Progressive construction of a parametric reduced-order model for PDE-constrained optimization

Matthew J. Zahr [1*] and Charbel Farhat [1,2,3]

[1] *Institute for Computational and Mathematical Engineering*
[2] *Department of Aeronautics and Astronautics*
[3] *Department of Mechanical Engineering*
Stanford University, Mail Code 4035, Stanford, CA 94305, U.S.A.

SUMMARY

An adaptive approach to using reduced-order models as surrogates in PDE-constrained optimization is introduced that breaks the traditional offline-online framework of model order reduction. A sequence of optimization problems constrained by a given Reduced-Order Model (ROM) is defined with the goal of converging to the solution of a given PDE-constrained optimization problem. For each reduced optimization problem, the constraining ROM is trained from sampling the High-Dimensional Model (HDM) at the solution of some of the previous problems in the sequence. The reduced optimization problems are equipped with a nonlinear trust-region based on a residual error indicator to keep the optimization trajectory in a region of the parameter space where the ROM is accurate. A technique for incorporating sensitivities into a Reduced-Order Basis (ROB) is also presented, along with a methodology for computing sensitivities of the reduced-order model that minimizes the distance to the corresponding HDM sensitivity, in a suitable norm. The proposed reduced optimization framework is applied to subsonic aerodynamic shape optimization and shown to reduce the number of queries to the HDM by a factor of 4-5, compared to the optimization problem solved using only the HDM, with errors in the optimal solution far less than 0.1%.
Copyright © 0000 John Wiley & Sons, Ltd.

1. INTRODUCTION

Optimization problems constrained by nonlinear Partial Differential Equations (PDE) arise in many engineering applications, including inverse modeling, optimal control, and design. In the context of design, PDE-constrained optimization provides a valuable tool for optimizing specific features of an existing design. In practice, the solution of the optimization problem will require many solutions of the underlying PDE, which will be expensive when the underlying PDE model is large. Large-scale PDE models commonly arise in industry-scale applications where high-fidelity simulations are required due to the complexity of the underlying physics. In particular, large-scale, nonlinear PDE models arise in Computational Mechanics (CM) simulations, such as viscous or high-speed Computational Fluid Dynamics (CFD), large-deformation Computational Structural Dynamics (CSD) of complex bodies, and high-frequency acoustics, to name a few.

---

*Correspondence to: Department of Aeronautics and Astronautics, Stanford University, Mail Code 4035, Stanford, CA 94305, U.S.A.

In practice, PDE-constrained optimization problems are solved using a Nested Analysis and Design (NAND) [1] approach, whereby the PDE state variable is considered an implicit function of the parameters of the PDE. The implication of this approach is that the PDE must be solved at every parameter value encountered in the design cycle. It is well-known that this approach requires a large number of PDE solutions, and sensitivities in a gradient-based approach, which is prohibitively expensive for industry-scale PDE models. Accordingly, the main objective of this paper is to present a method for finding an approximate solution to the PDE-constrained optimization problem at a reduced computational cost. This method relies on using a sequence of low-dimensional reduced-order models as surrogates for the underlying high-dimensional model.

Model Order Reduction (MOR) has become an attractive method for reducing the dimension of a PDE by constraining the solution to lie in a low-dimensional affine subspace. MOR has found the most success in real-time applications, such as optimal control [2, 3, 4, 5, 6, 7, 8] and non-destructive evaluation [9], and many-query applications [10, 11], such as design optimization [12, 13, 14, 15] and uncertainty quantification [16]. Model reduction is often embedded in an offline-online framework, whereby a low-dimensional Reduced-Order Basis (ROB) is constructed in the *non-time-critical* offline phase. The span of the ROB defines the subspace in which perturbations of the PDE solution from a chosen offset will be constrained to lie. Subsequently, the ROB is used to project the governing equations, in either a Galerkin or Petrov-Galerkin sense, onto a low-dimensional test subspace. The reduced equations and solution space collectively define the reduced-order model, which can be used in the *time-critical* online phase.

In this work, projection-based ROMs are considered with bases built using a variant of Proper Orthogonal Decomposition (POD) and the method of snapshots [17]. In this approach, information is collected from simulations of the HDM at various parameter configurations and compressed to generate the ROB. The data collected from the HDM simulation are usually state vectors (time history or steady state), but could be any quantity that provides information about the subspace on which the solution lies, such as state sensitivities, residuals, or Krylov vectors [18]. In the remainder, a parameter value at which a HDM is simulated to generate snapshots will be referred to as a *HDM sample*.

A well-known bottleneck in the online phase of projection-based ROMs of nonlinear systems is evaluation and projection of high-dimensional nonlinear quantities that cannot be precomputed in the offline phase. Only in the case of polynomial nonlinearities can all high-dimensional operations be confined to the offline phase. In the general case, despite the reduced dimensionality of the model, in the sense of a reduced set of equations and unknowns, the computation time is not necessarily reduced, just redistributed. Hyperreduction, a term introduced in [19], has been developed as an additional level of approximation built on top of the ROM to realize computational speedups in addition to dimensionality reduction. Many different flavors of hyperreduction exist [20, 21, 22, 23, 24], some of which are variants of Gappy POD [25] and have been shown to demonstrate promising (CPU time) speedup factors in excess of 400 in computing the unsteady solution of a large-scale CFD problem [23]. The method proposed in this work will only be applied at the level of the ROM; extensions to hyperreduction will be mentioned throughout, with complete formulation deferred to future work.

The standard offline-online [12, 26, 27, 13, 14, 15, 28, 29] decomposition of computational effort for reduced-order modeling is most appropriate in the real-time application context, where there is a natural offline-online breakdown and minimizing online computing time is paramount. In the context of many-query analyses, particularly design optimization, the offline-online decomposition is artificial in that the only interest is total wall-time to obtain an accurate solution. This implies the existence is a "break-even point" [30, 26, 31, 32, 33], an upper limit on the offline time, beyond which, it cannot be amortized in the online phase. Furthermore, it is well-known that ROMs lack robustness with respect to parametric variations [34, 35, 36, 37, 38, 39]. The assumption that the PDE solution can be represented by a single, static, low-dimensional subspace will break-down, particularly for nonlinear problems where the solution can change dramatically with parameter perturbations, such as high-speed or turbulent (chaotic) fluid flows. As the trajectory of the optimization iterates is unknown a-priori, a ROB must be trained such that it maintains accuracy

throughout the parameter space if the offline-online decomposition is used. The implication is that a large ROB built from extensive training with the HDM will be required. Despite the extensive training, only the portion of the HDM samples near the optimization trajectory will be relevant.

Reduced-order models have been used as surrogates in PDE-constrained optimization in two main contexts: offline-online framework [12, 26, 27, 13, 15, 28, 29, 40], where efficient HDM sampling [27, 28, 29] and ROB construction is paramount and an adaptive framework that merges the offline and online phases [3, 4, 5, 6, 7, 8, 41, 42]. The adaptive approaches to ROM-constrained optimization involve a sequence of reduced optimization problems, where the solution to a given reduced optimization problem is used to enrich the training, resulting in an improved basis for subsequent reduced optimization problems. Variations include trust region methods that treat the ROM objective as the model problem instead of the traditional quadratic model problem – Trust-Region Proper Orthogonal Decomposition (TRPOD) [5], using the gradient of the HDM to take a single optimization step before enriching the ROB – Optimality System Proper Orthogonal Decomposition (OS-POD) [8], and equipping the reduced optimization problem with a constraint (or penalty) on the error in the ROM, in the context of linear systems [41]. Additionally, in [42], an adaptive approach was introduced that defined a sequence of reduced optimization problems that balanced minimizing the objective function with targeting locations in the parameter space where the ROM lacks accuracy, i.e. a desirable sample point.

The proposed approach attempts to combine the desirable features of TRPOD, OS-POD, and the error-aware reduced optimization approach for general nonlinear problems, cognizant of the reality that hyperreduction will eventually be required. The approach defines a sequence of reduced optimization problems, where the ROM at a given point in the sequence is trained from snapshots of the HDM sampled at the solution of previous problems in the sequence. This is a common feature of aforementioned adaptive methods. Each reduced optimization problem is equipped with a constraint that sets an upper bound on the norm of the HDM residual evaluated at the solution of the ROM. The norm of the HDM residual is used as an error indicator for the ROM, as efficient and computable error bounds are not available for POD-based model order reduction in the general nonlinear parametric setting. The HDM residual norm and upper bound collectively define a nonlinear trust region. The upper bound on the nonlinear trust-region radius will be adaptively refined using an algorithm motivated from standard trust-region methods [5, 43]. Therefore, the proposed method incorporates generalizations of the error-awareness of [41] and the trust region ideology of [5]. Construction of a ROB and model order reduction framework such that the ROB accurately represents state vectors and sensitivities without degrading the approximation of either is crucial in the context of ROM-constrained optimization [5]. This issue is addressed in this work by collecting state and sensitivity snapshots separately, applying POD to each snapshot set, and concatenating the results to obtain the ROB [18]. Degradation of the ROM solution and their sensitivities are avoided by applying minimum-residual methods for solving for each. In particular, the Least-Squares Petrov-Galerkin projection from [20, 27, 22] will be used to define the reduced *state* equations, which has a minimum-residual property thereby ensuring that, under some assumptions, the ROM approximation to the PDE state variables can only improve as vectors are added to the ROB. In the present situation, this implies that the presence of sensitivity snapshots in the ROB cannot adversely affect the accuracy of the state approximation. Additionally, a novel minimum-error reduced sensitivity framework is introduced that ensures the reduced sensitivity approximation to the PDE sensitivities can only improve as vectors are added to the ROB, i.e. state snapshots in the ROB will not adversely affect the sensitivity approximation. The minimum-residual formulations for the reduced state and sensitivity equations, along with careful selection of the offset and basis vectors used to define the low-dimensional affine subspace of the ROM, will be shown to generate ROMs whose solution and sensitivities exactly match those of the HDM at training points, provided mild conditions are met. Finally, fast basis updating techniques [44] will be discussed for adding snapshot contributions to a ROB without requiring an SVD computation of the entire snapshot matrix appended with new data.

In this paper, *steady-state*, nonlinear PDE constraints are considered using the NAND framework in the discretize-then-differentiate setting for PDE-constrained optimization. A variant of POD,

presented in Section 3.3, is used to generate an optimization-oriented ROB from state and sensitivity snapshots collected from sampling the HDM at various parameter values. Although the proposed method for ROM-constrained optimization is derived in this framework, it can be generalized to other PDE-constrained optimization frameworks and other methods of ROB construction.

The remainder of the paper is organized as follows. Section 2 presents a brief overview of PDE-constrained optimization. Section 3 introduces ROMs and a technique for including sensitivity snapshots in the construction of the ROB that will have implications in the context of ROM-constrained optimization. Section 4 presents the proposed approach for solving a PDE-constrained optimization problem with a progressively-constructed ROM, including algorithmic details that guarantee desirable properties for each reduced optimization problem. Section 5 demonstrates the capabilities of the progressively-constructed ROM approach in solving relevant CFD problems. Section 6 offers conclusions and avenues for further research.

2. PDE-CONSTRAINED OPTIMIZATION

In this section, the framework chosen for PDE-constrained optimization is described. It is emphasized that these choices are made solely due to convenience of integration with currently available tools. The proposed approach to ROM-constrained optimization in Section 4 is general in that it is applicable to any PDE-constrained optimization framework.

Recall the focus of this work is *steady-state* partial differential equations. Define the PDE-constrained optimization problem as

$$
\begin{aligned}
\underset{\mathbf{w}\in\mathcal{W},\ \boldsymbol{\mu}\in\mathcal{D}}{\text{minimize}} \quad & \hat{\mathcal{J}}\left(\mathbf{w},\boldsymbol{\mu}\right) \\
\text{subject to} \quad & \hat{\mathbf{c}}(\mathbf{w},\boldsymbol{\mu}) \leq 0 \\
& \boldsymbol{\mathcal{L}}(\mathbf{w},\boldsymbol{\mu}) = 0,
\end{aligned}
\tag{1}
$$

where $\mathcal{W}$ is the state space of the corresponding PDE, $\mathcal{D} \subset \mathbb{R}^{n_p}$ is the parameter space, $\hat{\mathcal{J}} : \mathcal{W} \times \mathcal{D} \to \mathbb{R}$ is the objective function, $\hat{\mathbf{c}} : \mathcal{W} \times \mathcal{D} \to \mathbb{R}^{n_c}$ are n_c additional constraints on the system, and $\mathcal{L}$ is the nonlinear, vector-valued differential operator defining the PDE of interest. In the following, the discretization of (1) for integration with an optimization solver is discussed. The treatment of the PDE constraint and derivation of sensitivities are also discussed.

2.1. Discretize-then-differentiate vs. differentiate-then-discretize

The two approaches to deriving a discretized optimization problem of (1) are known as discretize-then-differentiate and differentiate-then-discretize [45, 46]. In the discretize-then-differentiate approach, the PDE is discretized prior to the derivation of the sensitivity equations, while the differentiate-then-discretize approach derives the sensitivity equations at the PDE-level and discretizes the result. The operations of discretization and differentiation do not commute and the distinction between these approaches is significant.

The discretize-then-differentiate approach is used throughout this work, however; formulation in the differentiate-then-discretize can be developed similarly. At this point, the PDE constraint in (1) is discretized, resulting in the following optimization problem

$$
\begin{aligned}
\underset{\mathbf{w}\in\mathbb{R}^{N},\ \boldsymbol{\mu}\in\mathcal{D}}{\text{minimize}} \quad & \mathcal{J}\left(\mathbf{w},\boldsymbol{\mu}\right) \\
\text{subject to} \quad & \mathbf{c}(\mathbf{w},\boldsymbol{\mu}) \leq 0 \\
& \mathbf{R}(\mathbf{w},\boldsymbol{\mu}) = 0,
\end{aligned}
\tag{2}
$$

where $\mathbf{w} \in \mathbb{R}^{N}$ is the discretized PDE state vector, $\mathcal{J} : \mathbb{R}^{N} \times \mathcal{D} \to \mathbb{R}$ is the discretized objective function, $\mathbf{c} : \mathbb{R}^{N} \times \mathcal{D} \to \mathbb{R}^{n_c}$ are the n_c additional constraints (discretized), and $\mathbf{R}(\mathbf{w},\boldsymbol{\mu}) = 0$ is the discretized PDE. In the next section, treatment of the discretized PDE constraint and sensitivity derivation is discussed.

