Space-local reduced-order bases for accelerating reduced-order models through sparsity

Spenser Anderson[a], Cristina White[b], Charbel Farhat[a,b,c,*]

[a]*Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305*
[b]*Department of Mechanical Engineering, Stanford University, Stanford, CA 94305*
[c]*Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305*

Abstract

Projection-based model order reduction (PMOR) methods based on linear or affine approximation subspaces accelerate numerical predictions by reducing the dimensionality of the underlying computational models. The state of the art of PMOR includes approximation methods based on state-local subspaces – that is, subspaces associated with different regions of the solution manifold – and methods based on adaptive reduced-order bases. For challenging applications such as those associated with highly nonlinear problems, such methods accelerate traditional PMOR by controling the dimension of the reduced-order basis and associated projection-based reduced-order model (PROM). This paper proposes an alternative as well as complementary approach for accelerating PMOR based on introducing sparsity into the reduced-order basis, in order to enhance the computational efficiency of the associated PROM. Specifically, the proposed approach introduces sparsity in PMOR by partitioning the computational domain rather than, or in addition to, the solution manifold, and therefore leads to the concept of a space-local reduced-order basis. This concept is compatible with both concepts of a state-local reduced-order basis and hyperreduction. It is demonstrated for two computational fluid dynamics problems in turbulent flow applications. Acceleration factors of the order of 1.5 relative to traditional PMOR and CPU time speedup factors of several orders of magnitude relative to high-dimensional models are reported.

Keywords: clustering, domain decomposition, machine learning, nonlinear model reduction, proper orthogonal decomposition

1. Introduction

Projection-based model order reduction (PMOR) methods are powerful techniques for reducing the dimensionality of parametric computational models and therefore their complexity. Their outcomes are referred to as projection-based reduced-order models (PROM)s. A well-designed PROM accelerates a parametric simulation grounded in a high-fidelity, high-dimensional computational model (HDM), by finding the best approximate solution in a lower-dimensional approximation manifold. By solving a much smaller set of equations formulated in terms of generalized coordinates, a PROM can produce an accurate solution several orders of magnitude faster than the underlying HDM. Because of this combination of accuracy and speed, PMOR is an enabling technology for many time-critical applications where parametric HDMs are often avoided or not even considered, when a single numerical simulation may require hours, days, or longer on expensive many-core computers. This is the case, for example, for simulation-based multidisciplinary design analysis and optimization, real-time interactive design, statistical analyses for uncertainty quantification, and real-time model predictive control.

During the last decade, the state of the art of nonlinear PMOR has significantly advanced. It has improved in robustness, online speed, and accuracy [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. For many PMOR methods and many applications however, the offline cost associated with training a PROM remains

*Corresponding author
Email addresses: aspenser@stanford.edu (Spenser Anderson), crwhite@stanford.edu (Cristina White), cfarhat@stanford.edu (Charbel Farhat)

excessive today, if not prohibitive; and the further acceleration of the online execution of a PROM continues to be desired. Recent effort on enhancing the performance of a PROM has focused on localized approximations, online adaptations, and nonlinear approximations for reducing further the dimension of the underlying reduced-order basis (ROB) [2, 11, 12, 13, 14, 15, 16, 17, 18, 9, 10], particularly for applications such as convection-dominated turbulent flows that are challenged by the Kolmogorov n-width [19] issue. Among these developments, the methods grounded in localized approximations such as piecewise affine approximations [2, 9] are referred to in this paper as PMOR methods with *state-local* ROBs, because the associated ROBs are local in state space. This is in contrast with the PMOR approach proposed in this paper, where a ROB has localized components in the ordinary $x-y-z$ space. However, both concepts are complementary and therefore can be combined.

Most current PMOR approaches are motivated by the reduction of the dimensionality of a ROB, in order to accelerate the performance of the associated PROM. However, the costs of most operations involving a ROB or its PROM do not necessarily scale directly with the dimension of these entities, if the ROB is sparse and stored as a sparse matrix. In this case, many operations such as matrix-matrix and matrix-vector multiplications scale with the number of nonzero entries of the ROB, rather than its dimension. Hence, within the context of nonlinear PMOR, this paper explores an approach for sparsifying a ROB even at the cost of increasing its dimension, in order to construct a faster but otherwise similarly accurate nonlinear PROM. The main idea is to introduce sparsity into a ROB by performing a nonoverlapping domain decomposition – or more specifically, a mesh partitioning – and constructing it from a set of *locally supported* basis components, each covering a different region of the computational domain – or more specifically, a different partition of the associated computational mesh. Hence, while state-local ROBs exploit the fact that locality in state space may allow smaller ROBs to perform well, the proposed approach exploits yet another form of locality common to many computational physics simulations, namely, spatial locality, to accelerate further PROM-based computations. It also exploits the fact that in many computational physics applications, the solution of a problem of interest may exhibit different behaviors in different regions of the computational domain. For example, in computational fluid dynamics (CFD), shocks, boundary layers, shock-boundary-layer interactions, flow separation, flames, and turbulent wakes may occur in different regions of the computational domain, where the various flow regimes may evolve at different timescales, and therefore the solution may be approximated best by different subspaces or approximation manifolds represented by independent ROBs. In such applications, a ROB that is constructed region-by-region has local spatial support and therefore is sparse. It typically requires a slightly larger dimensionality to deliver the same accuracy as a dense counterpart, for a reason that will be explained later: however, it may still lead to a faster PROM, thanks to sparse data structures and computations.

For applications where domain decomposition (or substructuring) is performed for various reasons, the collection of solution snapshots may be easier at the subdomain rather than domain level (for example, see [20, 21, 22]): in this case, the PMOR approach proposed in this paper is most suitable. In the limit where the dimension of a basis component associated with a region of the computational domain is set to the dimension of the trace of the solution on the corresponding submesh, the proposed concept of a space-local ROB recovers the idea of performing a domain decomposition and considering PROMs in some subdomains and HDMs in others, as appropriate [23, 24, 25, 26]. However, it does not necessarily recover the optimal implementation associated with this idea.

The work described in this paper is motivated by unsteady, convection-dominated, turbulent flow problems. For such problems, the Least Squares Petrov-Galerkin (LSPG) method [4, 5] equipped with the energy-conserving sampling and weighting (ECSW) hyperreduction method is a state-of-the-art nonlinear PMOR method [8, 9]. For these reasons, the proposed concept of a space-local ROB is fully developed here in the contexts of implicit time-discretization and nonlinear LSPG equipped with ECSW. However, it is emphasized that the main contributions of this paper are equally applicable to the Galerkin framework for PMOR and to second-order dynamical systems, including nonlinear structural dynamics problems with configuration-dependent mass matrices, nonlinear damping forces, and configuration-dependent external forces/moments. Furthermore, it is also emphasized that the main ideas presented in this paper and the associated numerical algorithms are independent from the parametric setting; and that their relative merits in terms of wall-clock time reduction for a desired level of accuracy can be assessed independently of any such setting. Hence, even though it is acknowledged that PMOR is

primarily beneficial for parametric problems, the proposed concept of a space-local ROB is presented and evaluated in this paper without reference to any parameter except time, which can be interpreted as the parameter of a one-dimensional parameter space.

The remainder of this paper is organized as follows. Section 2 reviews nonlinear PMOR, including Galerkin and Petrov-Galerkin projections, and proper orthogonal decomposition (POD) by the method of solution snapshots – or equivalently, the singular value decomposition (SVD) method – for the computation of ROBs. This section also comments on various aspects of the computational cost associated with performing nonlinear PMOR, particularly in the contexts specified above. Then, Section 3 introduces the concept of a space-local ROB, explains its computational benefits, highlights some of its properties, and presents numerical algorithms for computing a space-local ROB and exploiting it in PMOR operations. Section 4 describes how to combine space-local and state-local ROBs and incorporate hyperreduction in the resulting PMOR method to maximize computational speed. Section 5 demonstrates performance improvements for two representative applications. The first one is characterized by a relatively small-scale, two-dimensional, academic CFD problem designed to highlight the benefits of spatial locality in PMOR: it has the merit of being easily reproducible by the interested reader. The second application focuses on a larger-scale application that is more representative of industrical CFD problems. It highlights the benefits of combining state-local and space-local ROBs, and hyperreduction to squeeze the most performance out of a nonlinear PROM. Section 6 concludes this paper.

