2 &\approx \left\|\boldsymbol{\Phi}_\mathrm{C}\boldsymbol{\Sigma}_\mathrm{C}\boldsymbol{\Psi}_\mathrm{C}^T\boldsymbol{\omega} - \mathbf{d}_3\right\|_2 \\ &= \left\|\boldsymbol{\Phi}_\mathrm{C}\boldsymbol{\Sigma}_\mathrm{C}(\mathbf{C}_4\boldsymbol{\omega} - \mathbf{d}_4)\right\|_2 \\ &= \left\|\boldsymbol{\Sigma}_\mathrm{C}(\mathbf{C}_4\boldsymbol{\omega} - \mathbf{d}_4)\right\|_2. \end{aligned}$$

In this way, the approximation problem (A.4) becomes

$$\mathbf{C}_5 \boldsymbol{\omega} \approx \mathbf{d}_5, \tag{A.5}$$

where

$$\mathbf{C}_5 = \boldsymbol{\Sigma}_{\mathrm{C}} \mathbf{C}_4 \in \mathbb{R}^{N_{\mathrm{c}} \times N_{\mathrm{dof}}}.$$

Numerical results. The different strategies for constructing $\mathbf{C}$ are now evaluated for the application presented in Section 5.2. In each case, we report the size of the matrix $\mathbf{C}$. Moreover, Figure A.13 shows the runtime for solving the NNLS problem (4) and the accuracy of the ROM associated with the resulting weights $\boldsymbol{\omega}$, as a function of the reduced mesh size.

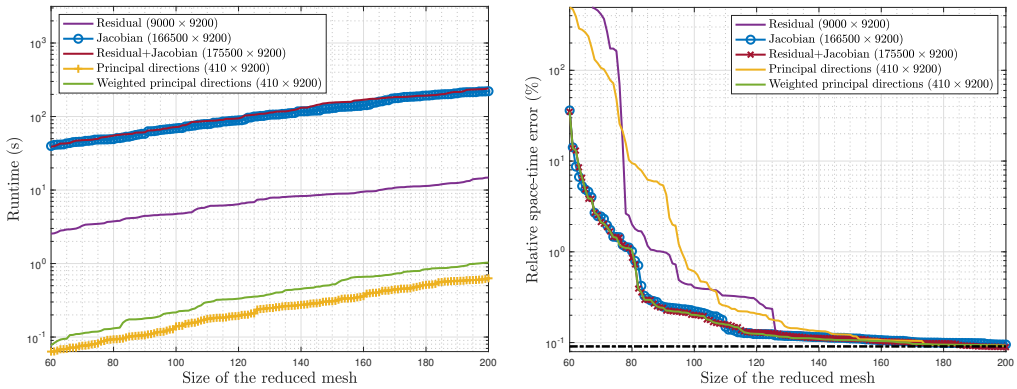


Figure A.13: Results for the application presented in Section 5.2. We consider the reproduction of the solution corresponding to the input parameter $\boldsymbol{\theta} = [0.8, 8]$. Solution snapshots are collected every 4 time-steps, leading to $N_s = 250$ snapshots. The reduced basis is built using the tolerance $\varepsilon_{\mathrm{pod}} = 10^{-3}$, which yields to $N_{\mathrm{v}} = 36$ basis vectors.

The runtime of the "principal directions" and "weighted principal directions" approaches are almost the same and are significantly lower than that of the other approaches, as the number of rows of the corresponding matrix $\mathbf{C}$ is drastically smaller. In addition, the error of the "jacobian", "residual+jacobian", and "weighted principal direction" approaches are about the same and are generally lower than that of the other approaches, since the "residual+jacobian" approach contains all the available information for computing the weights $\boldsymbol{\omega}$. For these reasons, the "weighted principal direction" approach is used in this work, as it provides the most accurate ROM, while drastically reducing the runtime to solve the NNLS problem (4).