2.2. Nested analysis and design (NAND)

The NAND [1] approach to PDE-constrained optimization uses the PDE to implicitly define the state vector as a function of the parameters. In this setting, the only optimization variables are the PDE parameters; however, any parameter value change requires solving the PDE to obtain the corresponding state. This can be viewed as enforcing satisfaction of the discrete PDE constraint at every iteration of the optimization, whereas most algorithms allow infeasibilities during intermediate iterations. In general, the NAND approach enables the use of a "black-box" PDE solver, although many PDE solutions may be required during the optimization process. Alternatives to the NAND approach include full-space and reduced-space Simultaneous Analysis and Design (SAND) [1, 47] methods. In this work, the proposed method is formulated in the NAND framework; however, it is equally applicable to SAND frameworks.

In the NAND approach, $\mathbf{w} = \mathbf{w}(\boldsymbol{\mu})$ implicitly through the discretized PDE, $\mathbf{R}(\mathbf{w}, \boldsymbol{\mu}) = 0$. Additionally, satisfaction of the PDE constraint for all $\boldsymbol{\mu} \in \mathcal{D}$ in the optimization procedure implies that the PDE states lie on the manifold defined by

$$\mathbf{R}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0. \tag{3}$$

On this manifold,

$$\frac{\mathrm{d}\mathbf{R}}{\mathrm{d}\boldsymbol{\mu}} = \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}} + \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}} = 0$$

holds and is used to compute the sensitivities of the state variables with respect to the PDE parameters

$$\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}} = - \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \right]^{-1} \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}, \tag{4}$$

which is subsequently used to compute the gradients of the objective function and constraints.

Consider some functional $\mathcal{F}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu})$, which may represent the objective function or a constraint. The gradient of $\mathcal{F}$ with respect to $\boldsymbol{\mu}$ takes the form

$$\frac{\mathrm{d}\mathcal{F}}{\mathrm{d}\boldsymbol{\mu}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} + \frac{\partial \mathcal{F}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}} = \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} - \frac{\partial \mathcal{F}}{\partial \mathbf{w}} \left(\left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \right]^{-1} \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}} \right) \tag{5}$$

$$= \frac{\partial \mathcal{F}}{\partial \boldsymbol{\mu}} - \left[\frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-T} \frac{\partial \mathcal{F}}{\partial \mathbf{w}}^{T} \right]^{T} \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}. \tag{6}$$

Equations (5) and (6) suggest two ways of computing the gradient of the functional, known as the direct method and adjoint method. In the direct approach in (5), the state sensitivity is computed directly, requiring the solution of n_p linear systems with the PDE Jacobian matrix. This may be very expensive, but is only required once, regardless of the number of functionals that need to be differentiated, allowing for an arbitrary number of objectives and constraints at the cost of n_p linear systems solves. The adjoint approach in (6) requires a single linear system solution with the transpose of the PDE Jacobian, regardless of the number of parameters involved. However, such a solve is required for each functional to be differentiated. Also notice that $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}$ is never directly computed in the adjoint method. Therefore, the direct method is preferred over the adjoint method when there are a large number of functionals (objective and constraint functions) compared to the number of parameters and vice versa. In this work, sensitivities of the state vector, $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}$, are to be used as snapshots in the training of a ROB, see Section 3.3, thereby motivating the use of the direct method.

3. MODEL REDUCTION

3.1. High-dimensional model

The discretization of the governing PDE will serve as a set of nonlinear equality constraints in the optimization problem. The discretized, steady, nonlinear PDE takes the form

$$\mathbf{R}(\mathbf{w}, \boldsymbol{\mu}) = 0 \tag{7}$$

where $\mathbf{w} \in \mathbb{R}^N$ is the discretized state vector, $\boldsymbol{\mu} \in \mathcal{D} \subset \mathbb{R}^{n_p}$ represents the parameters of the PDE, i.e. shape, control, material, and N assumed to be large. Depending on the application, it is common practice to use homotopy to solve the above nonlinear function as opposed to applying a nonlinear solver to (7) directly. Since a good initial guess is usually not known for (7), the goal of homotopy is to generate a sequence of "easier" subproblems, each with an obvious, high-quality initial guess. Common applications of homotopy are: structural mechanics where load stepping is used and CFD where pseudo-time-stepping is the standard [48, 49].

3.2. Reduced-order model

The goal of model order reduction is to reduce the HDM (7), in the sense of reducing the number of degrees of freedom and nonlinear equations, with the expectation of computational savings in terms of both storage and CPU time. The central assumption in MOR is the state vector lies (approximately) in a low-dimensional affine subspace

$$\mathbf{w} \approx \mathbf{w}_r = \bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \tag{8}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{N \times k_y}$ is the ROB of dimension $k_y \ll N$. The offset vector $\bar{\mathbf{w}}$ can be used to make the reduced approximation *exact* at a particular point in state space, namely $\bar{\mathbf{w}}$, usually the initial condition. This implies that the ROB can be constructed to represent *perturbations* about $\bar{\mathbf{w}}$, instead of the state vector itself [44]. The offset vector $\bar{\mathbf{w}}$ is assumed constant or piecewise constant with respect to $\boldsymbol{\mu}$ and is revisited in Sections 3.3 and 4. Implicit in assumption (8) is the sensitivities of the state vector can be represented in *the same* low-dimensional subspace, modulo the affine offset

$$\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}} \approx \frac{\partial \mathbf{w}_r}{\partial \boldsymbol{\mu}} = \boldsymbol{\Phi}\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}, \tag{9}$$

which is obtained by differentiating (8) with respect to the parameters, $\boldsymbol{\mu}$.

Substituting the assumption in (8) into the HDM in (7) and constraining the nonlinear equations to be orthogonal to the subspace spanned by a left subspace $\boldsymbol{\Psi} \in \mathbb{R}^{N \times k_y}$, the ROM is

$$\boldsymbol{\Psi}^T \mathbf{R}(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}; \boldsymbol{\mu}) = 0. \tag{10}$$

It was shown in [22] that the left basis can be constructed such that the nonlinear search directions of (10) minimize the distance to the Newton direction of the HDM, in some norm. For $\boldsymbol{\Psi} = \boldsymbol{\Phi}$, (10) corresponds to a Galerkin projection of the HDM (7) and leads to "optimal" search directions in the $\dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}$ norm for problems with Symmetric-Positive Definite (SPD) Jacobians. For $\boldsymbol{\Psi}(\mathbf{w}, \boldsymbol{\mu}) = \dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}(\mathbf{w}, \boldsymbol{\mu})\boldsymbol{\Phi}$, (10) is a Least-Squares Petrov-Galerkin (LSPG) [20, 27, 22] projection, which is "optimal" in the $\dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}^T \dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}$ norm for problems with Jacobians that are not necessarily SPD.

It can be shown that the LSPG form of (10) is precisely the first-order optimality condition of

$$\underset{\mathbf{y} \in \mathbb{R}^{k_y}}{\text{minimize}} \quad \frac{1}{2}||\mathbf{R}\left(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \boldsymbol{\mu}\right)||_2^2. \tag{11}$$

A ROM that is equivalent to a minimization problem of the form (11) will be said to possess the minimum-residual property. The minimum-residual property is desirable as it guarantees that

additional information in a ROB cannot degrade the solution approximation, neglecting issues related to convergence and assuming a monotonic relationship between residual norm and error. In this work, the general case of the LSPG ROM is used. The proposed method is equally applicable to Galerkin ROMs and the derivation of the reduced sensitivities simplifies as the left basis does not depend on the state vector, as will be shown in Section 4.1.

As previously mentioned, the projection-based ROM in (10) is not sufficient to generate non-trivial speedups due to the large-dimensional operations that must be done at every nonlinear iteration to evaluate and project the nonlinear terms. A slew of *hyperreduction* techniques have been developed, most of which aim to approximate the nonlinear terms in a low-dimensional subspace [21, 22]. An important component of these hyperreduction techniques is the concept of *masked* or gappy nonlinear terms, defined as a subset of the entries of a high-dimensional nonlinear quantity. The masked nonlinear terms are used to determine the remaining entries via interpolation or least-squares minimization. Additionally, the sample mesh is defined as the subset of the entire mesh required to compute the masked nonlinear terms. This approach has been shown to generate promising speedups as the nonlinear terms only need to be evaluated on a small subset of the original mesh and the large-dimensional projections can be precomputed.

In this work, the proposed method is only derived for the projection-based ROM (10); however, it is equally applicable in the context of hyperreduction with several complications that arise when only masked nonlinear terms are available. In the remainder of this paper, extensions to hyperreduction are mentioned, although a full exposition using hyperreduced models as a surrogate for PDE-constrained optimization is postponed to future work.

3.3. Offline basis construction: proper orthogonal decomposition and the method of snapshots

The method of snapshots in conjunction with Proper Orthogonal Decomposition (POD) [17] is among the most popular and successful techniques for collecting and compressing information to yield a ROB in the context of model order reduction. The method of snapshots consists of sampling the HDM at various parameter configurations and collecting snapshots, usually state vectors, from the simulations. POD, also known as Principal Component Analysis (PCA) or the Karhunen-Loeve procedure is used to compress the snapshots, exploiting the optimal approximation properties of the Singular Value Decomposition (SVD), to yield a ROB. It is worth mentioning that a complete singular value decomposition is among the most expensive matrix factorizations implying that basis construction via POD can constitute a significant portion of the traditional offline phase, particularly for a large dimensional state space and many snapshots.

Algorithm 1 Proper Orthogonal Decomposition

Input: Snapshot matrix, $\mathbf{X} \in \mathbb{R}^{N \times k_s}$ and ROB size, k_y
Output: ROB, $\mathbf{\Phi}$
  1: Compute the thin SVD of $\mathbf{X}$: $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{k_s} \end{bmatrix}$
  2: $\mathbf{\Phi} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{k_y} \end{bmatrix}$

From the low-dimensionality assumptions in (8) and (9), it is clear that the subspace spanned by the ROB needs to (approximately) contain $\mathbf{w}(\boldsymbol{\mu}) - \bar{\mathbf{w}}$ and $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu})$ if the ROM is to accurately represent the HDM solution and sensitivity at $\boldsymbol{\mu}$. This suggests using a snapshot matrix of the form

$$\mathbf{X} = \left[\mathbf{w}(\boldsymbol{\mu}) - \bar{\mathbf{w}}, \frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu}) \right] \tag{12}$$

is desirable in the context of optimization, as recognized in [50, 31, 33]. As POD is sensitive to the relative scale of the columns of the data matrix, it cannot be directly applied to a snapshot matrix containing states and sensitivities due to the inherent scale difference. A heuristic weighting procedure was developed in [31, 33] to address the scale differences between the different snapshot types.

An alternate approach [18] applies POD to the state and sensitivity snapshots individually and concatenates the result to form the ROB. If an orthogonal ROB is desired, a Gram-Schmidt-like procedure can be applied. This approach has the advantages of: (1) avoiding heuristic snapshot weighting, (2) individually controlling the contribution of each quantity to the ROB, and (3) exploiting the optimality properties of POD for both the states and the sensitivities as both are required in the optimization context . Then, if k_y^{state} is the dimension of the subspace spanned by the state vector basis and k_y^{sens} is the dimension of the sensitivity subspace, the dimension of the ROB is $k_y = k_y^{\text{state}} + k_y^{\text{sens}}$.

Algorithm 2 State-Sensitivity Proper Orthogonal Decomposition

---

Input: State snapshot matrix, $\mathbf{X}_{\text{state}} \in \mathbb{R}^{N \times k_s^{\text{state}}}$, sensitivity snapshot matrix, $\mathbf{X}_{\text{sens}} \in \mathbb{R}^{N \times k_s^{\text{sens}}}$, reduced state size, k_y^{state} and reduced sensitivity size, k_y^{sens}
Output: ROB, $\mathbf{\Phi}$
  1: Apply POD (Algorithm 1) to $\mathbf{X}_{\text{state}}$: $\text{POD}(\mathbf{X}_{\text{state}}, k_y^{\text{state}}) \rightarrow \mathbf{\Phi}_{\text{state}}$
  2: Apply POD (Algorithm 1) to $\mathbf{X}_{\text{sens}}$: $\text{POD}(\mathbf{X}_{\text{sens}}, k_y^{\text{sens}}) \rightarrow \mathbf{\Phi}_{\text{sens}}$
  3: $\mathbf{\Phi} = \begin{bmatrix} \mathbf{\Phi}_{\text{state}} & \mathbf{\Phi}_{\text{sens}} \end{bmatrix}$
  4: (optional) Orthogonalize $\mathbf{\Phi}$ via QR (modified Gram-Schmidt)

In [50], the authors recognized the importance for optimization of having the ROM objective gradient approximate well the HDM objective gradient, particularly when aiming for convergence to the optimal solution. It was further stated that the incorporation of sensitivities in the POD basis without compromising the state approximation quality is an open question. An equally important issue is that of how state information in the ROB affects the approximation of a sensitivity.

More recently, [37] introduced two approaches for incorporating sensitivities into a ROB, one of which successfully improved the state approximation at the cost of increasing the basis size. Additionally, a convergence theory for approximating sensitivities was provided in [51] in the context of the Empirical Interpolation Method (EIM). This theory implies that sensitivity snapshots are unnecessary as the sensitivity approximation error will be driven to zero as the state approximation goes to zero.

In this work, the questions regarding the degradation of the state or sensitivity approximation by the incorporation of extraneous snapshots are addressed by using *minimum-residual* formulations for both the state and sensitivity equations. In particular, the minimum-residual LSPG framework presented in Section 3.2 ensures that appending sensitivity information to the ROB cannot degrade the state approximation, assuming a monotonic relationship between residual norm and error. Additionally, a novel minimum-error framework for computing reduced sensitivities which ensures that appending state information to a ROB does not adversely affect the approximation of the sensitivities is presented in Section 4.1.