2. Nonlinear projection-based model order reduction

The focus of this paper is set on $\boldsymbol{\mu}$-parametric, time-dependent, semi-discrete, nonlinear dynamical systems of the form

$$\begin{aligned} \mathbf{M}(\boldsymbol{\mu})\dot{\mathbf{u}}(t;\boldsymbol{\mu}) + \mathbf{f}\left(\mathbf{u}(t;\boldsymbol{\mu});\boldsymbol{\mu}\right) &= \mathbf{g}(t;\boldsymbol{\mu}) \\ \mathbf{u}(0;\boldsymbol{\mu}) &= \mathbf{u}^0(\boldsymbol{\mu}) \end{aligned} \tag{1}$$

where $t \in [0, T_f]$ denotes time and 0 and T_f specify the simulation time-interval of interest; the dot denotes the derivative with respect to time; $\boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^p$ denotes a p-dimensional vector of parameters and $\mathcal{P}$ is the bounded parameter space of interest (in this work, this space is limited to the time-interval $[0,\, T_f]$ for the reasons explained in Section 1); $\mathbf{u}(t;\boldsymbol{\mu}) \in \mathbb{R}^N$ is the high-dimensional, time-dependent, solution vector and $\mathbf{u}^0(t;\boldsymbol{\mu})$ denotes its initial condition; $\mathbf{M}(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N}$ is the symmetric positive definite (SPD) mass matrix; $\mathbf{f}(\mathbf{u}(t;\boldsymbol{\mu});\boldsymbol{\mu}) \in \mathbb{R}^N$ is a nonlinear function arising from the semi-discretization in space of the governing partial differential equation (PDE); and $\mathbf{g}(t;\boldsymbol{\mu}) \in \mathbb{R}^N$ is a time-dependent source term vector that may be zero in some applications. Model (1) $\big($or problem (1)$\big)$ is referred to as the high-dimensional model (HDM): it may originate from the semi-discretization (spatial discretization) of a PDE with temporal derivatives of any order, because a higher-order ordinary differential equation (ODE) can be rewritten as a system of first-order ODEs by expanding the state vector $\mathbf{u}(t;\boldsymbol{\mu})$. Hence, model (1) is sufficiently general to cover not only first-order systems of conservation laws, but also second-order systems in structural dynamics and wave propagation, as well as many other systems.

For many applications, N is large enough to be prohibitive for computing the solution $\mathbf{u}(t;\boldsymbol{\mu})$ at many parameter points $\boldsymbol{\mu}$ (e.g. design analysis and optimization). Traditional PMOR accelerates the solution of problem (1) by seeking an approximate representation of $\mathbf{u}(t;\boldsymbol{\mu})$ in a low-dimensional affine subspace, which can be written as

$$\mathbf{u}(t;\boldsymbol{\mu}) \approx \tilde{\mathbf{u}}(t;\boldsymbol{\mu}) = \mathbf{u}_0 + \mathbf{V}\mathbf{y}(t;\boldsymbol{\mu}), \quad \forall (t,\boldsymbol{\mu}) \in [0, T_f] \times \mathcal{P} \tag{2}$$

where $\mathbf{u}_0 \in \mathbb{R}^N$ is a fixed vector defining an affine offset for the approximation subspace represented by the *right* ROB $\mathbf{V} \in \mathbb{R}^{N\times n}$, $\mathbf{y} \in \mathbb{R}^n$ is the vector of reduced-order (or generalized) coordinates of the representation of the approximation $\tilde{\mathbf{u}}(t;\boldsymbol{\mu})$ in the subspace, and $n \ll N$. Given an approximation subspace, it is usually desirable to choose a basis that is orthonormal – that is $\mathbf{V}^T\mathbf{Q}\mathbf{V} = \mathbf{I}_n$, where superscript T denotes the transpose operation, $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is a SPD matrix (e.g. $\mathbf{Q} = \mathbf{M}$ or $\mathbf{Q} = \mathbf{I}_n$), and $\mathbf{I}_n$ is the size-n identity matrix. In this paper, all right ROBs $\mathbf{V}$ are constructed to be orthonormal in the sense $\mathbf{V}^T\mathbf{V} = \mathbf{I}_n$.

Inserting the approximation (2) into model (1) and writing the resulting equation in residual form gives

$$\mathbf{r}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t;\boldsymbol{\mu}), \mathbf{V}\dot{\mathbf{y}}(t,\boldsymbol{\mu}), t; \boldsymbol{\mu}\right) = \mathbf{M}(\boldsymbol{\mu})\mathbf{V}\dot{\mathbf{y}}(t;\boldsymbol{\mu}) + \mathbf{f}(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t;\boldsymbol{\mu});\boldsymbol{\mu}) - \mathbf{g}(t;\boldsymbol{\mu}) = 0 \tag{3}$$

where $\mathbf{r}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t;\boldsymbol{\mu}), \mathbf{V}\dot{\mathbf{y}}(t,\boldsymbol{\mu}), t; \boldsymbol{\mu}\right) \in \mathbb{R}^N$ is the value of the high-dimensional residual for a given value of the vector of generalized coordinates $\mathbf{y}(t;\boldsymbol{\mu})$. Equation (3) is overdetermined as it is a system of N equations with $n \ll N$ unknowns: hence, it may not admit an exact solution. It can be solved in a least-squares sense, or transformed into a square system using left and right projections.

2.1. Galerkin and Petrov-Galerkin projections

Discretizing (3) in time and projecting it onto the space spanned by a *left* ROB $\mathbf{W} \in \mathbb{R}^{N\times n}$ gives

$$\mathbf{r}_n(\mathbf{y}(t^m;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}) = \mathbf{W}^T\mathbf{r}(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}) = 0 \tag{4}$$

where $\mathbf{r}_n(\mathbf{y}(t^m;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}) \in \mathbb{R}^n$ is the reduced-order residual and t^m is the time-instance at step m (e.g. in the case of a uniform time-step Δt, $t^m = m\Delta t$). Equation (4) is a square nonlinear system of n equations with n unknowns: it can be solved by Newton's method or a variant. In the absence of some special structure (e.g. a low-order polynomial structure) that allows the precomputation of those contributions to the evaluation of the right-hand side of (4) whose complexity scales with the large dimension N of the HDM, it may be necessary to compute $\mathbf{r}_n \in \mathbb{R}^n$ by evaluating first $\mathbf{r} \in \mathbf{R}^N$, then multiplying it by $\mathbf{W} \in \mathbb{R}^{N\times n}$. In this case, solving (4) will incur computational costs that scale with N unless equation (4) is hyperreduced (see Section 4) – that is, it is further approximated so that the complexity of its solution is independent of N.

The case where $\mathbf{W} = \mathbf{V}$ results in a Galerkin projection. Such a projection is suitable when the semi-discretization of the underlying PDE by a Galerkin projection is also suitable. Otherwise, a Petrov-Galerkin projection, where $\mathbf{W} \neq \mathbf{V}$, is more appropriate [27]: for example, this is the case for convection-dominated flow problems [8] for which the nonlinear LSPG method [4, 5] has been shown to be particularly robust and efficient [8, 9]. For this reason, and because LSPG is sufficiently general (it incorporates Galerkin projection as a particular case), the contributions of this paper are presented in the setting of this nonlinear PMOR method. However, it is emphasized again that they are equally applicable in many other PMOR settings.

The nonlinear LSPG method can be simply explained by examining the iterates arising from the application of the Newton algorithm to the solution of (4). Each k-th iterate can be written as

$$\mathbf{W}^{(k)T}\mathbf{J}^{(k)}\mathbf{V}\mathbf{p}^{(k)} = -\mathbf{W}^{(k)T}\mathbf{r}^{(k)} \tag{5}$$

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} + \beta^{(k)}\mathbf{p}^{(k)} \tag{6}$$

where all arguments of all quantities have been dropped for notational simplicity; the superscript (k) designates the k-th iteration; the left ROB $\mathbf{W}^{(k)}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}^{(k)}(t^m;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}\right) \in \mathbb{R}^{N\times n}$ is allowed to be a function of the solution iterate; $\mathbf{J}^{(k)}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}^{(k)}(t^m;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}\right) \in \mathbb{R}^{N\times N}$ is the Jacobian of the high-dimensional residual; $\mathbf{p}^{(k)} \in \mathbb{R}^n$ is the Newton step direction; $\mathbf{r}^{(k)}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}^{(k)}(t^m;\boldsymbol{\mu}), t^m; \boldsymbol{\mu}\right) \in \mathbb{R}^N$ is the residual at iteration k; and $\beta^{(k)}$ is the step length at iteration k and can be determined using a line search or set by default to 1.

If by design, the left ROB is chosen to be iteration-dependent and constructed as $\mathbf{W}^{(k)} = \mathbf{J}^{(k)}\mathbf{V}$, it can be shown that:

- The Newton iterates (5)-(6) become those of the Gauss-Newton method applied to the minimization of the Eucledian norm of the high-dimensional residual (3) over $\mathbf{y} \in \mathbb{R}^n$.
- The left ROB $\mathbf{W}^{(k)}$ minimizes the difference in the $\mathbf{J}^{(k)T}\mathbf{J}^{(k)}$ norm between each low-order approximate Newton step direction $\mathbf{V}\mathbf{p}^{(k)}$ and its high-dimensional counterpart $\mathbf{J}^{(k)^{-1}}\mathbf{r}^{(k)}$.