Appendix B. Proof of error bound (11)

Proposition 1. *Assume that equations* (9) *and* (10) *hold. Then, the reduced basis* $\mathbf{V}$ *verifies the error bound* (11)*.*

Proof. Given (9a), we first prove the inequality

$$
\begin{aligned}
&\left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}})\right\|_F^2 \\
&\quad = \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \widehat{\mathbf{S}}_{\mathrm{old}} + \widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}}) + \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 \\
&\quad \leqslant \left\|(\mathbf{I} - \mathbf{V}\mathbf{V}^T)(\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \widehat{\mathbf{S}}_{\mathrm{old}})\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 \\
&\quad \leqslant \left\|\mathbf{I} - \mathbf{V}\mathbf{V}^T\right\|_2^2 \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 \\
&\quad \leqslant \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 \\
&\quad \leqslant \varepsilon_1^2 \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}}\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2,
\end{aligned}
$$

since $\left\|\mathbf{I} - \mathbf{V}\mathbf{V}^T\right\|_2 = \left\|\mathbf{V}\mathbf{V}^T\right\|_2 = 1$ for any projector $\mathbf{V}\mathbf{V}^T \neq \mathbf{0}$ or $\mathbf{I}$ (see [32]). Similarly, we can show using (9b) that

$$
\left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{new}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{new}})\right\|_F^2 \leqslant \varepsilon_1^2 \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{new}}\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{new}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{new}}\right\|_F^2.
$$

Finally, combining these two inequalities with assumption (10) yields

$$
\begin{aligned}
&\left\|\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}})\right\|_F^2 \\
&\quad = \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}}^{\mathrm{old}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{old}})\right\|_F^2 + \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{new}} - \mathbf{V}\mathbf{V}^T(\mathbf{S}_{\mathrm{u}}^{\mathrm{new}} - \mathbf{S}_{\mathrm{u}_{\mathrm{off}}}^{\mathrm{new}})\right\|_F^2 \\
&\quad \leqslant \varepsilon_1^2(\left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{old}}\right\|_F^2 + \left\|\mathbf{S}_{\mathrm{u}}^{\mathrm{new}}\right\|_F^2) + \left\|\widehat{\mathbf{S}}_{\mathrm{old}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{old}}\right\|_F^2 + \left\|\widehat{\mathbf{S}}_{\mathrm{new}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}_{\mathrm{new}}\right\|_F^2 \\
&\quad = \varepsilon_1^2 \left\|\mathbf{S}_{\mathrm{u}}\right\|_F^2 + \left\|\widehat{\mathbf{S}} - \mathbf{V}\mathbf{V}^T\widehat{\mathbf{S}}\right\|_F^2 \\
&\quad \leqslant (\varepsilon_1^2 + \varepsilon_2^2) \left\|\mathbf{S}_{\mathrm{u}}\right\|_F^2,
\end{aligned}
$$

which concludes the proof. □

References

[1] B. Moore, Principal component analysis in linear systems: Controllability, observability, and model reduction, IEEE transactions on automatic control 26 (1) (1981) 17–32.

[2] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual review of fluid mechanics 25 (1) (1993) 539–575.

[3] K. Willcox, J. Peraire, Balanced model reduction via the proper orthogonal decomposition, AIAA journal 40 (11) (2002) 2323–2330.

[4] C. W. Rowley, T. Colonius, R. M. Murray, Model reduction for compressible flows using POD and Galerkin projection, Physica D: Nonlinear Phenomena 189 (1-2) (2004) 115–129.

[5] N. Ngoc Cuong, K. Veroy, A. T. Patera, Certified real-time solution of parametrized partial differential equations, Handbook of Materials Modeling: Methods (2005) 1529–1564.

[6] M. Bergmann, C.-H. Bruneau, A. Iollo, Enablers for robust POD models, Journal of Computational Physics 228 (2) (2009) 516–538.

[7] K. Carlberg, C. Farhat, J. Cortial, D. Amsallem, The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows, Journal of Computational Physics 242 (2013) 623–647.

[8] A. Iollo, D. Lombardi, Advection modes by optimal mass transfer, Physical Review E 89 (2) (2014) 022923.