4. OPTIMIZATION USING A PROGRESSIVE ROM

In this section, a framework for approximating the solution of the PDE-constrained optimization (2) using progressively constructed ROMs in a nonlinear trust region framework is presented. The PDE-constrained optimization (2) is reduced using the ROM (10) as a surrogate for the high-dimensional PDE to produce the reduced optimization problem, which will be used interchangeably with the term *ROM-constrained optimization*,

$$
\begin{aligned}
\underset{\mathbf{y} \in \mathbb{R}^{k_y}, \, \boldsymbol{\mu} \in \mathcal{D}}{\text{minimize}} \quad & \mathcal{J}(\bar{\mathbf{w}} + \mathbf{\Phi}\mathbf{y}, \boldsymbol{\mu}) \\
\text{subject to} \quad & \mathbf{c}(\bar{\mathbf{w}} + \mathbf{\Phi}\mathbf{y}, \boldsymbol{\mu}) \leq 0 \\
& \mathbf{\Psi}^T \mathbf{R}(\bar{\mathbf{w}} + \mathbf{\Phi}\mathbf{y}, \boldsymbol{\mu}) = 0.
\end{aligned}
\tag{13}
$$

In the remainder of this section, the gradients of the ROM-constrained optimization problem are derived for Galerkin and LSPG ROMs, including a minimum-error formulation for the sensitivity

computation. Subsequently, the proposed progressive framework to approximate the solution of (2) is presented with all relevant algorithmic details made explicit.

At this point, the reader is reminded that the main objective of this paper is to demonstrate the accuracy of the proposed ROB-based computational technology. Its potential for delivering the desired speedup will be demonstrated in future work, after it is equipped with hyperreduction. Therefore, in the remainder, large dimensional quantities will appear in the computation of reduced quantities.

4.1. Sensitivities

In the NAND framework, the reduced sensitivities $\dfrac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}$ can be derived by leveraging the fact that $\boldsymbol{\Psi}^T \mathbf{R}(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\mu}), \boldsymbol{\mu}) = 0$ is maintained throughout the solution process. Employing a procedure identical to that used in Section 2.2, in the general case where $\boldsymbol{\Psi} = \boldsymbol{\Psi}(\mathbf{w}_r(\mathbf{y}), \boldsymbol{\mu})$, the reduced sensitivities are

$$\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}} = - \left[\sum_{j=1}^{N} \mathbf{R}_j \frac{\partial \left(\boldsymbol{\Psi}^T \mathbf{e}_j \right)}{\partial \mathbf{w}} \boldsymbol{\Phi} + \boldsymbol{\Psi}^T \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \boldsymbol{\Phi} \right]^{-1} \left[\sum_{j=1}^{N} \mathbf{R}_j \frac{\partial \left(\boldsymbol{\Psi}^T \mathbf{e}_j \right)}{\partial \boldsymbol{\mu}} + \boldsymbol{\Psi}^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}} \right] \qquad (14)$$

where $\mathbf{e}_j$ is the jth canonical unit vector and the relation $\dfrac{\partial \mathbf{w}_r}{\partial \mathbf{y}} = \boldsymbol{\Phi}$ is used from (8). In the case of a Galerkin ROM, the left basis is state- and parameter-independent and the reduced sensitivities become

$$\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}} = - \left[\boldsymbol{\Phi}^T \frac{\partial \mathbf{R}}{\partial \mathbf{w}} \boldsymbol{\Phi} \right]^{-1} \left[\boldsymbol{\Phi}^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}} \right]. \qquad (15)$$

The reduced sensitivities in the LSPG case are

$$\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}} = - \left[\sum_{j=1}^{N} \mathbf{R}_j \boldsymbol{\Phi}^T \frac{\partial^2 \mathbf{R}_j}{\partial \mathbf{w} \partial \mathbf{w}} \boldsymbol{\Phi} + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \boldsymbol{\Phi} \right)^T \left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \boldsymbol{\Phi} \right) \right]^{-1} \left[\sum_{j=1}^{N} \mathbf{R}_j \boldsymbol{\Phi}^T \frac{\partial^2 \mathbf{R}_j}{\partial \mathbf{w} \partial \boldsymbol{\mu}} + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}} \boldsymbol{\Phi} \right)^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}} \right] \qquad (16)$$

and require the (reduced) second-order sensitivities $\boldsymbol{\Phi}^T \dfrac{\partial^2 \mathbf{R}_j}{\partial \mathbf{w} \partial \mathbf{w}} \boldsymbol{\Phi}$ and $\boldsymbol{\Phi}^T \dfrac{\partial^2 \mathbf{R}_j}{\partial \mathbf{w} \partial \boldsymbol{\mu}}$, which may not be available in black-box PDE solvers and expensive to compute with finite differences. The higher-order terms may be neglected with minor loss in accuracy provided $\|\mathbf{R}\|$ is small, which will not, in general, be true in the context of parametric model order reduction.

In the case of hyperreduction, equations (14) - (16) hold with the nonlinear terms replaced with the *masked* nonlinear terms and the state vector $\mathbf{w}$ replaced with the restriction of the state vector to the sample mesh. This ensures that all operations involved in the sensitivity computations are independent of the large dimension N.

Instead of computing the reduced sensitivities directly using one of (14) - (16), approximate sensitivities can be constructed that minimize the distance to the corresponding HDM sensitivity in some norm $\boldsymbol{\Theta} \succ 0$. In this spirit, for each parameter, the desired sensitivity approximation is given by

$$\widehat{\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}_j}} = \arg\min_{\mathbf{a} \in \mathbb{R}^{k_y}} \frac{1}{2} \|\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}_j} - \boldsymbol{\Phi}\mathbf{a}\|_{\boldsymbol{\Theta}}^2, \qquad (17)$$

where $\widehat{\dfrac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}}$ is to be used as a surrogate for $\dfrac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}$. Using the expression for the HDM sensitivity (4) in (17), the solution of the linear least-squares problems result in

$$\widehat{\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}} = - \left(\boldsymbol{\Theta}^{1/2} \boldsymbol{\Phi} \right)^{\dagger} \boldsymbol{\Theta}^{1/2} \frac{\partial \mathbf{R}}{\partial \mathbf{w}}^{-1} \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}, \qquad (18)$$

where † denotes the Moore-Penrose pseudoinverse. Selecting $\boldsymbol{\Theta}^{1/2} = \dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}$, reduces (18) to

$$\widehat{\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}} = -\left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}\right)^{\dagger} \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}, \tag{19}$$

which is equivalent to the optimality condition of

$$\underset{\mathbf{a} \in \mathbb{R}^{k_y}}{\text{minimize}} \quad \frac{1}{2}||\frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}_j} + \frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}\mathbf{a}||_2^2. \tag{20}$$

This shows that choosing $\boldsymbol{\Theta} = \dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}^T \dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}$ has the advantage of minimizing the state sensitivity error in the reduced subspace with respect to the $\boldsymbol{\Theta}$-norm *and* minimizing $||\dfrac{d\mathbf{R}}{d\boldsymbol{\mu}}||$ (used in derivation of HDM state sensitivity) in the span of $\dfrac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}$ with respect to the $2-$norm. The corresponding normal equations are

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}\right)^T \left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}\right) \widehat{\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}} = -\left(\frac{\partial \mathbf{R}}{\partial \mathbf{w}}\boldsymbol{\Phi}\right)^T \frac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}. \tag{21}$$

These are equivalent to the LSPG sensitivity equations in (16) with the second-order terms dropped. In the context of optimization, the danger in using the minimum-error sensitivity computation is that the computed gradients will not be consistent with the functionals to which they correspond, which may result in convergence issues for the reduced optimization problem (13). The LSPG approach possesses the minimum-error sensitivity property (by dropping second-order terms), which approaches the true reduced sensitivity in regions of the parameter space where the ROM has high accuracy ($||\mathbf{R}||$ small), i.e. near HDM samples.

The minimum-error approach to computing reduced sensitivities ensures additional, unnecessary information in the ROB cannot degrade the sensitivity approximation, closing the loop in the discussion from Section 3.3. Additionally, using the state-sensitivity variant of POD in Algorithm 2 and skipping the truncation step of the *sensitivity* basis, we have $\boldsymbol{\Phi}\widehat{\dfrac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}}}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu}) = \dfrac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu})$ for the values of the parameters $\boldsymbol{\mu}$ at which the HDM was sampled, i.e. the reconstructed reduced senitivities exactly match the HDM sensitivities at training points. This property follows directly from the minimum-error sensitivity formulation of the reduced sensitivities and the fact that the desired sensitivity is contained in the span of the ROB since the truncation step is skipped.

4.2. ROM-constrained optimization with progressive reduced-basis construction

The difficulty of training a static, global ROB which is expected to perform well throughout the parameter space, even for a small number of parameters, suggests the need for an adaptive procedure that merges the offline and online phases. In the case of optimization, adaptive basis construction results in a ROM specialized for a region of the parameter space about the optimization trajectory. In this work, the adaptive optimization procedure is driven by (13) equipped with a residual-based nonlinear trust region

$$\begin{aligned}
\underset{\mathbf{y} \in \mathbb{R}^{k_y},\ \boldsymbol{\mu} \in \mathcal{D}}{\text{minimize}} \quad & \mathcal{J}\left(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \boldsymbol{\mu}\right) \\
\text{subject to} \quad & \mathbf{c}(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \boldsymbol{\mu}) \leq 0 \\
& \boldsymbol{\Psi}^T \mathbf{R}(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \boldsymbol{\mu}) = 0 \\
& \frac{1}{2}||\mathbf{R}(\bar{\mathbf{w}} + \boldsymbol{\Phi}\mathbf{y}, \boldsymbol{\mu})||_2^2 \leq \epsilon.
\end{aligned} \tag{22}$$

The proposed approach to ROM-constrained optimization is defined as a sequence of reduced optimization problems of the form (22). The ROB, $\boldsymbol{\Phi}$, used in the jth reduced optimization problem is constructed using HDM samples at the solution of a subset of all previous reduced optimization

problems. Assuming a monotonic relationship between residual norm and error, proper selection of ϵ implies the trajectory of the jth reduced optimization problem will remain in a region of the parameter space where the corresponding ROM is accurate. If the solution of the jth instance of (22), denoted $\boldsymbol{\mu}_j^*$, does not satisfy the optimality conditions of the PDE-constrained optimization problem (2), the ROB can be improved by sampling the HDM at $\boldsymbol{\mu}_j^*$. As there is no formal convergence proof to guarantee that $\boldsymbol{\mu}_j^*$ will converge to a point satisfying the optimality conditions of (2), a heuristic convergence criteria

$$||\boldsymbol{\mu}_{j-1} - \boldsymbol{\mu}_j|| \leq \delta ||\boldsymbol{\mu}_j|| \tag{23}$$

is used, similar to [4].

With the fundamental approach introduced, subsequent sections will discuss crucial details that endow the proposed method with desirable properties, making the method viable for practical applications. Among these details are: initial parameter value used for the jth optimization subproblem; the reference vector, $\bar{\mathbf{w}}$, and initial state, $\mathbf{w}^{(0)}$, used in the nonlinear iteration to solve the ROM (10) at each parameter configuration; selection of the residual upper bound, ϵ, for the jth reduced optimization problem; and an algorithm for efficiently updating the ROB with new snapshots and reference vector without re-computing the SVD of the entire snapshot matrix, which grows as samples of the HDM are added. For future reference, define the kth iteration of the jth reduced optimization problem as $\boldsymbol{\mu}_j^{(k)}$ and the solution of the jth reduced optimization problem as $\boldsymbol{\mu}_j^*$. For notational convenience, define $\boldsymbol{\mu}_{-1}^* = \boldsymbol{\mu}_0^{(0)}$.

This section concludes with three definitions that are used in the remainder of this paper. First, define $\mathcal{S}_j^{\boldsymbol{\mu}}$ as the set of parameter values at which the HDM has been sampled after the first $j+1$ instances of (22)

$$\mathcal{S}_j^{\boldsymbol{\mu}} = \{\boldsymbol{\mu}_{-1}^*, \boldsymbol{\mu}_0^*, \boldsymbol{\mu}_1^*, \ldots, \boldsymbol{\mu}_j^*\}. \tag{24}$$

Secondly, define the set of HDM solutions for each parameter in $\mathcal{S}_j^{\boldsymbol{\mu}}$

$$\mathcal{S}_j^{\mathbf{w}} = \{\mathbf{w}(\boldsymbol{\mu}_{-1}^*), \mathbf{w}(\boldsymbol{\mu}_0^*), \mathbf{w}(\boldsymbol{\mu}_1^*), \ldots, \mathbf{w}(\boldsymbol{\mu}_j^*)\}. \tag{25}$$

It may be advantageous to add several HDM samples to the training set before the optimization process is started in order to improve the quality of the initial ROB. This can be accomplished by extending the definitions of $\mathcal{S}_j^{\boldsymbol{\mu}}$ and $\mathcal{S}_j^{\mathbf{w}}$ to include such samples. Further, define the ratio of actual objective reduction to predicted reduction, mimicking the corresponding quantity from traditional trust-region theory

$$\rho(\boldsymbol{\mu}, \boldsymbol{\nu}) = \frac{\mathcal{J}(\mathbf{w}(\boldsymbol{\nu}), \boldsymbol{\nu}) - \mathcal{J}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu})}{\mathcal{J}(\bar{\mathbf{w}}(\boldsymbol{\nu}) + \boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\nu}), \boldsymbol{\nu}) - \mathcal{J}(\bar{\mathbf{w}}(\boldsymbol{\mu}) + \boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\mu}), \boldsymbol{\mu})}. \tag{26}$$

Then, define $\rho_j = \rho(\boldsymbol{\mu}_{j-1}^*, \boldsymbol{\mu}_j^*)$. The dependence of the reference vector, $\bar{\mathbf{w}}$, on the parameter vector, $\boldsymbol{\mu}$, is piecewise constant, as discussed in Section 4.4, validating the implicit assumption $\frac{\partial \bar{\mathbf{w}}}{\partial \boldsymbol{\mu}} = 0$ in Sections 3.2 and 4.1.