In this case, equation (5) becomes the normal equation for a linear least-squares problem, the Newton iterations associated with LSPG can be rewritten as

$$\begin{aligned} \mathbf{p}^{(k)} &= \arg\min_{\mathbf{a}\in\mathbb{R}^n} \left\| \mathbf{J}^{(k)}\mathbf{V}\mathbf{a} + \mathbf{r}^{(k)} \right\|_2 \\ \mathbf{y}^{(k+1)} &= \mathbf{y}^{(k)} + \beta^{(k)}\mathbf{p}^{(k)} \end{aligned} \tag{7}$$

and the main computational cost of each Gauss-Newton iteration originates from the following two operations:

- The matrix-matrix product $\mathbf{J}^{(k)}\mathbf{V}$. In many computational physics applications, the Jacobian matrix $\mathbf{J}^{(k)}$ is sparse with l nonzero entries per row, the product $\mathbf{J}^{(k)}\mathbf{V}$ is (or should be) computed using a sparse matrix-matrix multiplication, and therefore its computational complexity is $\mathcal{O}(Nnl)$.
- The solution of a linear least-squares problem governed by the matrix $\mathbf{J}^{(k)}\mathbf{V}$ and characterized by the right-hand side $-\mathbf{r}^{(k)}$. The complexity of the solution of this problem by a preferred technique such as the normal equation, SVD, or the QR decomposition, has a complexity $\mathcal{O}(Nn^2)$.

In practice, n is sufficiently small so that the computation of the product $\mathbf{J}^{(k)}\mathbf{V}$ dominates the main computational cost of each Gauss-Newton iteration. Therefore, any techniques that can significantly reduce the cost of the matrix-matrix product $\mathbf{J}^{(k)}\mathbf{V}$ will accelerate the online execution of the nonlinear PROM. It follows that as already stated in Section 1, any PMOR techniques that can deliver the desired level of accuracy using a smaller dimension of the PROM can achieve this objective, by lowering n and thus the complexity $\mathcal{O}(Nnl)$. This paper develops an alternative as well as complementary approach for achieving the same objective based on sparsifying the right ROB $\mathbf{V}$ and therefore reducing the constant of the complexity $\mathcal{O}(Nnl)$ of the matrix-matrix product $\mathbf{J}^{(k)}\mathbf{V}$ (see Section 3) – which may accelerate faster the online processing of the nonlinear PROM.

2.2. Proper orthogonal decomposition, SVD, and properties

There is a variety of established approaches for constructing a right-ROB $\mathbf{V} \in \mathbf{R}^{N\times n}$. For linear problems, a robust theory exists for this purpose and is supported by a variety of stability and accuracy guarantees [28, 29]. For nonlinear problems, the most popular PMOR methods rely on the collection of a series of solution snapshots (or related quantities) $\mathcal{S} = \{\mathbf{u}_i\}_{i=1}^{N_s}$, where $\mathbf{u}_i = \mathbf{u}(t_i;\boldsymbol{\mu}_i) - \mathbf{u}_0$ is the solution at a particular parameter point (including time in the definition of the parameter space) of the problem arising from the discretization of the HDM (1), adjusted to account for the affine offset; and N_s denotes the number of collected solution snapshots.

Let $\widetilde{\mathbf{S}} = [\mathbf{u}_1 \mid \cdots \mid \mathbf{u}_{N_s}] \in \mathbb{R}^{N\times N_s}$ denote the snapshot matrix and let $\mathbf{S}$ denote its *scaled* counterpart – that is

$$\mathbf{S} = \widetilde{\mathbf{S}}\mathbf{D}, \quad \text{where} \quad \mathbf{D} = \operatorname{diag}\left\{ \frac{1}{\|\mathbf{u}_1\|_2}, \frac{1}{\|\mathbf{u}_2\|_2}, \cdots, \frac{1}{\|\mathbf{u}_{N_s}\|_2} \right\}$$

It follows that each scaled solution snapshot stored in $\mathbf{S}$ has a unit two-norm. Then, a ROB can be constructed by collating, selecting, or compressing the columns of $\mathbf{S}$ – that is, the scaled solution snapshots. POD by the method of solution snapshots [30, 31] constructs a right ROB of dimension n by minimizing in a least-squares sense the projection error of the solution snapshots – that is, as follows

$$\mathbf{V} = \arg\min_{\mathbf{U}\in\mathscr{S}_{n,N}} \sum_{i=1}^{N_s} \|\mathbf{u}_i - \Pi_{\mathbf{U}}\mathbf{u}_i\|_2^2 \tag{8}$$

where $\mathscr{S}_{n,N}$ denotes the Stiefel manifold of all orthogonal matrices in $\mathbb{R}^{N\times n}$; and $\Pi_{\mathbf{U}}$ denotes the operator that performs an orthogonal projection onto the subspace spanned by $\mathbf{U}$, which, for an orthogonal matrix $\mathbf{U}$, is given by $\Pi_{\mathbf{U}} = \mathbf{U}\mathbf{U}^T$. Hence, the aforementioned scaling of the solution snapshots is performed to prevent biasing otherwise the solution of the above optimization problem towards the snapshots with larger magnitudes.

The minimization problem implied in (8) has an analytical solution given by the SVD, as described in Algorithm 1.

Algorithm 1 Computation of a POD basis using SVD.

Input: scaled snapshot matrix $\mathbf{S} \in \mathbb{R}^{N \times N_s}$; basis dimension n or selection criterion for n
Output: POD basis $\mathbf{V} \in \mathbb{R}^{N \times n}$
1: Compute the thin SVD of the snapshot matrix: $\mathbf{S} = \mathbf{X}\mathbf{\Sigma}\mathbf{Y}^T$
2: Set or determine the basis dimension n, which may be a function of the singular values stored in $\mathbf{\Sigma}$
3: Return the truncated matrix of left singular vectors: $\mathbf{V} = [\mathbf{x}_1 \mid \cdots \mid \mathbf{x}_n]$

POD provides the n-dimensional basis $\mathbf{V} \in \mathbb{R}^{N \times n}$ that is optimal in the sense that it achieves the smallest least-squares error when approximating the solution snapshots using (2). Furthermore, the truncated singular values of the matrix $\mathbf{S}$ provides the following simple and efficient formula for computing the projection error e_n of the affine approximation of the scaled solution snapshots

$$e_n = \sum_{i=1}^{N_s} \|\mathbf{u}_i - \Pi_{\mathbf{V}} \mathbf{u}_i\|_2^2 = \sum_{k=n+1}^{r} \sigma_k^2 \tag{9}$$

where $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_{N_s})$ denotes the matrix of singular values ordered as $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > (\sigma_{r+1} = 0) \geq \cdots \geq \sigma_{N_s} = 0$, r denotes the rank of the matrix $\mathbf{S}$ – and therefore r is the index of the smallest nonzero σ value – and $n \leq r$. Collectively, the solution of problem (8) given by Algorithm 1 and the formula (9) are known as the Eckart-Young-Mirsky [32] theorem. The error (9) can be normalized to obtain its counterpart $\epsilon_n \in]0, 1[$

$$\epsilon_n = \frac{\sum_{k=n+1}^{r} \sigma_k^2}{\sum_{k=1}^{r} \sigma_k^2} = \frac{\sum_{i=1}^{N_s} \|\mathbf{u}_i - \Pi_{\mathbf{V}} \mathbf{u}_i\|_2^2}{\sum_{i=1}^{N_s} \|\mathbf{u}_i\|_2^2}$$

where $\{\mathbf{u}_i\}_{i=1}^{N_s}$ denotes in this case the set of scaled solution snapshots. A popular and reasonable truncation criterion for determining an appropriate dimension n of the right ROB is then given by the relative value of the total singular value energy accounted for

$$\mathcal{E}_n = 1 - \epsilon_n = \frac{\sum_{k=1}^{n} \sigma_k^2}{\sum_{k=1}^{r} \sigma_k^2}$$

Hence, rather than specifying a ROB dimension n, which may be hard to choose *a priori*, it is common to specify a desired singular value energy threshold $0 < \tau < 1$ and choose the smallest n that meets the specified threshold – that is,

$$\frac{\sum_{k=1}^{n} \sigma_k^2}{\sum_{k=1}^{r} \sigma_k^2} \geq \tau \tag{10}$$

This can be simply done in the second line of Algorithm 1, once the SVD has been computed.