[9] F. Bernard, A. Iollo, S. Riffaud, Reduced-order model for the BGK equation based on POD and optimal transport, Journal of Computational Physics 373 (2018) 545–570.

[10] L. Sirovich, Turbulence and the dynamics of coherent structures. I. Coherent structures, Quarterly of applied mathematics 45 (3) (1987) 561–571.

[11] L. W. Ng, K. E. Willcox, Multifidelity approaches for optimization under uncertainty, International Journal for numerical methods in Engineering 100 (10) (2014) 746–772.

[12] B. Peherstorfer, K. Willcox, M. Gunzburger, Optimal model management for multifidelity Monte Carlo estimation, SIAM Journal on Scientific Computing 38 (5) (2016) A3163–A3194.

[13] F. Bai, Y. Wang, DEIM reduced order model constructed by hybrid snapshot simulation, SN Applied Sciences 2 (12) (2020) 2165.

[14] F. Bai, Y. Wang, Reduced-order modeling based on hybrid snapshot simulation, International Journal of Computational Methods 18 (01) (2021) 2050029.

[15] L. Feng, G. Fu, Z. Wang, A FOM/ROM hybrid approach for accelerating numerical simulations, Journal of Scientific Computing 89 (2021) 1–16.

[16] V. Zucatti, M. J. Zahr, An adaptive, training-free reduced-order model for convection-dominated problems based on hybrid snapshots, arXiv preprint arXiv:2301.01718 (2023).

[17] T. Bui-Thanh, K. Willcox, O. Ghattas, Model reduction for large-scale systems with high-dimensional parametric input space, SIAM Journal on Scientific Computing 30 (6) (2008) 3270–3288.

[18] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations, International Journal for numerical methods in engineering 86 (2) (2011) 155–181.

[19] C. Farhat, P. Avery, T. Chapman, J. Cortial, Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency, International Journal for Numerical Methods in Engineering 98 (9) (2014) 625–662.

[20] C. Farhat, T. Chapman, P. Avery, Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models, International journal for numerical methods in engineering 102 (5) (2015) 1077–1110.

[21] R. Tezaur, F. As'ad, C. Farhat, Robust and globally efficient reduction of parametric, highly nonlinear computational models and real time online performance, Computer Methods in Applied Mechanics and Engineering 399 (2022) 115392.

[22] C. Eckart, G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1 (3) (1936) 211–218.

[23] C. L. Lawson, R. J. Hanson, Solving least squares problems, SIAM, 1995.

[24] D. Amsallem, M. J. Zahr, K. Washabaugh, Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction, Advances in Computational Mathematics 41 (2015) 1187–1230.

[25] M. Brand, Fast low-rank modifications of the thin singular value decomposition, Linear algebra and its applications 415 (1) (2006) 20–30.

[26] K. Lee, K. T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, Journal of Computational Physics 404 (2020) 108973.

[27] S. Fresca, A. Manzoni, POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, Computer Methods in Applied Mechanics and Engineering 388 (2022) 114181.

[28] J. Barnett, C. Farhat, Y. Maday, Neural-network-augmented projection-based model order reduction for mitigating the Kolmogorov barrier to reducibility, Journal of Computational Physics 492 (2023) 112420.

[29] S. Riffaud, M. Bergmann, C. Farhat, S. Grimberg, A. Iollo, The DGDD method for reduced-order modeling of conservation laws, Journal of Computational Physics 437 (2021) 110336.

[30] S. Anderson, C. White, C. Farhat, Space-local reduced-order bases for accelerating reduced-order models through sparsity, International Journal for Numerical Methods in Engineering 124 (7) (2023) 1646–1671.

[31] A. Iollo, G. Sambataro, T. Taddei, A one-shot overlapping Schwarz method for component-based model reduction: application to nonlinear elasticity, Computer Methods in Applied Mechanics and Engineering 404 (2023) 115786.

[32] D. B. Szyld, The many proofs of an identity on the norm of oblique projections, Numerical Algorithms 42 (2006) 309–323.