4.3. Initial guess for optimization subproblem: parameter space

In the proposed progressive framework, a natural selection of the initial guess for the jth reduced optimization problem, $\boldsymbol{\mu}_j^{(0)}$, is the solution of the $j-1$ problem, $\boldsymbol{\mu}_{j-1}^*$. However, if ϵ is too large in (22), the optimization trajectory may proceed into regions of the parameter space where it lacks accuracy. In this case, the reduced optimization problem can converge to a parameter that *increases* the true objective function compared to the initial guess, i.e. a "worse" point in terms of objective function minimization.

In anticipation of the case where ϵ is set too large, the initial guess for the reduced optimization problems is the parameter at which the HDM was sampled that achieves the lowest value of the objective function, namely,

$$\boldsymbol{\mu}_{j+1}^{(0)} = \arg \min_{\boldsymbol{\mu} \in \mathcal{S}_j^{\boldsymbol{\mu}}} \mathcal{J}(\mathbf{w}(\boldsymbol{\mu}), \boldsymbol{\mu}). \tag{27}$$

If the jth reduced optimization problem reduces the true objective function, the starting point for the subsequent problem is $\boldsymbol{\mu}_j^*$, as expected. Notice that this choice of $\boldsymbol{\mu}_{j+1}^{(0)}$ does not require additional PDE solves as $\mathbf{w}(\boldsymbol{\mu})$ is computed at every HDM sample.

In the case where ϵ is set too large, the HDM is still sampled at the solution, $\boldsymbol{\mu}_j^*$, and the ROB updated. Therefore, the reduced optimization problem is not wasted because it adds information to the basis regarding areas of the parameter space to avoid, ensuring subsequent reduced optimization problems will not return to this point. The heuristic regarding sampling the "worse" point, while starting subsequent problems from the best point is an attempt to reduce the likelihood that an aggressive choice of ϵ will adversely affect convergence.

4.4. Initial guess for ROM solve and reference vector selection: state space

The solution of each reduced optimization problem (22) requires the solution of the nonlinear ROM equation (10) at each parameter value visited in the optimization procedure. Regardless of the method used to solve the system of nonlinear equations, a starting point that is close to the true solution will significantly improve convergence to the solution. Therefore, in the jth reduced optimization problem, given a parameter configuration $\boldsymbol{\mu}_j^{(k)}$ at which the ROM is to be solved, define the initial guess as

$$\mathbf{w}^{(0)} = \arg \min_{\mathbf{w} \in \mathcal{S}_j^{\mathbf{w}}} ||\mathbf{R}(\mathbf{w}, \boldsymbol{\mu}_j^{(k)})||. \tag{28}$$

This choice of the initial guess leverages the HDM samples taken thus far to use the best *available* starting point, in the sense of the residual norm, which is usually used to assess convergence in nonlinear solvers. From the previous section, the starting *parameter configuration* for the jth optimization subproblem is also chosen from the set of HDM samples. Therefore, if the optimization trajectory does not deviate too far from the starting point (27), the initial state (28) is expected to be a good starting point for the nonlinear solve.

The reference vector is set to the initial condition, $\bar{\mathbf{w}} = \mathbf{w}^{(0)}$. Selecting the reference vector in this way is a sufficient condition for consistency of a LSPG ROM [22, 23], in the sense that the exact solution is recovered at sampled parameter values when POD is applied *without* basis truncation. Furthermore, it was observed in [44] that using the initial condition as the reference vector outperforms other possible choices for small reduced-order bases. Perhaps the most significant implication of this choice of reference vector is that the ROM can exactly represent any state in the training set, regardless of truncation. To elaborate on this point, consider $\boldsymbol{\mu}_{j+1}^{(0)} = \boldsymbol{\mu}_k^*$ with $k \leq j$. From (28), $\bar{\mathbf{w}} = \mathbf{w}^{(0)} = \mathbf{w}(\boldsymbol{\mu}_k^*)$ the MOR assumption in (8) becomes $\mathbf{w}_r = \mathbf{w}(\boldsymbol{\mu}_k^*) + \boldsymbol{\Phi}\mathbf{y}$. Taking $\mathbf{y} = 0$, the solution of $\mathbf{R}(\mathbf{w}, \boldsymbol{\mu}_k^*) = 0$ is obtained, implying that the solution $\mathbf{w}(\boldsymbol{\mu}_k^*)$ is *reachable* by the ROM independently of $\boldsymbol{\Phi}$. The minimum-residual ROM formulation guarantees that the solution will be found, ignoring issues related to convergence. The implication of this discussion is that the choices in (27) - (28) along with the affine representation in (8) guarantee that the ROM is exact at the initial guess of each reduced optimization problem. Therefore, the reduced optimization functionals precisely match those of the HDM, and the minimum-error reduced sensitivity framework guarantees that gradients will also match, provided that the sensitivity basis is not truncated in Algorithm 2.

Notice that modifying the reference vector changes the ROB due to the definition of state snapshots in Section 3.3. Since the reference vector can potentially change at every iteration of each reduced optimization problem, it is paramount to avoid recomputing the ROB via POD with each new reference state. Therefore, efficient, rank-1 basis updates are used following the work in [44] to obtain a ROB corresponding to the *translated* snapshots (see Algorithm 6 and Section 4.6 for additional details).

Using basis updates to ensure that the ROB corresponds (approximately) to the POD of the snapshots referenced by the appropriate initial condition has been shown to increase accuracy of a given ROM [44]. However, in the context of optimization, the modification to the ROB corresponds to changing the definition of the objective function and constraints *within* the iterations of an optimization solve (22). Namely, the definition of the optimization functions are changing

throughout the optimization process, which will likely have adverse effects on convergence. In practice, this does not show up as an issue because of the following observation. At the jth reduced optimization problem, if $\boldsymbol{\mu}_j^{(0)}$ is close to $\boldsymbol{\mu}^*$, the solution of (2), it is unlikely that $\bar{\mathbf{w}}$ will change during the optimization procedure as $\boldsymbol{\mu}$ will not move far from $\boldsymbol{\mu}_j^{(0)}$: therefore, the ROM initial condition and reference vector will not move from $\mathbf{w}(\boldsymbol{\mu}_j^{(0)})$. If $\boldsymbol{\mu}_j^{(0)}$ is far from $\boldsymbol{\mu}^*$, converging the optimization subproblem is not important provided progress is made toward $\boldsymbol{\mu}^*$; usual practice is to set an upper bound on the number of iterations allowed during the reduced optimization problems. Therefore, the potential issue with convergence of the reduced optimization problem is benign.

4.5. Adaptive selection of residual upper bound

The selection of the residual upper bound, ϵ, for a given reduced optimization problem, is discussed in this section. As it is difficult to find a single value of ϵ a-priori that can be used for all reduced optimization problems, an adaptive approach is considered that modifies ϵ between successive reduced optimization solves. Consider a value of the residual upper bound used in the jth reduced optimization problem, ϵ_j; the value of ϵ_{j+1} is determined based on the value of ρ_j (26), the ratio of actual reduction of the objective function to predicted reduction. It is clear that $\rho_j \approx 1$ indicates the ROM was a good model for the HDM in the prescribed trust region, while ρ_j far from unity implies a poor model. Therefore, the following update strategy from standard trust-region theory [43] is proposed

$$\epsilon' = \begin{cases} \dfrac{1}{\tau}\epsilon & \rho_k \in \left[\dfrac{1}{2}, 2\right] \\ \epsilon & \rho_k \in \left[\dfrac{1}{4}, \dfrac{1}{2}\right) \cup (2, 4] \\ \tau\epsilon & \text{otherwise,} \end{cases} \tag{29}$$

where $0 < \tau < 1$. Solutions of the reduced optimization problem (22) that *increase* the true objective function ($\rho_k < 0$) result in a reduction of the residual upper bound, ϵ, by a factor of τ, add an HDM sample to the training set, and "reject" the step in the sense that subsequent reduced optimization problems will be started from samples with lower values of the objective function as per (27). This is one among many possible adaptation strategies. It is important to note that the adaptive ϵ only changes *between* reduced optimization problems, and not within iterations of a given reduced problem. This ensures that the constraint definitions will not change between iterations in an optimization solve, unless $\bar{\mathbf{w}}$ changes as discussed in Section 4.4.

The adaptive approach requires an initial value, ϵ_0 to be manually defined, which can be done using problem-specific information, if available. The goal of the adaptive ϵ procedure and the safe guards in Sections 4.3-4.4, is to make the progressive optimization process robust with respect to the initial value ϵ_0. If ϵ_0 is smaller than necessary, the initial ROM will be highly accurate within the nonlinear trust region and result in a value of ρ_1 close to unity, which will cause the adaptive algorithm to increase the value of ϵ for subsequent reduced optimization problems. If ϵ_0 is too large, the optimization trajectory may venture into regions of the parameter space that cannot be captured by the ROM, resulting in a value of ρ_1 far from unity, causing the value of ϵ to be decreased for the next reduced optimization problem. Notice that either under- or over-shooting the initial value of epsilon should only result in a few more HDM samples than would otherwise be necessary.

4.6. Online algorithm for updating reduced-basis with new snapshots and reference vectors

With each new HDM sample, a new ROB needs to be computed from the concatenation of the previous snapshot matrix with the new snapshots. As more HDM samples are taken, the number of columns in the snapshot matrix grows by the number of snapshots collected per HDM sample. For large problems, this quickly becomes prohibitively expensive as the cost of computing the SVD of a matrix scales quadratically with the number of columns [52]. Additionally, new reference vectors computed in each reduced optimization problem from (28) require computation of a new ROB with the state snapshots translated by the new $\bar{\mathbf{w}}$. This will also introduce a significant bottleneck as (28)

allows for the possibility of changing $\bar{\mathbf{w}}$ at each iteration of each reduced optimization problem, although in practice it is not this often. In this section, an extension of the work described in [44] that applies low-rank SVD updates [53] is presented to economically update a ROB with new snapshots and a new reference vector by leveraging the SVD of previous snapshot matrices.

The low-rank SVD update algorithm in [53] is used to cheaply compute the thin SVD of $\mathbf{X} + \mathbf{A}\mathbf{B}^T$ from the SVD of $\mathbf{X}$. With proper selection of $\mathbf{X}$, $\mathbf{A}$, and $\mathbf{B}$, this algorithm can be tailored to the cases of appending vectors to the data matrix and translating the columns [53], see Algorithms 5 and 6, respectively, in the Appendix.

In the context of the state-sensitivity POD (Algorithm 2), a new reference vector and additional state and sensitivity snapshots imply that the state basis must incorporate the additional state snapshots and reference vector and the sensitivity basis must incorporate the additional sensitivity snapshots. If $\mathbf{X}_{\text{state}}$ is the original state snapshot matrix and $\bar{\mathbf{w}}$ is the original reference vector, then $\boldsymbol{\Phi}_{\text{state}} = \text{POD}(\mathbf{X}_{\text{state}} - \bar{\mathbf{w}}\mathbf{1}^T)$, where $\mathbf{1}$ is the vector of ones of appropriate size. Similarly, the sensitivity basis is $\boldsymbol{\Phi}_{\text{sens}} = \text{POD}(\mathbf{X}_{\text{sens}})$, where $\mathbf{X}_{\text{sens}}$ is the sensitivity snapshot matrix. With a new reference vector, $\widehat{\mathbf{w}}$, and snapshot matrices, $\mathbf{Y}_{\text{state}}$, and $\mathbf{Y}_{\text{sens}}$, bases of the form $\widehat{\boldsymbol{\Phi}}_{\text{state}} = \text{POD}\left(\begin{bmatrix}\mathbf{X}_{\text{state}} & \mathbf{Y}_{\text{state}}\end{bmatrix} - \widehat{\mathbf{w}}\mathbf{1}^T\right)$ and $\widehat{\boldsymbol{\Phi}}_{\text{sens}} = \text{POD}\left(\begin{bmatrix}\mathbf{X}_{\text{sens}} & \mathbf{Y}_{\text{sens}}\end{bmatrix}\right)$ are desired. This can be achieved without directly recomputing the POD of the modified snapshot matrices by applying low-rank basis updates. The desired bases can be written as

$$\widehat{\boldsymbol{\Phi}}_{\text{state}} = \text{POD}\left(\begin{bmatrix}\mathbf{X}_{\text{state}} - \bar{\mathbf{w}}\mathbf{1}^T + (\bar{\mathbf{w}} - \widehat{\mathbf{w}})\mathbf{1}^T & \mathbf{Y}_{\text{state}} - \widehat{\mathbf{w}}\mathbf{1}^T\end{bmatrix}\right)$$
$$\widehat{\boldsymbol{\Phi}}_{\text{sens}} = \text{POD}\left(\begin{bmatrix}\mathbf{X}_{\text{sens}} & \mathbf{Y}_{\text{sens}}\end{bmatrix}\right).$$

This form makes it clear that a *translation* update with the vector $\bar{\mathbf{w}} - \widehat{\mathbf{w}}$ can be applied to $\boldsymbol{\Phi}_{\text{state}}$ and an *append* update with the vectors $\mathbf{Y}_{\text{state}} - \widehat{\mathbf{w}}\mathbf{1}^T$ to yield the desired $\widehat{\boldsymbol{\Phi}}_{\text{state}}$. Similarly, $\widehat{\boldsymbol{\Phi}}_{\text{sens}}$ is obtained by applying an *append* update with the matrix $\mathbf{Y}_{\text{sens}}$. This procedure is summarized in Algorithm 3. To obtain the updated ROB with the new reference vector $\widehat{\mathbf{w}}$ without additional snapshots, as required in Section 4.4, Algorithm 3 can be used with $\mathbf{Y}_{\text{state}} = \mathbf{Y}_{\text{sens}} = \emptyset$.