3. Space-local reduced-order bases

3.1. The concept

The POD method described above pre-supposes that a *small* set of vectors that closely approximates the solution snapshots is sought-after. Underlying this choice is the assumption that the cost of exploiting the resulting PROM scales with the number of constructed basis vectors. For this reason, the objective is set to constructing a right ROB that is as accurate as possible for a given dimension n. However, the primary operations performed using a right ROB during a PROM-based computation are matrix-matrix

and matrix-vector multiplications. In general, such computations scale with the number of *nonzero entries* of the ROB, when it is stored in a sparse data structure (e.g. the popular CSR or CSC sparse matrix formats). On the other hand, if a sparse ROB is constructed, its dimension n may not be as critical for computational complexity as the number of nonzero entries it contains. This observation motivates an alternative framework for constructing a right ROB that introduces sparsity into the basis.

Many physical simulations are characterized by regions of the computational domain where the solution exhibits different physical behaviors. For example, in fluid flow problems, a solution computed during a single simulation may exhibit a shock in one region, boundary layers near wall boundaries, and/or a turbulent wake behind some of them. Most importantly, these flow features may *evolve* with the parameters of the parameter space $\mathcal{P}$ (which in this work includes time) – that is, with $\boldsymbol{\mu} \in \mathcal{P}$. In each of these regions, the fluid is in a different flow regime and therefore the solution may evolve in different subspaces. In such a scenario, it may be beneficial to construct a space-local right ROB rather than a space-global one to efficiently capture all of the disparate physics simultaneously. One may also expect that by tailoring a space-local right ROB to the feature richness/poorness of the solution in each nonoverlapping subdomain, and more importantly, its evolution with $\boldsymbol{\mu} \in \mathcal{P}$, one may be able to reach a desired level of accuracy using fewer basis vectors than in the case of a space-global ROB. Furthermore, each component of a space-local basis is by definition confined to a region of the computational domain and therefore locally supported. When all locally supported components of a right ROB are concatenated to construct a right ROB for the entire computational domain, such a right basis will be sparse, and this sparsity can be exploited to enhance the performance of the resulting PROM.

Having motivated it above in broad strokes, the concept of a space-local right ROB is next explained in detail.

A schematic of the proposed approach is shown in Figure 1. First, a snapshot matrix is collected by sampling the high-dimensional model in the parameter space of interest (here, time), as in the traditional POD method. Then, the computational domain Ω – or more specifically, the computational mesh – is partitioned into N_c nonoverlapping subdomains Ω_i (submeshes, or clusters of mesh nodes or elements), each to be covered by its own specialized ROB. The partitioning may be done manually *a priori*, or by clustering the nodes of the mesh using the solution snapshots as features and a clustering algorithm (see Section 3.3).

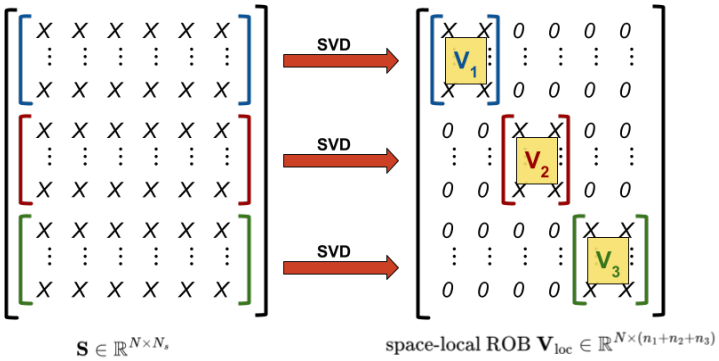

Figure 1: A schematic overview of the proposed approach for constructing a space-local right ROB using a snapshot matrix.

After the mesh is partitioned, the unknowns – or degrees of freedom (dofs) – attached to the mesh nodes of a nonoverlapping subdomain Ω_i can be renumbered consecutively, using any optimal numbering scheme. Then, the snapshot matrix $\mathbf{S}$ can be partitioned into row-wise blocks (or submatrices) according to the nonoverlapping mesh partition, as follows

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 \\ \vdots \\ \mathbf{S}_{N_c} \end{bmatrix}$$

where $\mathbf{S} \in \mathbb{R}^{N \times N_s}$, $\mathbf{S}_i \in \mathbb{R}^{N_i \times N_s}$, for $i = 1, \ldots, N_c$, and N_i denotes the number of dofs contained in Ω_i. Then, SVD is applied separately to compress each snapshot submatrix into an independent right ROB

$\mathbf{V}_i \in \mathbb{R}^{N_i \times n_i}$, where the dimension n_i depends on the behavior of the solution in the nonoverlapping subdomain Ω_i. Strategies for determining the dimensions n_i are discussed in Section 3.2.

Next, the independent right ROBs $\mathbf{V}_i$, $i = 1, \cdots, N_c$, are concatenated to form what has been referred to above as a space-local right ROB

$$\mathbf{V}_{\text{loc}} = \begin{bmatrix} \mathbf{V}_1 & & \\ & \ddots & \\ & & \mathbf{V}_{N_c} \end{bmatrix} \tag{11}$$

where $\mathbf{V}_{\text{loc}} \in \mathbb{R}^{N \times (n_1 + \cdots + n_{N_c})}$ and each submatrix $\mathbf{V}_i$ is a locally supported basis and defines what has been referred to so far as a component of the space-local right ROB $\mathbf{V}_{\text{loc}}$. Note that

$$\mathbf{V}_{\text{loc}}^T \mathbf{V}_{\text{loc}} = \begin{bmatrix} \mathbf{V}_1^T \mathbf{V}_1 & & \\ & \ddots & \\ & & \mathbf{V}_{N_c}^T \mathbf{V}_{N_c} \end{bmatrix}$$

Hence, if each component $\mathbf{V}_i$ is orthonormal, $\mathbf{V}_{\text{loc}}$ is orthonormal.

One might hope that by tailoring each approximation subspace represented by one component $\mathbf{V}_i$ of the space-local right ROB (11) to the spatial features exhibited by the solution in the corresponding nonoverlapping subdomain Ω_i, one could reduce the dimension n_i of $\mathbf{V}_i$ to the extent that for a desired level of accuracy, the total dimension $n_{\text{loc}} = \sum_{i=1}^{i=N_c} n_i$ of $\mathbf{V}_{\text{loc}}$ could be smaller than that of a counterpart right ROB constructed using the traditional POD method. Indeed, if for a fixed accuracy $n_{\text{loc}} < n$ were possible, performance acceleration of the resulting PROM could be expected even in the absence of sparse data structures for storing and manipulating $\mathbf{V}_{\text{loc}}$. Unfortunately, this is impossible – at least in the following sense, and for the following reason. From the Eckart-Young-Mirsky theorem [32], it follows that for a given dimension n, the right ROB constructed using POD by the method of solution snapshots reconstructs the snapshot matrix $\mathbf{S}$ with the best possible accuracy. In other words, for a given dimension n, a traditional, POD-based, dense right ROB will always outperform any other right ROB, including $\mathbf{V}_{\text{loc}}$, from a projection error viewpoint. Consequently, in the context of a space-local right ROB (11) of dimension n_{loc}, the objective should be to find the mesh partition for which:

- $\mathbf{V}_{\text{loc}}$ delivers the same level of accuracy as a traditional, POD-based, dense counterpart right ROB of dimension n, using a dimension n_{loc} that is only marginally greater than n.
- The sparsity pattern of $\mathbf{V}_{\text{loc}}$ and its dimension n_{loc} are such that the space-local PROM based on $\mathbf{V}_{\text{loc}}$ is faster than its counterpart based on a traditional, POD-based, dense right ROB $\mathbf{V}$.

Specifically, exploiting the sparsity pattern of $\mathbf{V}_{\text{loc}}$ (11) during the online solution of problem (7), whose complexity dominates the overall complexity of LSPG during its online stage, can accelerate the performance of this nonlinear PMOR method as follows:

- The first step in solving problem (7) consists in computing the matrix-matrix product $\mathbf{J}^{(k)}\mathbf{V}$. For all applications where the semi-discrete HDM (1) arises from a finite element (FE), finite volume (FV), or finite difference spatial approximation of the underlying PDE, the Jacobian matrix $\mathbf{J}^{(k)}$ is typically sparse. In this case, the product $\mathbf{J}^{(k)}\mathbf{V}_{\text{loc}}$ can be performed using sparse-sparse matrix-matrix multiplication. For a traditional, POD-based, dense right ROB $\mathbf{V}$ and a sparse Jacobian with l nonzero entries per row, the complexity of computing $\mathbf{J}^{(k)}\mathbf{V}$ is $O(Nln)$. In the case of the sparse ROB $\mathbf{V}_{\text{loc}}$ however, if $0 \leq \gamma \leq 1$ denotes the fraction of $\mathbf{V}_{\text{loc}}$'s entries that are nonzero, skipping the zero entries of $\mathbf{V}_{\text{loc}}$ during the evaluation of the product $\mathbf{J}^{(k)}\mathbf{V}_{\text{loc}}$ reduces its complexity to less than $O(Nln\gamma)$. This upper bound corresponds to a worst-case algorithm that expends time on operations involving two entries J_{ij} and V_{kl} if *either* of them is nonzero. Therefore, in the presence of a sparse right ROB, minimizing the quantity $n\gamma$ is more important than minimizing n, as far as computational performance is concerned. Consequently, if a space-local right ROB can be constructed such that it delivers a similar accuracy to that of a traditional,

POD-based, dense counterpart using a marginally larger dimension but a substantially smaller γ factor, such a right ROB can lead to substantial computational savings in the evaluation of the product $\mathbf{J}^{(k)}\mathbf{V}$; thus, it can accelerate the wall-clock time performance of LSPG.