Algorithm 3 Updating ROB from state-sensitivity POD: appending vectors and changing reference vector

Input: State and sensitivity bases: $\boldsymbol{\Phi}_{\text{state}}$, $\boldsymbol{\Phi}_{\text{sens}}$ as well as corresponding singular values ($\boldsymbol{\Sigma}_{\text{state}}$ and $\boldsymbol{\Sigma}_{\text{sens}}$) and right singular vectors ($\mathbf{V}_{\text{state}}$ and $\mathbf{V}_{\text{sens}}$); original reference vector, $\bar{\mathbf{w}}$; new reference vector, $\widehat{\mathbf{w}}$; new state snapshots, $\mathbf{Y}_{\text{state}}$; new sensitivity snapshots, $\mathbf{Y}_{\text{sens}}$

Output: Updated ROB, $\widehat{\boldsymbol{\Phi}}$

1: Update $\boldsymbol{\Phi}_{\text{sens}}$ with additional snapshots, $\mathbf{Y}_{\text{sens}}$ via Algorithm 5 $\rightarrow \widehat{\boldsymbol{\Phi}}_{\text{sens}}$
2: Update $\boldsymbol{\Phi}_{\text{state}}$ with new reference vector via Algorithm 6 applied to $\bar{\mathbf{w}} - \widehat{\mathbf{w}} \rightarrow \bar{\boldsymbol{\Phi}}_{\text{state}}$
3: Update $\bar{\boldsymbol{\Phi}}_{\text{state}}$ with additional snapshots, $\mathbf{Y}_{\text{state}} - \widehat{\mathbf{w}}\mathbf{1}^T \rightarrow \widehat{\boldsymbol{\Phi}}_{\text{state}}$
4: Define $\widehat{\boldsymbol{\Phi}} = \begin{bmatrix}\widehat{\boldsymbol{\Phi}}_{\text{state}} & \widehat{\boldsymbol{\Phi}}_{\text{sens}}\end{bmatrix}$
5: (optional) Orthogonalize $\widehat{\boldsymbol{\Phi}}$ via QR (modified Gram-Schmidt)

It is important to mention that the low-rank SVD updates in [53] assume the entire economy SVD is available to perform the update. In our application, only the truncated SVD may be available, resulting in an error in the update. It can be shown that, in the case of column translation, the error is proportional to the largest neglected singular value (usually small in MOR applications). Additionally, it can be shown that the error in the update does not noticeably affect the quality of the basis. Finally, it can also be shown that the thin SVD update of Brand can be applied in the context of hyperreduction, where only *masked* singular factors are available.

4.7. Summary: proposed ROM-constrained optimization algorithm

Section 4 concludes with a summary of the proposed approach to ROM-constrained optimization using progressively-constructed ROMs, including all relevant implementation details.

Algorithm 4 PDE-constrained optimization using progressively-constructed ROMs

---

Input: Initial guess for optimization, $\boldsymbol{\mu}_0^{(0)}$, initial residual upper bound, ϵ_0, convergence tolerance, δ and maximum number of reduced optimization solves, $N_{\max}$

Output: Approximation to solution of (2), $\boldsymbol{\mu}_r^*$

1: Set $\mathcal{S}_{-1}^{\boldsymbol{\mu}} = \{\}$ and $\mathcal{S}_{-1}^{\mathbf{w}} = \{\}$
2: Define $\boldsymbol{\mu}_{-1}^* = \boldsymbol{\mu}_0^{(0)}$
3: **for** $j = 0, 1, \ldots, N_{\max}$ **do**
4:      Compute $\mathbf{w}(\boldsymbol{\mu}_{j-1}^*)$ and $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu}_{j-1}^*)$ (sample HDM)
5:      Set $\mathcal{S}_j = \mathcal{S}_{j-1}^{\boldsymbol{\mu}} \cup \{\boldsymbol{\mu}_{j-1}^*\}$ and $\mathcal{S}_j^{\mathbf{w}} = \mathcal{S}_{j-1}^{\mathbf{w}} \cup \{\mathbf{w}(\boldsymbol{\mu}_{j-1}^*)\}$
6:      **if** j == 0 **then**
7:          Apply Algorithm 2 to state and sensitivity snapshots, $\mathbf{w}(\boldsymbol{\mu}_{j-1}^*)$ and $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu}_{j-1}^*) \to \boldsymbol{\Phi}$
8:      **else**
9:          Compute ϵ_j using (29)
10:         Apply Algorithm 3 to update the ROM, $\boldsymbol{\Phi}$ with additional state and sensitivity snapshots $\mathbf{w}(\boldsymbol{\mu}_{j-1}^*)$ and $\frac{\partial \mathbf{w}}{\partial \boldsymbol{\mu}}(\boldsymbol{\mu}_{j-1}^*) \to \boldsymbol{\Phi}$
11:     **end if**
12:     Compute $\boldsymbol{\mu}_j^{(0)}$ using (27)
13:     Solve (22) with ROB $\boldsymbol{\Phi}$ and residual upper bound ϵ_j; initial condition and offset, $\bar{\mathbf{w}}$, determined via (28) for each ROM solve (use Algorithm 3 with $\mathbf{Y}_{\text{state}} = \mathbf{Y}_{\text{sens}} = \emptyset$ to update the ROB with the new $\bar{\mathbf{w}}$ when necessary) $\to \boldsymbol{\mu}_j^*$
14:     **if** $||\boldsymbol{\mu}_j^* - \boldsymbol{\mu}_{j-1}^*||_2 < \delta ||\boldsymbol{\mu}_j^*||_2$ **then**
15:         $\boldsymbol{\mu}_r^* = \boldsymbol{\mu}_j^*$
16:         break
17:     **end if**
18: **end for**

---

5. APPLICATIONS

Here, the approach to PDE-constrained optimization using a progressively-constructed parametric ROM described in Section 4 is illustrated with the solution of an aerodynamic *shape* optimization problem. Specifically, pressure distribution discrepancy in the subsonic regime is used to recover the shape of the RAE2822 airfoil starting from that of the NACA0012 airfoil.

5.1. Shape parametrization and problem setup

A plethora of shape parametrization techniques exist [54, 55], each with strengths and weaknesses. They typically trade-off between efficiency and flexibility. A subset of these techniques has been studied in the context of model order reduction [13]. In this work, the SDESIGN software [56, 57], based on the design element approach [58, 59], is used for shape parameterization. In this approach, an initial shape is mapped onto a set of design (or control) elements with nodal displacement degrees of freedom. These are used to morph the given shape into a different one by applying to it a displacement field represented by shape functions generated by the design elements and their nodal degrees of freedom. Here, a single "cubic" design element is used to parametrize the NACA0012 airfoil using this approach. Such a design element has 8 control nodes. They are used to define cubic Lagrangian polynomials for describing the displacement fields along the horizontal edges of the element, and linear functions for representing the displacement fields along its vertical ones. For this application, the set of admissible shapes generated by this representation is restricted by constraining the control nodes to move in the *vertical* direction only. This results in a parameterization with 8 variables where each of them represents the *displacement* of a control node in the vertical direction. The case where all parameters are equal, $\boldsymbol{\mu} = c\mathbf{1}$ for $c \in \mathbb{R}$, corresponds to a rigid translation in the vertical direction. Because such a translation does not affect the definition of a shape, it is eliminated

by constraining one of the displacement variables to zero. Furthermore, because the control nodes are allowed to move only in the vertical direction, rigid rotations are automatically eliminated.

A visualization of the vertices of the design element and the deformation induced by perturbing each design variable is given in Figure 1. While SDESIGN is used to deform the surface nodes of the airfoil, a robust mesh motion algorithm based on a structural analogy is used to deform the surrounding body-fitted CFD mesh accordingly.

(a) $\boldsymbol{\mu}(1) = 0.1$ (b) $\boldsymbol{\mu}(2) = 0.1$

(c) $\boldsymbol{\mu}(3) = 0.1$ (d) $\boldsymbol{\mu}(4) = 0.1$

(e) $\boldsymbol{\mu}(5) = 0.1$ (f) $\boldsymbol{\mu}(6) = 0.1$

(g) $\boldsymbol{\mu}(7) = 0.1$ (h) $\boldsymbol{\mu}(8) = 0.1$
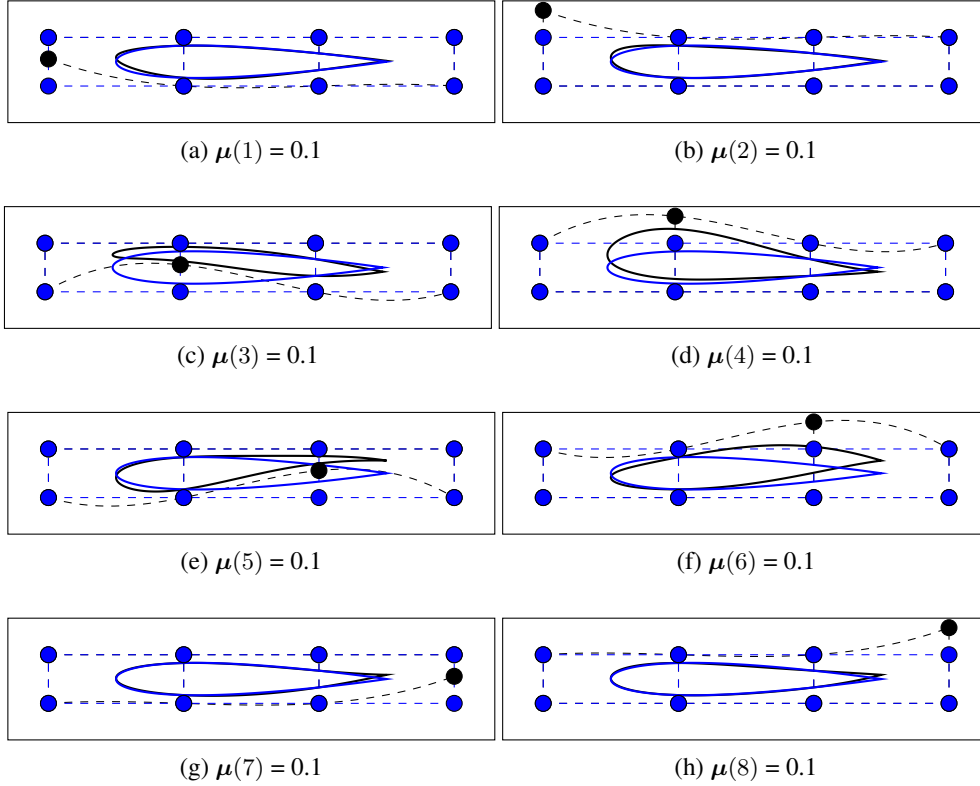
Figure 1. Shape parametrization of a NACA0012 airfoil using a *cubic* design element (the notation $\boldsymbol{\mu}(i)$ designates the i-th component of the vector $\boldsymbol{\mu}$ which refers to the i-th displacement degree of freedom of the shape parameterization)

The flow over the airfoil is modeled using the compressible Euler equations, and these are solved numerically using AERO-F [60]. Because this flow solver is three-dimensional, the two-dimensional fluid domain around the airfoil is represented as a slice of a three-dimensional domain. This slice is discretized using a body-fitted CFD mesh with 54816 tetrahedra and 19296 nodes (Figure 2a). Specifically, the flow equations are semi-discretized by AERO-F on this CFD mesh using a second-order finite volume method based on Roe's flux [61].

For each airfoil configuration generated during the iterative optimization procedure, the steady state solution of the flow problem is computed iteratively using pseudo-time-integration. For this purpose, each sought-after steady state solution is initialized using the *best* previously computed steady state solution available in the database[†]. The best steady state solution is defined here as that steady state solution available in the database which, for the given airfoil configuration, minimizes the residual of the discretized steady state Euler equations. Because the database of steady state flow solutions is initially empty, the iterative computation of the steady state flow over the initial shape — in this case, that of the NACA0012 airfoil — is initialized with the uniform flow solution.

[†]In this context, the database refers to the flow solutions computed for all shapes previously visited by the optimization trajectory.

(a) CFD mesh for the NACA0012 airfoil
(undeformed, 19296 nodes)

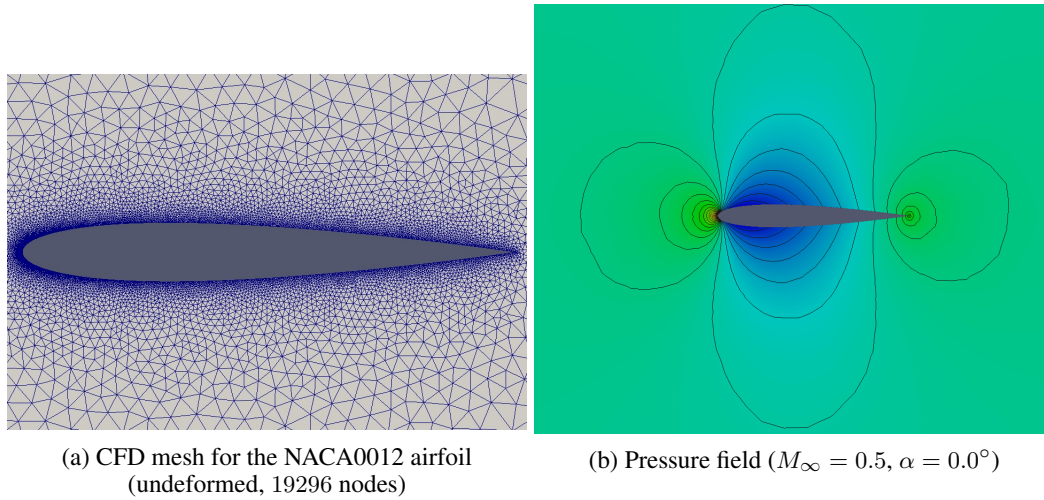(b) Pressure field ($M_\infty = 0.5$, $\alpha = 0.0°$)

Figure 2. NACA0012 mesh and pressure distribution at Mach 0.5 and zero angle of attack

The framework for model order reduction described in Section 3.2 is used to solve the aerodynamic shape optimization problem. At each HDM sample, the steady state solution and sensitivities with respect to shape parameters are computed and used as snapshots. As the chosen shape parameterization has 8 parameters, 9 snapshots are generated per HDM sample: one snapshot corresponding to the steady state solution, and 8 to the sensitivity of this solution with respect to all 8 shape parameters. A ROB is extracted from these snapshots using the state-sensitivity variant of the POD method described in Section 3.3. Because very few snapshots are generated for this problem, the truncation step in the POD algorithm (Algorithm 1) is skipped. Consequently, the size of the constructed ROB is $k_y = 9s$, where s is the number of sampled HDMs.

Each ROM problem is solved using the Gauss-Newton method equipped with a backtracking linesearch algorithm to minimize (11). For this purpose, the initial condition and reference state vector are chosen as in (28). The python interface to the SNOPT [62] software, pyOpt [63], is used to solve the optimization problem itself.

At this point, it is noted that since the exact profile of the RAE2822 airfoil does not lie in the space of admissible airfoil profiles defined by the cubic design element parametrization described above, it is approximated by the closest admissible profile. This approximation is referred to in the remainder of this section as the Cub-RAE2822 airfoil. It is graphically depicted in Figure 3 which also shows the pressure isolines computed for this airfoil at the free-stream Mach number $M_\infty = 0.5$ and angle of attack $\alpha = 0.0°$.

5.2. Subsonic inverse design

The free-stream conditions of interest are set to the subsonic Mach number $M_\infty = 0.5$ and zero angle of attack ($\alpha = 0°$), and the following optimization problem is considered

$$\begin{aligned}
\underset{\mathbf{w} \in \mathbb{R}^N, \, \boldsymbol{\mu} \in \mathcal{D}}{\text{minimize}} \quad & \frac{1}{2}||\mathbf{p}(\mathbf{w}(\boldsymbol{\mu})) - \mathbf{p}(\mathbf{w}(\boldsymbol{\mu}^{\text{RAE2822}}))||_2^2 \\
\text{subject to} \quad & \boldsymbol{\mu}(3) = 0 \\
& \boldsymbol{\ell} \leq \boldsymbol{\mu} \leq \mathbf{u} \\
& \mathbf{R}(\mathbf{w}, \boldsymbol{\mu}) = 0
\end{aligned} \quad\quad (30)$$

where $\mathbf{p}(\mathbf{w})$ is the vector of nodal pressures, and $\boldsymbol{\mu}^{\text{RAE2822}}$ designates the parameter solution vector morphing the NACA0012 airfoil into the Cub-RAE2822 airfoil. The first constraint is introduced to eliminate the rigid body translation in the vertical direction as discussed in the previous section. The box constraints are introduced to prohibit the optimization trajectory from going through highly distorted shapes for which it is difficult to compute steady state flow solutions.

(a) CFD mesh for the Cub-RAE2822 airfoil   (b) Pressure field ($M_\infty = 0.5$, $\alpha = 0.0^\circ$)
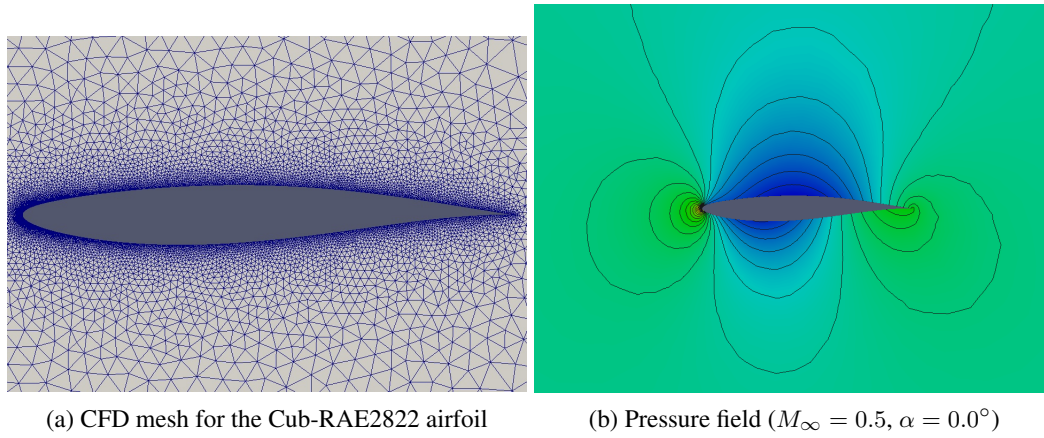
Figure 3. Cub-RAE2822 mesh and pressure isolines computed at Mach 0.5 and zero angle of attack

To obtain a reference solution that can be used for assessing the performance of the proposed ROM-based optimization method, problem (30) is first solved using the HDM as the constraining PDE. In this case, the optimizer is found to reduce the initial value of the objective function by 9 orders of magnitude, before numerical difficulties cause it to terminate (see Figure 4). Relevant statistics associated with this HDM-based reference solution of the optimization problem are gathered in Table I. Essentially, 24 optimization iterations are required for obtaining a solution $\boldsymbol{\mu}^{\text{RAE2822}}$ with a relative error well below 0.1%. These iterations incur a total of 29 HDM queries (including those associated with the linesearch iterations). Figure 5 shows that the pressure coefficient function associated with this reference solution matches very well the target pressure coefficient function.
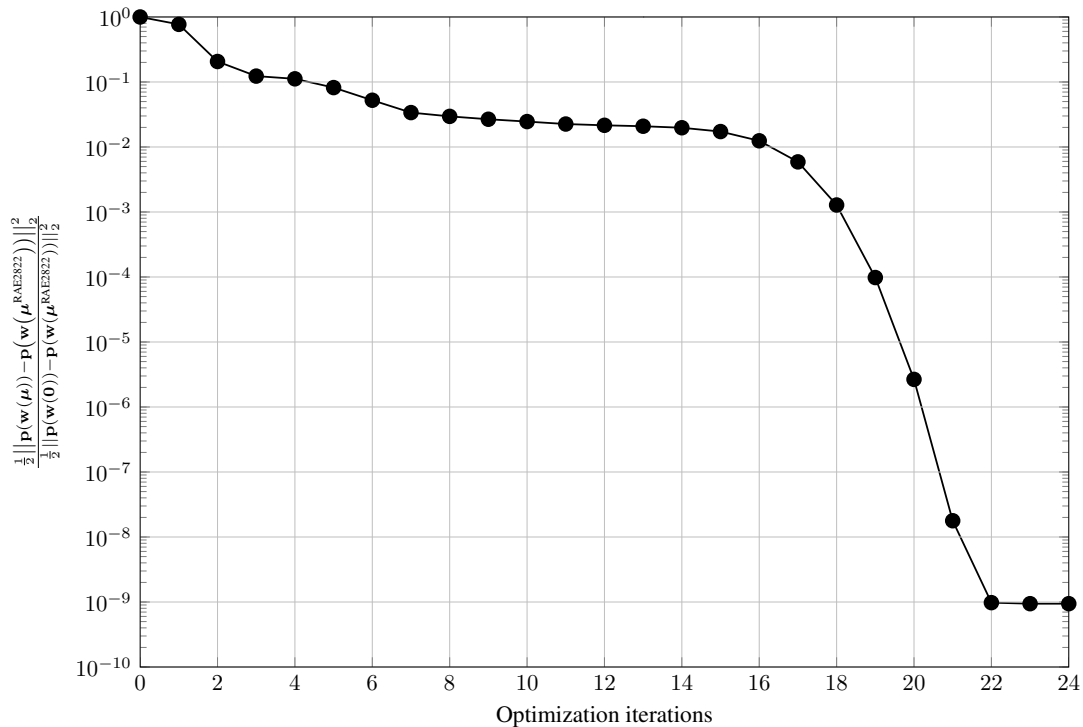
Figure 4. Progression of the objective function during the HDM-based optimization. The initial guess is defined as the 0th optimization iteration.
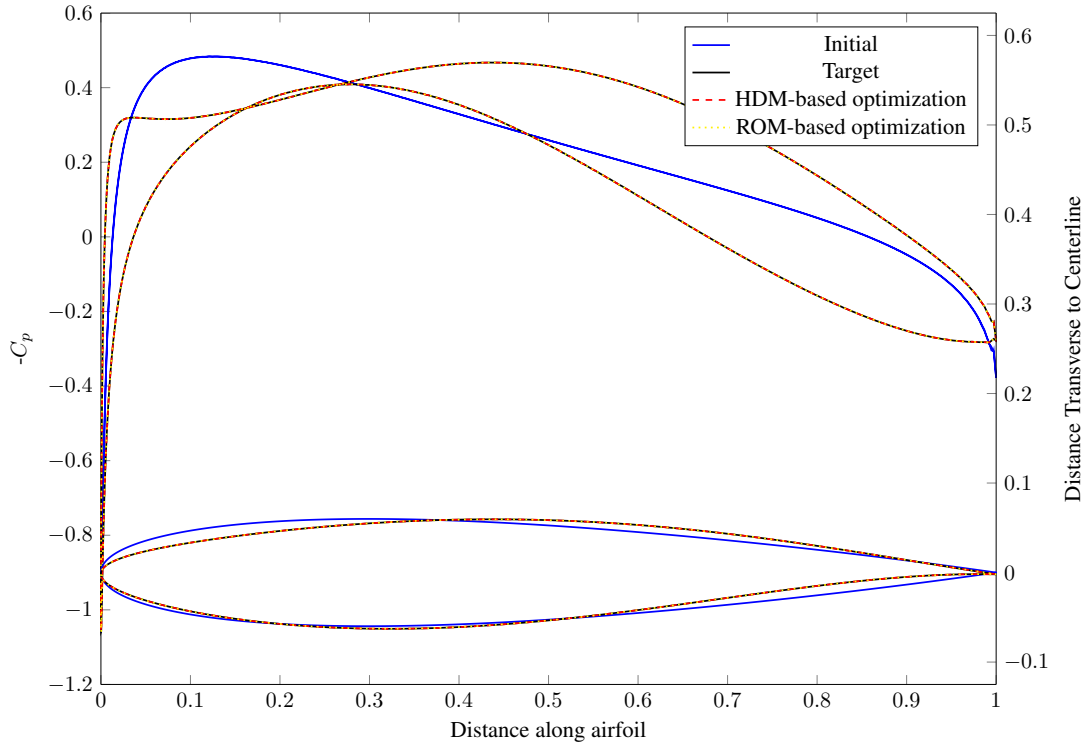
Figure 5. Subsonic inverse design of the airfoil Cub-RAE2822: initial shape (NACA0012) and associated C_p function, and final shape (Cub-RAE2822) and associated C_p functions delivered by the HDM- and ROM-based optimizations, respectively

Next, the framework for ROM-constrained optimization described in Section 4, which is summarized in Algorithm 4, is applied to the solution of problem (31) via the solution of a sequence of reduced optimization problems of the form

$$
\begin{aligned}
\underset{\mathbf{w}\in\mathbb{R}^N,\ \boldsymbol{\mu}\in\mathcal{D}}{\text{minimize}} \quad & \frac{1}{2}||\mathbf{p}(\bar{\mathbf{w}}+\boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\mu})) - \mathbf{p}(\mathbf{w}(\boldsymbol{\mu}^{\text{RAE2822}}))||_2^2 \\
\text{subject to} \quad & \boldsymbol{\mu}(3) = 0 \\
& \boldsymbol{\ell} \le \boldsymbol{\mu} \le \mathbf{u} \\
& \boldsymbol{\Psi}^T\mathbf{R}(\bar{\mathbf{w}}+\boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\mu}),\boldsymbol{\mu}) = 0 \\
& \frac{1}{2}||\mathbf{R}(\bar{\mathbf{w}}+\boldsymbol{\Phi}\mathbf{y}(\boldsymbol{\mu}),\boldsymbol{\mu})||_2^2 \le \epsilon.
\end{aligned}
\tag{31}
$$

The HDM is sampled at the initial configuration and the resulting 9 snapshots are used to build a ROB using the state-sensitivity POD method summarized in Algorithm 2, without truncation. The resulting ROB is used to solve (31). Indeed, as the minimum-error sensitivity computation described in Section 4.1 is not consistent with the true reduced sensitivities for large residuals, convergence of the optimization problem is not guaranteed. To address this issue, an upper bound is set on the number of optimization iterations (25 in this case) and the goal of the reduced optimization problem is set to finding an improvement to the current solution before updating the ROB. The HDM is also sampled at the termination point of each reduced optimization problem yielding 9 additional snapshots which are appended to the ROB using Algorithm 3. Linear independence of the basis is maintained by truncating vectors with corresponding singular values below some tolerance. For the present application, such truncation was not necessary as the snapshots added to the ROB at a given iteration were not contained in the span of the snapshots from previous iterations. When solving for the flow using a given ROM, the initial guess and snapshot translation vector are defined

in Section 4.4 as the best HDM sample. Additionally, an aggressive choice of $\tau = 0.1$ was used for the nonlinear trust region radius bound adaptation in (29).

Using only 7 HDM samples, the progressive ROM optimization framework reduces the initial pressure discrepancy by 18 orders of magnitude, to essentially machine zero. Interestingly, this is 4 times fewer HDM queries than required by the HDM-based optimization. Figure 5 shows that both the shape of the Cub-RAE2822 airfoil and the associated pressure coefficient at the chosen free-stream conditions match the target shape and pressure coefficient function very well.

Surprisingly, the ROM-based optimization process achieves a *lower* value of the objective function than the HDM-based one. This can be traced to convergence tolerance on the HDM sensitivity analysis. The HDM-based sensitivities are obtained by solving the multiple right-hand side linear system of equations in (4) using GMRES. The convergence tolerance is $||\mathbf{Ax} - \mathbf{b}||_2 \leq \gamma ||\mathbf{b}||_2$ for solving the linear system of equations $\mathbf{Ax} = \mathbf{b}$, with $\gamma = 10^{-10}$ in this case. If $||\mathbf{b}||$ is large ($\mathbf{b} = \dfrac{\partial \mathbf{R}}{\partial \boldsymbol{\mu}}$ in this case), the convergence requirement may be rather flexible. Conversely, the ROM-based sensitivities in (19) are solved directly using the QR factorization, where no tolerances are involved. Recall from Section 4.1 that the minimum-error reduced sensitivities approach the true sensitivities for LSPG projection as the HDM residual approaches zero. Figure 8 verifies that the HDM residual is small after 6 HDM samples are taken, which implies the minimum-error ROM sensitivities are (nearly) consistent with the true ROM sensitivities. This consistency will guarantee convergence of the reduced optimization problem when using a globally-convergent optimization solver. Additionally, the small HDM residual implies that the ROM is highly accurate in this region, making it likely that the reduced optimization problem will converge to a point close to the true optimum.