- The second step in solving problem (7) consists in solving a linear least-squares problem using the outcome of the matrix-matrix product $\mathbf{J}^{(k)}\mathbf{V}$. When $\mathbf{V}$ is sparse, this outcome is a sparse matrix whose sparsity pattern can be exploited during the online execution of LSPG. For example, sparse iterative least-squares solvers are available for the solution of problem (7) (e.g. see [33]). If the normal equation approach is chosen for solving this problem and $\mathbf{V} = \mathbf{V}_{\text{loc}}$, the product $\left(\mathbf{J}^{(k)}\mathbf{V}_{\text{loc}}\right)^T \left(\mathbf{J}^{(k)}\mathbf{V}_{\text{loc}}\right)$ can be evaluated using sparse-sparse matrix-matrix multiplication. In practice however, the cost of the first step outlined above tends to dominate the total computational cost of solving problem (7). Therefore, treating the outcome of the product $\mathbf{J}^{(k)}\mathbf{V}_{\text{loc}}$ as a sparse matrix rather than an ordinary dense matrix is likely unnecessary for optimizing the performance of the nonlinear LSPG PROM.

While the benefits of sparsity can outweigh the cost of a few additional right basis vectors, there is a limit to this reasoning. Eventually, the cost of the solution of the least-squares problem in the second step discussed above, which scales as $O(n^2)$, will not only become less negligible compared to that of the first step, but will dominate the total cost of solving the optimization problem (7). For example, $\mathbf{V} = \mathbf{I}_N$ is the sparsest possible right ROB, with only one nonzero entry per dof. However, this choice leads to the HDM and therefore is inefficient. For this reason, a relatively small number of spatial clusters is recommended as in this case, a space-local right ROB does not need many more additional basis vectors than a dense POD right ROB to deliver a comparable level of accuracy.

3.2. Singular values and projection error

A beneficial element of the POD method for the construction of a traditional PROM is the singular value energy criterion for selecting an appropriate dimension of the right ROB. For a space-local right ROB however, the basis truncation process is more complex because there are $N_c > 1$ appropriate dimensions to be determined – one for each component $\mathbf{V}_i$ of $\mathbf{V}_{\text{loc}}$ (11). Hence, it is desirable to develop for a space-local right ROB a connection between singular values and the projection error to aid with the selection of the dimensions n_i of the basis components $\mathbf{V}_i$, $i = 1, \ldots, N_c$.

To this end, consider a space-local right ROB with N_c components (11) rewritten in compact form as $\mathbf{V}_{\text{loc}} = \text{diag}\left(\mathbf{V}_1, \cdots, \mathbf{V}_{N_c}\right)$, where $\mathbf{V}_i \in \mathbb{R}^{N_i \times n_i}$, $i = 1, \ldots, N_c$. The *total* projection error of the set of N_s solution snapshots stored in $\mathbf{S} = [\mathbf{u}_1 \mid \cdots \mid \mathbf{u}_{N_s}]$ is

$$e_{n,\text{loc}} = \sum_{i=1}^{N_s} \|\mathbf{u}_i - \Pi_{\mathbf{V}_{\text{loc}}}\mathbf{u}_i\|_2^2 = \left\|\left(\mathbf{I}_N - \mathbf{V}_{\text{loc}}\mathbf{V}_{\text{loc}}^T\right)\mathbf{S}\right\|_F = \left\|\left(\mathbf{I}_N - \text{diag}\left(\mathbf{V}_1\mathbf{V}_1^T, \cdots, \mathbf{V}_{N_c}\mathbf{V}_{N_c}^T\right)\right)\mathbf{S}\right\|_F^2$$

where the subscript F denotes the Frobenius norm. Due to the block-diagonal structure of the matrix $\mathbf{V}_{\text{loc}}$ and to the result (9), the above error can be rewritten as

$$e_{n,\text{loc}} = \sum_{i=1}^{N_c} \left\|\left(\mathbf{I}_{N_i} - \mathbf{V}_i\mathbf{V}_i^T\right)\mathbf{S}_i\right\|_F^2 = \sum_{i=1}^{N_c} \sum_{j=n_i+1}^{r_i} \left(\sigma_{i,j}\right)^2 \tag{12}$$

where $\sigma_{i,j}$ denotes the j-th singular value returned by the POD performed in the nonoverlapping subdomain Ω_i and r_i denotes the rank of the basis component $\mathbf{V}_i$.

The error result (12) suggests the following potential truncation strategies:

- S1: Construct the n_{loc}-dimensional space-local right ROB that achieves the smallest snapshot projection error by choosing the left singular vectors associated with the n_{loc} largest singular values, irrespective of which nonoverlapping subdomain Ω_i they belong to. This strategy automatically determines the relative dimensions of the various components $\mathbf{V}_i$ of $\mathbf{V}_{\text{loc}}$.

- S2: Alternatively, construct a space-local right ROB with N_{nnz} nonzero entries that achieves the smallest projection error, by progessively choosing the singular vectors associated with the largest scaled singular values $\sigma_{i,j}/N_i$, $i = 1, \ldots, N_c$, $j = 1, \ldots, n_{\text{loc}}$, where in this case n_{loc} is the smallest integer for which the resulting space-local right ROB $\mathbf{V}_{\text{loc}}$ has N_{nnz} nonzero entries.

- S3: Apply a separate energy truncation criterion in each nonoverlapping subdomain Ω_i. For example, if the solution or its behavior in Ω_i is known to be more or less influencial than elsewhere on the quantities of interest (QoIs), retain more or fewer basis vectors in Ω_i.

3.3. Feature-based nonoverlapping domain decomposition and snapshot clustering

It remains to describe how to perform nonoverlapping mesh partitioning (and therefore nonoverlapping domain decomposition), to support the construction of a space-local right ROB $\mathbf{V}_{\text{loc}}$ (11). Because there is no unique solution to this problem, two sample partitioning approaches are presented below. The first one is a heuristic for *manually* decomposing the computational domain. The second approach is an *automated* approach for discovering *parameter-dependent* spatial features of the solution – that is, spatial features that evolve when $\boldsymbol{\mu}$ is varied in the parameter space $\mathcal{P}$ (which in this work includes time) and therefore cannot be captured by a small number of solution snapshots. For some applications, such features may be obvious to the eye of a trained analyst, may suggest a simple manual partitioning, or may have properties that allow a convervative identification of their locations via a simple numerical algorithm.

3.3.1. Heuristic for manual nonoverlapping mesh partitioning

The main idea here is that in many computational physics applications, the analyst can distinguish a priori between a region of the computational domain (which does not necessarily have to be singly-connected), where the solution of the problem of interest may exhibit parameter-dependent spatial features; and the remainder of the computational domain, where the solution is either feature-poor, or its features are relatively parameter-independent. For example, in CFD and solid mechanics applications, the analyst can often anticipate the regions of the computational domain where eddies and/or shocks in the first case, or large displacements/rotations and/or stress concentration in the second case, may occur and evolve in time or with the design parameters. Hence, a simple heuristic for manually decomposing a computational domain Ω into two nonoverlapping subdomains via nonoverlapping mesh partitioning is to distinguish between a region Ω_1, where it can be anticipated that the solution will develop parameter-dependent features; and the remainder of the computational domain $\Omega_2 = \Omega \backslash \Omega_1$. This heuristic, which is valid for both steady (static) and unsteady (dynamic) problems, is discussed below.

Borrowing terminology from the field of CFD for external flows, the region of the computational domain Ω where the solution is feature-rich and the features are known to be parameter-dependent, is referred to here as the "near-field" (NF) and denoted by Ω_{NF}; and the region of Ω where the solution is feature-poor or the features are known to be parameter-independent is referred to as the "far-field" (FF) and denoted by Ω_{FF}. Again, neither Ω_{NF} nor Ω_{FF} needs to be singly-connected. Then, the proposed heuristic for manual nonoverlapping mesh partitioning, which is not limited to CFD and solid mechanics applications, can be described as follows: specify, within a reasonable level of conservatism, the interface boundaries delimiting the near-field region $\Omega_{\text{NF}} \subset \Omega$; treat the remainder of the computational domain as the far-field region – that is, $\Omega_{\text{FF}} = \Omega \backslash \Omega_{\text{NF}}$; and perform the nonoverlapping mesh partitioning accordingly. The expectation is that the right ROB component associated with the near-field will require a larger dimension n_{NF} than its counterpart n_{FF} associated with the far-field – that is, $n_{\text{NF}} > n_{\text{FF}}$. However, the spatial support of this ROB component in terms of dofs may be smaller than that of its counterpart, which is the scenario where the heuristic for manual nonoverlapping mesh partitioning is most appropriate.