Figure 6 reports on the evolution of the objective function with the number of performed optimization iterations, and marks each new HDM query along the optimization trajectory. The reader can observe that the proposed ROM-based optimization method performs a total of 160 iterations (Figure 7) requiring 346 ROM evaluations (see Table I) and 7 HDM queries. From a computational complexity viewpoint, this compares favorably with the 24 HDM-based optimization iterations requiring 29 HDM queries (see Table I).

Figure 7 graphically depicts the progression of the *reduced* objective function across all reduced optimization problems using a dashed line to indicate a new HDM sample and a subsequent update of the ROB. For each optimization problem, it also reports the size of the ROB/ROM.

Figure 8 shows the evolution of the HDM residual evaluated at the solution of the ROM — which is an indicator of the ROM error — across all reduced optimization problems, along with the upper bound ϵ, defined in (31) and adapted between reduced optimization problems according to (29). It is noted here that it is common practice in nonlinear programming software to allow violation of nonlinear constraints during an optimization procedure, which explains the residual bound violation seen in this figure. Figure 8 also shows that the ROM solution coincides with the HDM solution at the initial condition of each optimization problem, as expected from the choice of the reference vector $\bar{\mathbf{w}}$ discussed in Section 4.4. In the first few reduced optimization problems where the iterate solutions are far from the optimal solution, the residual grows as the iterates move into areas of the parameter space away from HDM samples. However, near the optimal solution, the residual remains small as the optimization iterates remain in a small neighborhood of the most recent HDM sample.

Remark. There are two mechanisms that prevent the reduced optimization problem from venturing into regions of the parameter space where it lacks accuracy: (1) the objective function, and (2) the nonlinear trust region. In the present inverse design example, the objective function is mostly sufficient to keep the ROM in regions of accuracy, as can be seen from Figure 8 where the trust region bound is only reached once and the upper bound always increases. For other objective functions such as drag or downforce, the nonlinear trust region will be necessary as it is likely that an inaccurate ROM can predict a lower value in such objective functions than is actually present. In practice, inaccurate ROMs have been observed to predict the unphysical situation of negative drag (i.e. thrust), which motivates the need for the residual-based trust region.

	HDM-based optimization	ROM-based optimization
# of HDM Evaluations	29	7[‡]
# of ROM Evaluations	-	346
$\dfrac{\|\boldsymbol{\mu}^* - \boldsymbol{\mu}^{RAE2822}\|}{\|\boldsymbol{\mu}^{RAE2822}\|}$	$2.28 \times 10^{-3}\%$	$4.17 \times 10^{-6}\%$

Table I. Performance of the HDM- and ROM-based optimization methods

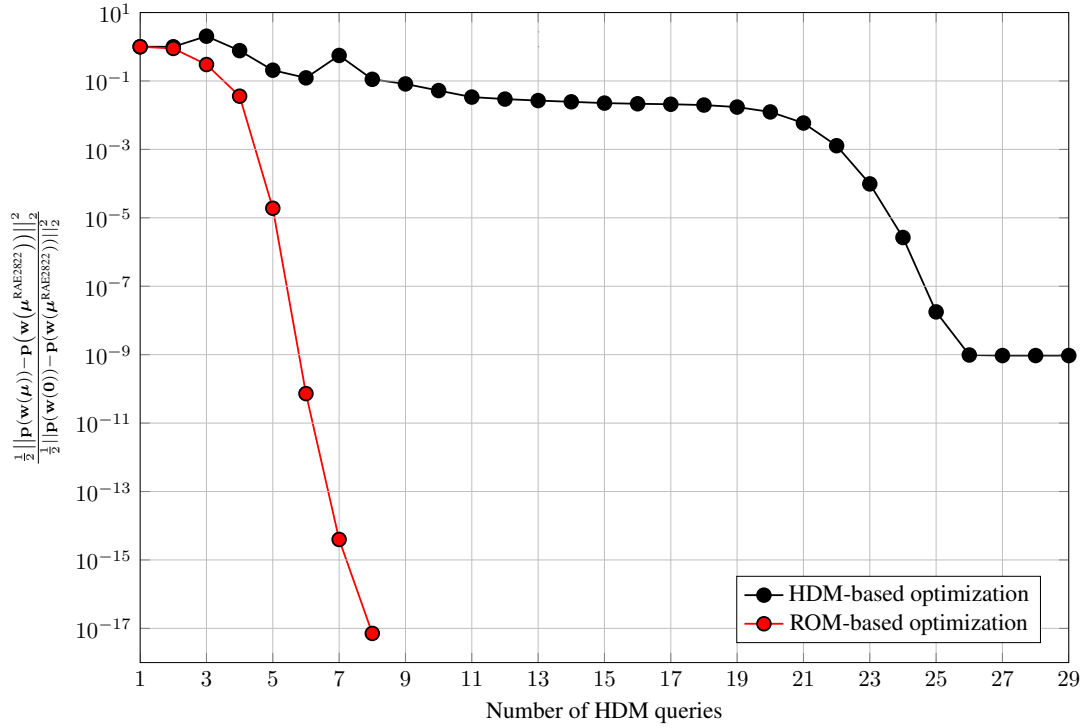Figure 6. Objective function versus number of queries to the HDM: ROM-based optimization (red) and HDM-based optimization (black)

[‡]The last HDM sample in Figure 6 was not included in this count as the residual-based error indicator is small at this configuration (Figure 8). A similar argument could also be made for the 7th HDM sample.
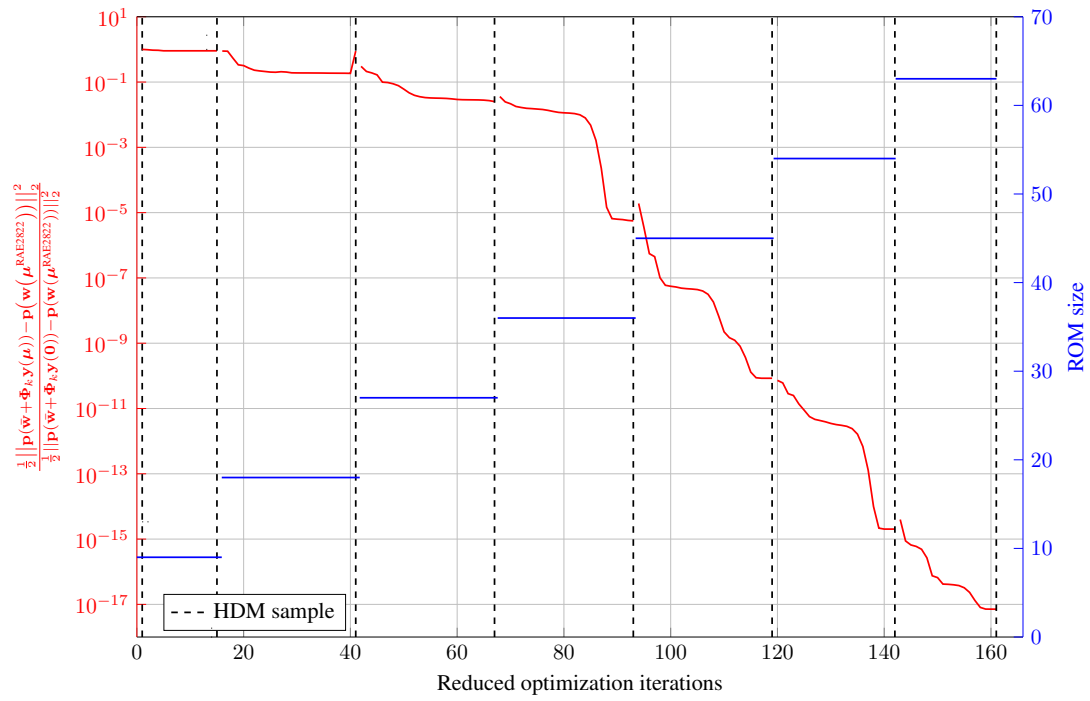
Figure 7. Progression of *reduced* objective function: dashed line indicates an HDM sample and a subsequent update of the ROB
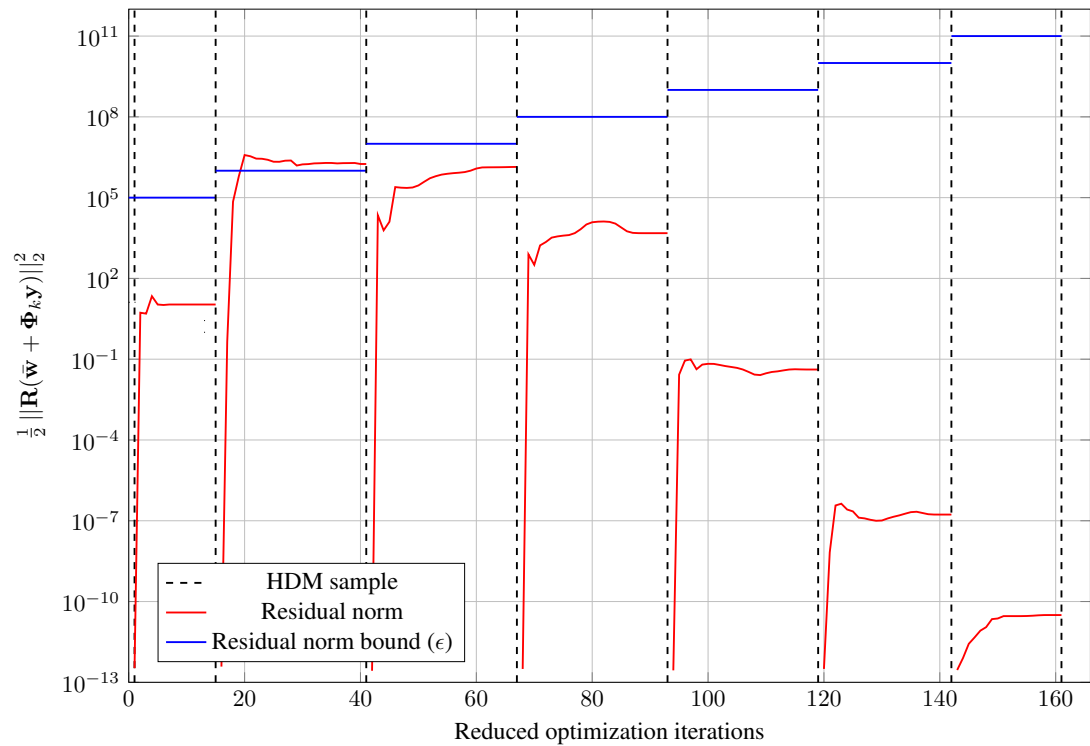
Figure 8. Progression of HDM residual: dashed line indicates an HDM sample and a subsequent update of the ROB

6. CONCLUSIONS

The proposed adaptive approach for using Reduced-Order Models (ROMs) as surrogate models in PDE-constrained optimization breaks the traditional offline-online framework of model order reduction. It builds a Reduced-Order Basis (ROB) using a variant of the POD method that incorporates both state and sensitivity snapshots. This is because the low-dimensionality assumption of model reduction implicitly assumes that the ROB can approximately represent the state vector and its sensitivities. The resulting ROM is used as a surrogate for the HDM in the optimization problem and a nonlinear trust region constraint is included to restrict the growth of a residual-based error indicator. The adaptive approach samples the HDM at the solution of this reduced optimization problem and uses the new information to update the ROB and define a new reduced optimization problem.

The minimum-error reduced sensitivity framework introduced in this paper offers the advantage of returning the best approximation to the HDM sensitivities that is available in the given ROB, for some norm. In the context of the Least-Squares Petrov-Galerkin projection, the minimum-error reduced sensitivities approach the true reduced sensitivities as the error in the ROM approaches zero. The proposed minimum-error reduced sensitivity framework ensures that the approximation of the sensitivities cannot be degraded by state vector snapshots in the ROB.

Most importantly, the proposed progressive approach to ROM-constrained optimization avoids the difficult task of building a single *global* ROB that is accurate throughout the parameter space. Because it also avoids sampling in regions of the parameter space that have not been visited during the optimization procedure, it requires fewer queries to the HDM than traditional alternatives. A factor of 4 fewer HDM queries were observed on a problem from aerodynamic shape optimization, where the optimal solution was recovered to machine precision. Transforming this computational complexity advantage into a CPU advantage requires however equipping this proposed approach for ROM-based optimization with hyperreduction, a step which is the subject of on-going work.

A. LOW-RANK SVD UPDATE ALGORITHMS

Algorithm 5 Brand's Algorithm for low-rank SVD updates: appending vector

---

Input: Data matrix, $\mathbf{X} \in \mathbb{R}^{m \times n}$ of rank r; thin SVD of data matrix, $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$; and full-rank matrices of vectors to append data matrix, $\mathbf{Y} \in \mathbb{R}^{m \times k}$.

Output: SVD of updated data matrix: $\begin{bmatrix} \mathbf{X} & \mathbf{Y} \end{bmatrix} = \bar{\mathbf{U}}\bar{\mathbf{\Sigma}}\bar{\mathbf{V}}^T$

1: Compute $\mathbf{M} = \mathbf{U}^T\mathbf{Y} \in \mathbb{R}^{r \times k}$
2: Compute $\bar{\mathbf{P}} = \mathbf{Y} - \mathbf{U}\mathbf{M} \in \mathbb{R}^{m \times k}$
3: Compute QR decomposition of $\bar{\mathbf{P}} = \mathbf{P}\mathbf{R}_A$, where $\mathbf{P} \in \mathbb{R}^{m \times k}$, $\mathbf{R}_A \in \mathbb{R}^{k \times k}$
4: Form $\mathbf{K} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{M} \\ \mathbf{0} & \mathbf{R}_A \end{bmatrix} \in \mathbb{R}^{(r+k) \times (r+k)}$
5: Compute SVD of $\mathbf{K} = \mathbf{C}\mathbf{S}\mathbf{D}^T$, where $\mathbf{C}, \mathbf{S}, \mathbf{D} \in \mathbb{R}^{(r+k) \times (r+k)}$
6: $\bar{\mathbf{U}} = \begin{bmatrix} \mathbf{U} & \mathbf{P} \end{bmatrix} \mathbf{C}$
7: $\bar{\mathbf{\Sigma}} = \mathbf{S}$
8: $\bar{\mathbf{V}} = \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{D}$

REFERENCES

1. L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, *Large-scale PDE-constrained optimization: an introduction*. Springer, 2003.
2. K. Ito and S. Ravindran, "A reduced-order method for simulation and control of fluid flows," *Journal of computational physics*, vol. 143, no. 2, pp. 403–425, 1998.

Algorithm 6 Brand's Algorithm for low-rank SVD updates: translating columns

---

Input: Data matrix, $\mathbf{X} \in \mathbb{R}^{m \times n}$ of rank r; thin SVD of data matrix, $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$; and desired translation vector, $\mathbf{a} \in \mathbb{R}^m$

Output: SVD of updated data matrix: $\mathbf{X} + \mathbf{a}\mathbf{1}^T = \bar{\mathbf{U}}\bar{\mathbf{\Sigma}}\bar{\mathbf{V}}^T$

1: Compute $\mathbf{n} = \mathbf{V}^T\mathbf{1}$, $\mathbf{q} = \mathbf{1} - \mathbf{V}\mathbf{n}$, $q = ||\mathbf{q}||_2$, and $\mathbf{Q} = \frac{1}{q}\mathbf{q}$
2: Compute $\mathbf{m} = \mathbf{U}^T\mathbf{a} \in \mathbb{R}^r$
3: Compute $\mathbf{p} = \mathbf{a} - \mathbf{U}\mathbf{m} \in \mathbb{R}^m$
4: Define $\hat{r} \in \mathbb{R}$ and $\mathbf{v} \in \mathbb{R}^N$ such that: $\mathbf{p} = \hat{r}\mathbf{v}$ where $||\mathbf{v}||_2 = 1$
5: Form $\mathbf{K} = \begin{bmatrix} \mathbf{\Sigma} + \mathbf{m}\mathbf{n}^T & q\mathbf{m} \\ \hat{r}\mathbf{n} & \hat{r}q \end{bmatrix} \in \mathbb{R}^{(r+1) \times (r+1)}$
6: Compute SVD of $\mathbf{K} = \mathbf{C}\mathbf{S}\mathbf{D}^T$, where $\mathbf{C}, \mathbf{S}, \mathbf{D} \in \mathbb{R}^{(r+1) \times (r+1)}$
7: $\bar{\mathbf{U}} = \begin{bmatrix} \mathbf{U} & \mathbf{v} \end{bmatrix}\mathbf{C}$
8: $\bar{\mathbf{\Sigma}} = \mathbf{S}$
9: $\bar{\mathbf{V}} = \begin{bmatrix} \mathbf{V} & \mathbf{Q} \end{bmatrix}\mathbf{D}$

---

3. S. Ravindran, "Reduced-order adaptive controllers for fluid flows using pod," *Journal of scientific computing*, vol. 15, no. 4, pp. 457–478, 2000.
4. K. Afanasiev and M. Hinze, "Adaptive control of a wake flow using proper orthogonal decomposition," *Lecture Notes in Pure and Applied Mathematics*, pp. 317–332, 2001.
5. M. Fahl, *Trust-region methods for flow control based on reduced order modelling*. PhD thesis, Universitätsbibliothek, 2001.
6. M. Bergmann, L. Cordier, and J.-P. Brancher, "Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model," *Physics of Fluids*, vol. 17, no. 9, pp. 097101–097101, 2005.
7. M. Bergmann and L. Cordier, "Optimal control of the cylinder wake in the laminar regime by trust-region methods and pod reduced-order models," *Journal of Computational Physics*, vol. 227, no. 16, pp. 7813–7840, 2008.
8. K. Kunisch and S. Volkwein, "Proper orthogonal decomposition for optimality systems," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 42, no. 01, pp. 1–23, 2008.
9. M. A. Grepl, N. C. Nguyen, K. Veroy, A. T. Patera, and G. R. Liu, "Certified rapid solution of partial differential equations for real-time parameter estimation and optimization," in *Proceedings of the 2nd Sandia workshop of PDE-constrained optimization: Real-time PDE-constrained optimization. SIAM computational science and engineering book series. SIAM, Philadelphia*, pp. 197–216, 2007.
10. D. Amsallem, J. Cortial, and C. Farhat, "Towards real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information," *AIAA Journal*, vol. 48, no. 9, pp. 2029–2037, 2010.
11. U. Hetmaniuk, R. Tezaur, and C. Farhat, "Review and assessment of interpolatory model order reduction methods for frequency response structural dynamics and acoustics problems," *International Journal for Numerical Methods in Engineering*, vol. 90, pp. 1636–1662, 2012.
12. P. A. LeGresley and J. J. Alonso, "Airfoil design optimization using reduced order models based on proper orthogonal decomposition," in *Fluids 2000 conference and exhibit, Denver, CO*, 2000.
13. G. Rozza and A. Manzoni, "Model order reduction by geometrical parametrization for shape optimization in computational fluid dynamics," in *Proceedings of ECCOMAS CFD*, 2010.
14. T. Lassila and G. Rozza, "Parametric free-form shape design with pde models and reduced basis method," *Computer Methods in Applied Mechanics and Engineering*, vol. 199, no. 23, pp. 1583–1592, 2010.
15. A. Manzoni, A. Quarteroni, and G. Rozza, "Shape optimization for viscous flows by reduced basis methods and free-form deformation," *International Journal for Numerical Methods in Fluids*, vol. 70, no. 5, pp. 646–670, 2012.
16. T. Bui-Thanh and K. Willcox, "Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications," *AIAA Journal*, vol. 46, pp. 2520–2529, October 2008.
17. L. Sirovich, "Turbulence and the dynamics of coherent structures. i-coherent structures. ii-symmetries and transformations. iii-dynamics and scaling," *Quarterly of applied mathematics*, vol. 45, pp. 561–571, 1987.
18. K. Washabaugh, "Personal communication." 2013.
19. D. Ryckelynck, "A priori hyperreduction method: an adaptive approach," *Journal of Computational Physics*, vol. 202, no. 1, pp. 346–366, 2005.
20. M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, "An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations," *Comptes Rendus Mathematique*, vol. 339, no. 9, pp. 667–672, 2004.
21. S. Chaturantabut and D. C. Sorensen, "Nonlinear model reduction via discrete empirical interpolation," *SIAM Journal on Scientific Computing*, vol. 32, no. 5, pp. 2737–2764, 2010.
22. K. Carlberg, C. Bou-Mosleh, and 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*, vol. 86, no. 2, pp. 155–181, 2011.
23. K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, "The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows," *Journal of Computational Physics*, vol. 242, pp. 623—647, 2013.

---

24. C. Farhat, P. Avery, T. Chapman, and 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*, vol. 98, no. 9, pp. 625–662, 2014.
25. R. Everson and L. Sirovich, "Karhunen–loeve procedure for gappy data," *JOSA A*, vol. 12, no. 8, pp. 1657–1664, 1995.
26. I. Oliveira and A. Patera, "Reduced-basis techniques for rapid reliable optimization of systems described by affinely parametrized coercive elliptic partial differential equations," *Optimization and Engineering*, vol. 8, no. 1, pp. 43–65, 2007.
27. T. Bui-Thanh, K. Willcox, and O. Ghattas, "Model reduction for large-scale systems with high-dimensional parametric input space," *SIAM Journal on Scientific Computing*, vol. 30, no. 6, pp. 3270–3288, 2008.
28. K. Urban, S. Volkwein, and O. Zeeb, "Greedy sampling using nonlinear optimization," in *Reduced Order Methods for modeling and computational reduction*, pp. 137–157, Springer, 2014.
29. M. Hinze and U. Matthes, "Model order reduction for networks of ode and pde systems," in *System Modeling and Optimization*, pp. 92–101, Springer, 2013.
30. D. Huynh and A. Patera, "Reduced basis approximation and a posteriori error estimation for stress intensity factors," *International Journal for Numerical Methods in Engineering*, vol. 72, no. 10, pp. 1219–1259, 2007.
31. K. Carlberg and C. Farhat, "A compact proper orthogonal decomposition basis for optimization-oriented reduced-order models," in *The 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, vol. 5964, 10 - 12 September 2008.
32. G. Rozza, D. Huynh, and A. T. Patera, "Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations," *Archives of Computational Methods in Engineering*, vol. 15, no. 3, pp. 229–275, 2008.
33. K. Carlberg and C. Farhat, "A low-cost, goal-oriented compact proper orthogonal decompositionbasis for model reduction of static systems," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 3, pp. 381–402, 2011.
34. B. Epureanu, "A parametric analysis of reduced order models of viscous flows in turbomachinery," *Journal of Fluids and Structures*, vol. 17, no. 7, pp. 971 – 982, 2003.
35. T. Lieu, C. Farhat, and M. Lesoinne, "Reduced-order fluid/structure modeling of a complete aircraft configuration," *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 41, pp. 5730–5742, 2006.
36. T. Lieu and C. Farhat, "Adaptation of aeroelastic reduced-order models and application to an F-16 configuration," *AIAA Journal*, vol. 45, no. 6, pp. 1244–1257, 2007.
37. A. Hay, J. T. Borggaard, and D. Pelletier, "Local improvements to reduced-order models using sensitivity analysis of the proper orthogonal decomposition," *Journal of Fluid Mechanics*, vol. 629, pp. 41–72, 2009.
38. D. Amsallem and C. Farhat, "Interpolation method for adapting reduced-order models and application to aeroelasticity," *AIAA Journal*, vol. 46, no. 7, pp. 1803–1813, 2008.
39. D. Amsallem, J. Cortial, K. Carlberg, and C. Farhat, "A method for interpolating on manifolds structural dynamics reduced-order models," *International Journal for Numerical Methods in Engineering*, vol. 80, no. 9, pp. 1241–1258, 2009.
40. M. Dihlmann and B. Haasdonk, "Certified pde-constrained parameter optimization using reduced basis surrogate models for evolution problems," *SimTech Preprint, University of Stuttgart*, 2013.
41. Y. Yue and K. Meerbergen, "Accelerating optimization of parametric linear systems by model order reduction," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 1344–1370, 2013.
42. M. J. Zahr, D. Amsallem, and C. Farhat, "Construction of parametrically-robust cfd-based reduced-order models for pde-constrained optimization," in *42nd AIAA Fluid Dynamics Conference and Exhibit*, (San Diego, CA), June 24 - 27 2013.
43. J. Nocedal and S. Wright, *Numerical optimization, series in operations research and financial engineering*. Springer, 2006.
44. K. Washabaugh, D. Amsallem, M. Zahr, and C. Farhat, "Nonlinear model reduction for cfd problems using local reduced order bases," in *42nd AIAA Fluid Dynamics Conference and Exhibit*, (New Orleans, LA), June 25-28 2012.
45. M. D. Gunzburger, *Perspectives in flow control and optimization*, vol. 5. Siam, 2003.
46. M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich, *Optimization with PDE constraints*. Springer, New York, 2009.
47. V. Akçelik, G. Biros, O. Ghattas, J. Hill, D. Keyes, and B. van Bloemen Waanders, "Parallel algorithms for pde-constrained optimization," *Frontiers of parallel computing. SIAM*, 2006.
48. C. Kelley and D. E. Keyes, "Convergence analysis of pseudo-transient continuation," *SIAM Journal on Numerical Analysis*, vol. 35, no. 2, pp. 508–523, 1998.
49. C. Kelley, L.-Z. Liao, L. Qi, M. T. Chu, J. Reese, and C. Winton, "Projected pseudo-transient continuation," 2007.
50. E. Arian, M. Fahl, and E. W. Sachs, "Trust-region proper orthogonal decomposition for flow control," tech. rep., DTIC Document, 2000.
51. J. L. Eftang, M. A. Grepl, A. T. Patera, and E. M. Rønquist, "Approximation of parametric derivatives by the empirical interpolation method," *Foundations of Computational Mathematics*, vol. 13, no. 5, pp. 763–787, 2013.
52. G. H. Golub and C. F. Van Loan, *Matrix computations*, vol. 3. JHU Press, 2012.
53. M. Brand, "Fast low-rank modifications of the thin singular value decomposition," *Linear algebra and its applications*, vol. 415, no. 1, pp. 20–30, 2006.
54. J. A. Samareh, "A survey of shape parameterization techniques," in *NASA CONFERENCE PUBLICATION*, pp. 333–344, Citeseer, 1999.
55. G. R. Anderson, M. J. Aftosmis, and M. Nemec, "Parametric deformation of discrete geometry for aerodynamic shape design," *AIAA Paper 2012*, vol. 965, 2012.
56. K. Maute, M. Nikbay, and C. Farhat, "Sensitivity analysis and design optimization of three-dimensional nonlinear aeroelastic systems by the adjoint method," *The International Journal for Numerical Methods in Engineering*, vol. 56, pp. 911–933, 2003.
57. K. Maute and M. Raulli, "Fem—optimization module and sdesign user guides, 0.2," June 2006.

58. M. H. Imam, "Three-dimensional shape optimization," *International Journal for Numerical Methods in Engineering*, vol. 18, no. 5, pp. 661–673, 1982.
59. G. E. Farin, *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Code*. Academic Press, Inc., 1996.
60. P. Geuzaine, G. Brown, C. Harris, and C. Farhat, "Aeroelastic dynamic analysis of a full F-16 configuration for various flight conditions," *AIAA Journal*, vol. 41, pp. 363–371, 2003.
61. P. L. Roe, "Approximate riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, vol. 43, no. 2, pp. 357 – 372, 1981.
62. P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM journal on optimization*, vol. 12, no. 4, pp. 979–1006, 2002.
63. R. E. Perez, P. W. Jansen, and J. R. R. A. Martins, "pyOpt: A Python-based object-oriented framework for nonlinear constrained optimization," *Structures and Multidisciplinary Optimization*, vol. 45, no. 1, pp. 101–118, 2012.