Let N_{NF} and N_{FF} denote the number of dofs attached to the nonoverlapping subdomains Ω_{NF} and Ω_{FF}, respectively, with $N_{\text{NF}} < N_{\text{FF}}$, and let $N = N_{\text{NF}} + N_{\text{FF}}$. Combining both ROB components associated with Ω_{NF} and Ω_{FF} as in (11) leads to a space-local right ROB with $N_{nnz} = N_{\text{NF}} n_{\text{NF}} + N_{\text{FF}} n_{\text{FF}}$ nonzero entries. Hence, for $\mathbf{V}_{\text{loc}}$, the average number of nonzero entries per dof is

$$\frac{N_{nnz}}{N} = \frac{N_{\text{NF}}}{N} n_{\text{NF}} + \frac{N_{\text{FF}}}{N} n_{\text{FF}}$$

Then, a set of conditions that promtotes computational efficiency in a space-local right ROB, relative to a traditional POD-based dense counterpart is:

- $n_{\mathrm{NF}} \leq n$, where n denotes the dimension of a traditional POD-based dense right ROB for the entire computational domain Ω. This is a necessary condition for building a computationally efficient space-local right ROB of the form given in (11) for a trivial reason. For many applications, it is rather easy to satisfy.
- $n_{\mathrm{FF}} < n_{\mathrm{NF}}$, which is easily satisfied when Ω_{NF} is properly defined.
- N_{NF}/N is small, which requires that the solution in the chosen Ω_{NF} be not only feature-rich, but that its features in this subdomain evolve with $\boldsymbol{\mu} \in \mathcal{P}$.

To illustrate this simple heuristic, consider the following one-dimensional initial boundary value problem (IBVP) based on the inviscid Burgers equation

$$\begin{aligned} \frac{\partial}{\partial t} w(x,t) + \frac{\partial}{\partial x}\left(\frac{1}{2}(w(x,t))^2\right) &= 0 \\ w(x,0) &= \begin{cases} 4 + 0.005x & 0 \leq x \leq 50 \\ 1 & 50 < x \leq 100 \end{cases} \\ w(0,t) &= 4 \end{aligned} \tag{13}$$

where $w(x,t) \in \mathbb{R}$ denotes the scalar state variable, $x \in [0,100]$, and $t \in [0,20]$. The IBVP (13) is discretized in space using a first-order Godunov method and 500 evenly spaced mesh nodes; and in time using the second-order trapezoidal rule and the time-step $\Delta t = 0.05$. Its numerical solution obtained using the outlined discrete HDM is represented by the dashed lines in Figure 2. It shows that for the specified initial condition, the initial shock advects rightward across the spatial domain. In the left half of this domain, the solution is nearly constant in time: in the context of the affine approximation (2), it can be captured by a low-dimensional right ROB. In the right half however, the solution exhibits an advecting shock: in the same context, its approximation requires a much higher-dimensional right ROB. To deliver the high degree of accuracy shown by the red lines in Figure 2, an LSPG PROM constructed using a traditional POD-based dense right ROB requires a dimension $n \geq 111$ ($\mathcal{E}_n = 99.9999\%$). On the other hand, the counterpart space-local LSPG PROM constructed using the partitioning of the domain $x \in [0,100]$ into $\Omega_{\mathrm{NF}} = \{50 < x \leq 100\}$ and $\Omega_{\mathrm{FF}} = \{0 \leq x \leq 50\}$, and using the truncation strategy S1 (see Section 3.2) achieves the same total singular value energy ($\mathcal{E}_{n_{\mathrm{loc}}} = \mathcal{E}_n$) with the same dimension $n_{\mathrm{loc}} = n = 111$ distributed as follows: $n_{\mathrm{NF}} = 110$; and $n_{\mathrm{FF}} = 1$. Then, as expected and shown by the blue lines in Figure 2, the space-local LSPG PROM of the same dimension as the traditional LSPG PROM delivers the same accuracy, but benefits computationally from the sparsity of its ROB $\mathbf{V}_{n_{\mathrm{loc}}}$.

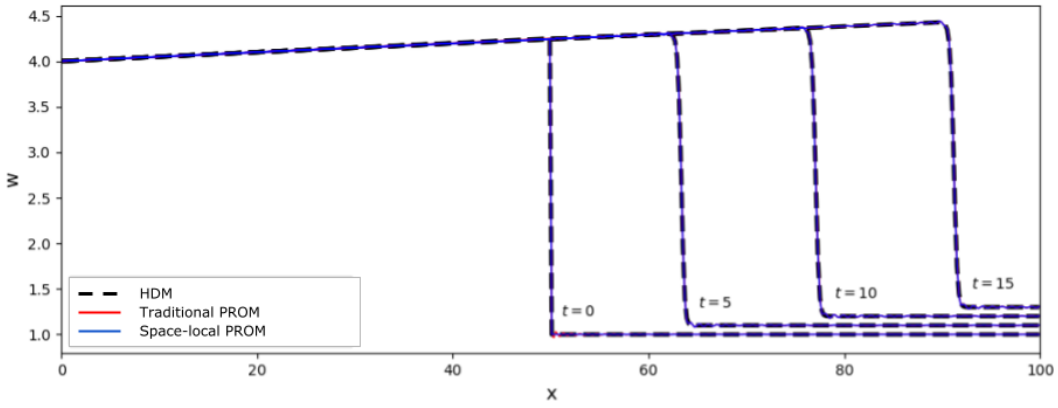

Figure 2: IBVP based on the inviscid Burgers equation – Numerical predictions obtained using: the HDM; an LSPG PROM of dimension $n = 111$ constructed using a traditional, POD-based, dense right ROB and $\mathcal{E}_n = 99.9999\%$; and a space-local LSPG PROM of dimension $n_{\mathrm{loc}} = 111$ constructed using a space-local right ROB and $\mathcal{E}_n = 99.9999\%$.

In summary, for this example:

- For a similar high level of accuracy, the traditional POD-based dense right ROB $\mathbf{V}$ is characterized by the dimension $n = 111$; and the space-local right ROB is characterized by $N_{\text{NF}} = N_{\text{FF}} = 250$, $n_{\text{NF}} = 110$, $n_{\text{FF}} = 1$, and the *same dimension* $n_{\text{loc}} = n = 111$.
- For the traditional, POD-based, dense right ROB, the average number of nonzero entries per dof is $N_{nnz}/N = (N \times n)/N = n = 111$; for the space-local right ROB $\mathbf{V}_{\text{loc}}$ however, this number is $N_{nnz}/N = 55.5$ – that is, twice smaller.
- From the complexity analysis performed in Section 3.1, it follows that the LSPG PROM built using the space-local right ROB should be more computationally efficient than its counterpart constructed using the traditional, POD-based, dense right ROB.

The heuristic for manual nonoverlapping mesh partitioning presented here is demonstrated for a large-scale realistic turbulent flow problem in Section 5, where the wall-clock time performance of the LSPG PROM constructed using the resulting space-local right ROB $\mathbf{V}_{\text{loc}}$ is shown to be superior to that of its counterpart built using the traditional POD-based dense right ROB.

3.3.2. FFT-k-means for automatic spatial feature identification and nonoverlapping mesh partitioning

An alternative approach is presented here for decomposing the computational domain Ω via nonoverlapping mesh partitioning, in order to cover the case where the manual partitioning approach described above is not easily applicable. It is designed specifically for time-dependent problems. In this approach, the nonoverlapping domain decomposition task is formulated as a clustering problem: for example, a node-clustering problem; or a dof-clustering problem. In the former case, the time-history of any scalar component of the solution (e.g. pressure, velocity, density, etc.) can be inputted as the vector-valued feature to be clustered. In the latter case, all rows of the snapshot matrix $\mathbf{S}$, which are associated with the dofs of the problem of interest, are considered to be the vector-valued features to be clustered. In both cases, the clustering problem is solved using Algorithm 2, which builds on the k-means algorithm. Specifically, Algorithm 2 is described in the case of a one-dimensional parameter space $\mathcal{P}$ containing only the time variable, as it assumes that two consecutive columns of the snapshot matrix $\mathbf{S}$ correspond to two consecutive time-instances. Furthermore, it is written in its simplest and most elegant form in order to focus on its essence. In this form, it is applicable to the node-clustering of a scalar problem; and to the dof-clustering of scalar as well as vector problems. However, the generalizations of this algorithm to a higher-dimensional parameter space and to node-clustering of vector problems are straightforward and require only minor modifications. Numerous experimentations performed by the authors with this algorithm and unsteady flow problems have shown that it is capable of identifying many flow features, when the solution snapshots are collected at evenly spaced time-intervals (e.g. see Section 5).

Considering that the dynamics of distinct spatial features associated with different physics regimes are likely to evolve at different time-scales and tend to be characterized by different frequency-domain characteristics, Algorithm 2 begins by applying the fast Fourier transform (FFT) to each row of the snapshot matrix $\mathbf{S}$ – that is, to the time-history of the state variable with which a row is associated. Then, it clusters the computed frequency-magnitude spectra using the standard k-means algorithm: this unsupervised machine learning algorithm is described in Algorithm 3, to keep this paper as self-contained as possible.

Algorithm 2 Decomposition of the computational domain using FFT-k-means.

Input: snapshot matrix $\mathbf{S} \in \mathbb{R}^{N \times N_s}$; number of spatial clusters N_c
Output: assignment $a_i \in \{1, \ldots, N_c\}$ of each row of $\mathbf{S}$ to a spatial cluster, $i = 1, \ldots, N$

1: Let $\mathbf{S}[i, \cdot]$ denote the i-th row of $\mathbf{S}$: compute the FFT magnitude spectra $\mathbf{S}[i, \cdot]_{\text{FFT-mag}}$, $i = 1, \ldots, N$
2: $a_i \leftarrow$ `kmeans` $(\mathbf{S}[i, \cdot]_{\text{FFT-mag}}, \ldots, \mathbf{S}[N_{\text{FFT-mag}}, \cdot])$

For many unsteady flow problems, including the first application discussed in Section 5, the FFT-k-means approach outlined above demonstrated the ability to identify the regions of a computational domain where the solution exhibits distinct localized behaviors. However, once the nonoverlapping subdomains containing the localized features have been identified, they should be further expanded to

Algorithm 3 Clustering algorithm `kmeans`.

Input: N vectors to cluster $\mathbf{x}_1, \ldots, \mathbf{x}_N$; number of clusters N_c
Output: assignment $a_i \in \{1, \ldots, N_c\}$ of each vector $\mathbf{x}_i$ to a cluster, $i = 1, \ldots, N$
1: Choose N_c random vectors as initial centroids $\mathbf{c}_i$, $i = 1, \ldots, N_c$
2: **while** at least one assignment a_i has changed between the last two iterations **do**
3: $\quad a_i \leftarrow \arg\min_j \|\mathbf{x}_i - \mathbf{c}_j\|_2$, $i = 1, \ldots, N$
4: $\quad \mathbf{c}_j \leftarrow \left(\sum_{a_i = j} \mathbf{x}_i \right) \Big/ \text{count}_{i=1,\ldots,N}(a_i = j)$
5: **end while**

introduce conservatism in the overall approach and avoid, for example, feature clipping at the subdomain edges. Algorithm 4 accomplishes this task by essentially adding a few layers of neighboring mesh nodes to each nonoverlapping subdomain identified by Algorithm 2.

Algorithm 4 Subdomain expansion algorithm.

Input:
- assignment $a_i \in \{1, \ldots, N_c\}$ of each mesh node $i \in \{1, \ldots, N\}$ to a cluster
- adjacency operator that lists the mesh neighbors neigh(i) of mesh node i
- label m for the spatial cluster to be expanded
- number of extra mesh layers N_m to add to the cluster m

Output: assignment $a_i \in \{1, \ldots, N_c\}$ of each mesh node $i \in \{1, \ldots, N\}$ to a cluster, with cluster m expanded by n_m additional layers of mesh nodes
1: $it \leftarrow 1$
2: **while** $it \leq n_m$ **do**
3: $\quad a_j \leftarrow m$ for all nodes i, j, where $a_i = m$, $j \in$ neigh(i), and $a_j \neq m$
4: $\quad it \leftarrow it + 1$
5: **end while**

For scalar problems, there is no difference between node-clustering and dof-clustering. For vector problems, the main difference is between the outcomes of both clustering approaches: in the former case, the dofs attached to a same node may be placed in different clusters; in the latter case, the dofs attached to a same node can always be placed in the same cluster. Node-clustering simplifies the implementation in an application software of the concept of a space-local right ROB. On the other hand, dof-clustering can be viewed as more natural for the space-local ROB concept proposed in this paper, as the local features of the various components of the solution may appear and evolve in different subdomains, However, for some applications, dof-clustering may complicate the expansion task of Algorithm 4 – and requires in any case its adaptation to the dof context – or may simply call to forgo this task and the conservatism in nonoverlapping mesh partitioning it enables.

4. State-space reduced-order bases and hyperreduction

Here, it is shown that the concept of a space-local right ROB can be combined with that of a state-local right ROB [2, 9] and with hyperreduction, to squeeze the most performance out of a nonlinear PROM. For this purpose, the concepts of a state-local right ROB and hyperreduction are also overviewed in order to keep this paper as self-contained as possible; however, details are omitted for clarity and brevity.

4.1. State-local reduced-order bases and their spatial partitioning

The traditional affine approximation (2) was extended in [2] to a piecewise affine approximation, in order to construct a simple (if not the simplest) nonlinear low-dimensional approximation $\tilde{\mathbf{u}}(t; \boldsymbol{\mu})$ of the solution $\mathbf{u}(t; \boldsymbol{\mu})$ that mitigates the effect of the Kolmogorov n-width on nonlinear PMOR. This extension led to the concept of a most-appropriate right *local* ROB, referred to here more precisely as

a most-appropriate state-local right ROB, where locality pertains to the region of the approximation manifold $\mathcal{M}$ where the current approximate solution lies. Hence, this concept requires partitioning first $\mathcal{M}$ into subregions $\mathcal{M}_i$; then constructing and assigning to each $\mathcal{M}_i$ a local, traditional (and therefore dense) right ROB $\mathbf{V}^{(i)}$. It also requires identifying online (and thus in real-time) the closest subregion $\mathcal{M}_\ell$ to the location on $\mathcal{M}$ of the current approximation of the high-dimensional solution; then advancing it (or for steady-state problems, updating it) in the subspace generated by the identified local right ROB – that is, the most-appropriate state-local right ROB $\mathbf{V}^{(\ell)}$. Recently, it was shown in [9] that when combined with LSPG for nonlinear PMOR and ECSW for hyperreduction, this approach is very computationally efficient: it enabled the acceleration by several orders of magnitude of the solution of a large-scale turbulent flow problem characterized by a complex aircraft geometry and $\mathcal{O}(10^8)$ dofs, while delivering exceptional accuracy.

In practice, the concept of a state-local right ROB is grounded in snapshot clustering. Its offline and online stages are typically implemented as follows (see also Figure 3):

1. **Offline**
   - Cluster the columns of the snapshot matrix $\mathbf{S}$ into k clusters, for example, using Algorithm 3. Then, partition $\mathbf{S}$ accordingly – that is, $\mathbf{S} = \left[\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(k)}\right]$, where $\mathbf{S}^{(j)} \in \mathbb{R}^{N \times N_s^{(j)}}$ and $N_s^{(j)}$ denotes the number of solution snapshots assigned to cluster j. Next, compute the centroid of each snapshot cluster matrix $\mathbf{S}^{(j)}$

     $$\mathbf{c}^{(j)} = \frac{1}{N_s^{(j)}} \sum_{j'=1}^{N_s^{(j)}} \mathbf{S}^{(j)}[\cdot, j']$$

     where $\mathbf{S}^{(j)}[\cdot, j']$ denotes the j'-th column of $\mathbf{S}^{(j)}$. This clustering defines a partitioning of the state space and enables the characterization of each point in the state space by its proximity to the nearest centroid $\mathbf{c}^{(\ell)}$.
   - Compresss each snapshot cluster matrix $\mathbf{S}^{(j)}$ using SVD. This generates a set of state-local right ROBs $\mathbf{V}^{(1)}, \ldots, \mathbf{V}^{(k)}$, where $\mathbf{V}^{(j)} \in \mathbb{R}^{N \times n^{(j)}}$ and $n^{(j)}$ denotes the size of the j-th state-local right ROB – determined, for example, using the total singular value energy criterion in cluster j.
2. **Online**
   - To advance the approximate solution from time-step t^m to time-step t^{m+1}: 1) identify the cluster whose centroid is nearest to the current reconstructed state-vector $\tilde{\mathbf{u}}^m$ – that is, identify the index

     $$\ell = \arg \min_{j \in \{1, \ldots, k\}} \left\| \tilde{\mathbf{u}}^m - \mathbf{c}^{(j)} \right\|_2$$

     then 2) construct the approximation $\tilde{\mathbf{u}}^{m+1}$ using the state-local right ROB $\mathbf{V}^{(\ell)}$. For this purpose, if the approximation $\tilde{\mathbf{u}}^m$ was computed using a different state-local right ROB $\mathbf{V}^{(\ell')}$, compute first the projection of $\tilde{\mathbf{u}}^{(m)}$ onto the subspace represented by $\mathbf{V}^{(\ell)}$, $\Pi_{\mathbf{V}^{(\ell)}} \mathbf{u}^m$, then apply the chosen time-integrator to this projected quantity to compute $\tilde{\mathbf{u}}^{m+1}$.

As presented above, the clustering of the solution snapshots leads to a nonoverlapping partitioning of the approximation manifold $\mathcal{M}$, which promotes discontinuities in the piecewise affine approximation of the solution. This issue can be addressed by a post-processing step, where in each cluster a few solution snapshots assigned to its neighbors are duplicated (see [2, 9]).

The concept a state-local right ROB blends well with that of a space-local right ROB, as both concepts are complementary: the former concept partitions a snapshot matrix column-wise; the latter one partitions it row-wise (see Figure 1 and Figure 3). Hence, both concepts can be combined by first clustering the columns of the snapshot matrix to create k snapshot matrix clusters $\mathbf{S}^{(1)}, \ldots, \mathbf{S}^{(k)}$ representing k subregions of the approximation manifold $\mathcal{M}$; then clustering the rows of each snapshot

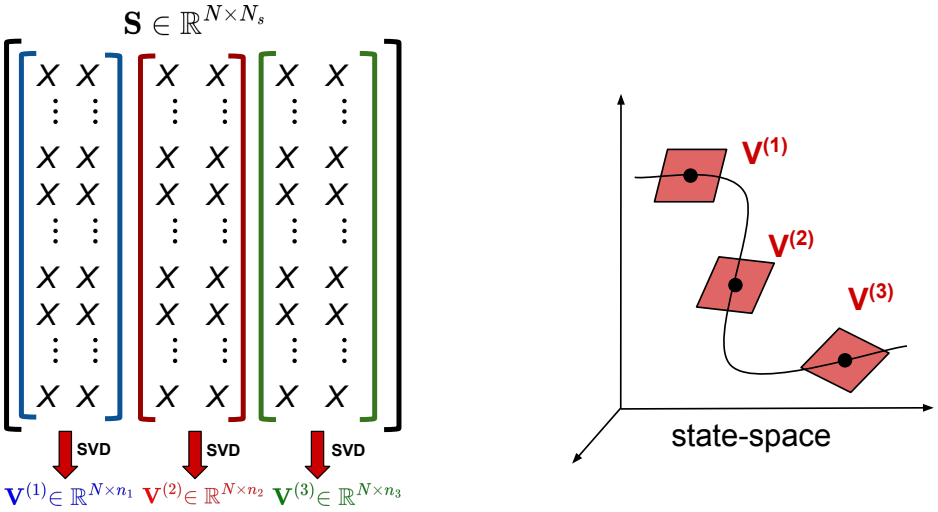

Figure 3: Construction and exploitation of state-local right ROBs: snapshot clustering and data compression within a cluster (left); and identification and exploitation at each time-step of the most-appropriate state-local right ROB $\mathbf{V}^{(\ell)}$.

matrix cluster $\mathbf{S}^{(j)}$ to partition it in N_c row-wise blocks, as follows

$$\mathbf{S}^{(j)} = \begin{bmatrix} \mathbf{S}_1^{(j)} \\ \vdots \\ \mathbf{S}_{N_c}^{(j)} \end{bmatrix} \tag{14}$$

where $\mathbf{S}_i^{(j)} \in \mathbb{R}^{N_i \times N_s^{(j)}}$ and $\sum_{i=1}^{N_c} N_i = N$.

In (14), each row-wise block $\mathbf{S}_i^{(j)}$ is separately compressed using SVD to compute a j-th space-local right ROB with components $\left\{\mathbf{V}_i^{(j)}\right\}_{i=1}^{N_c}$. Then, the components are concatenated to form k sparse, state-local right ROBs of the form

$$\mathbf{V}_{\text{loc}}^{(j)} = \text{diag}\left(\mathbf{V}_1^{(j)}, \ldots, \mathbf{V}_{N_c}^{(j)}\right), \quad j = 1, \ldots, k$$

In principle, each snapshot cluster matrix $\mathbf{S}^{(j)}$ could be partitioned using a different nonoverlapping domain decomposition: however, this would unnecessarily complicate the corresponding software implementation. In any case, the combination of both concepts of state-local and space-local right ROBs leads to the construction of k sparse state-local right ROBs that can be expected to be more computationally efficient for nonlinear PMOR than the k original state-local right ROBs.

4.2. Hyperreduction of state-local projection-based reduced-order models

For most highly nonlinear problems, those contributions to the construction and solution of the nonlinear PROM-based problem (4) whose complexity scales with the large dimension N of the HDM, cannot be pre-computed. For example, in the context of the nonlinear LSPG method, such contributions include: the evaluations of the Jacobian matrix $\mathbf{J}^{(k)}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu}), t^m;\boldsymbol{\mu}\right)$ and the residual vector $\mathbf{r}^{(k)}\left(\mathbf{u}_0 + \mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu}), t^m;\boldsymbol{\mu}\right)$; the matrix-matrix multiplication $\mathbf{J}^{(k)}\mathbf{V}$; and the solution of the least-squares problem (7). In this case, computational efficiency calls for hyperreducing (4) – that is, introducing approximations in the aforementioned computations that enable their execution at each time-step and for each queried parameter vector $\boldsymbol{\mu}$ with a complexity that is independent of N.

Currently, the discrete empirical interpolation method (DEIM) [1] is arguably the most popular hyperreduction method. However, the ECSW method [6] is chosen for this work for four main reasons: for computational structural dynamics and wave propagation applications, it provably preserves the

Lagrangian structure associated with Hamilton's principle for second-order dynamical systems and thus provably preserves the numerical stability properties of the chosen time-integration scheme [34]; for many CFD applications, it was numerically shown to be stable [8]; for challenging engineering computations, it was shown to accelerate solution time by more than three orders of magnitude (for example, see [9]); and the developments to be presented here are anyway independent of the specifics of the chosen hyperreduction method.

ECSW can be described in one line as a cubature formula for approximating projected matrices and vectors. For example, noting that the reduced-order residual (4) can be written as

$$\mathbf{r}_n\left(\mathbf{y}(t^m;\boldsymbol{\mu}),t^m;\boldsymbol{\mu}\right)=\mathbf{W}^T\mathbf{r}\left(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu}),t^m;\boldsymbol{\mu}\right)=\sum_{e\in\mathcal{E}}\mathbf{W}^T\mathbf{L}_e^T\mathbf{r}^e\left(\mathbf{L}_{e+}\left(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu})\right),t^m;\boldsymbol{\mu}\right) \tag{15}$$

ECSW approximates this projected quantity as

$$\mathbf{r}_n(\mathbf{y}(t^m;\boldsymbol{\mu}),t^m;\boldsymbol{\mu})\approx\widetilde{\mathbf{r}}_n(\mathbf{y}(t^m;\boldsymbol{\mu}),t^m;\boldsymbol{\mu})=\sum_{e\in\widetilde{\mathcal{E}}\subset\mathcal{E}}\xi_e\mathbf{W}^T\mathbf{L}_e^T\mathbf{r}^e(\mathbf{L}_{e+}(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu})),t^m;\boldsymbol{\mu}) \tag{16}$$

In (15) and (16) above:

- $\mathcal{E}$ denotes the computational mesh underlying the HDM.
- $\mathbf{L}_e\in\{0,1\}^{d_e\times N}$ is the Boolean matrix that localizes a high-dimensional vector of size N to the d_e dofs associated with the mesh entity e (e.g. the finite volume cell, finite element, or finite differencing mesh node e).
- $\mathbf{L}_{e+}\in\{0,1\}^{d_{e+}\times N}$ is the Boolean matrix that localizes a high-dimensional vector of size N to the $d_{e+}\geq d_e$ dofs attached to the mesh entity e and a set of neighboring mesh entities determined by the stencil of the semi-discretization method underlying the construction of the HDM.
- $\mathbf{r}^e\left(\mathbf{L}_{e+}(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu})),t^m;\boldsymbol{\mu}\right)\in\mathbb{R}^{d_e}$ is the contribution of mesh entity e to the high-dimensional residual.
- $\widetilde{\mathcal{E}}\subset\mathcal{E}$ is a subset of $\mathcal{E}$ whose mesh entities e constitute the "points" of the cubature rule. Typically, $\left(\widetilde{N}_e=\left|\widetilde{\mathcal{E}}\right|\right)\ll(N_e=|\mathcal{E}|)$, where $|\clubsuit|$ denotes the cardinal of the set $\clubsuit$.
- $\{\xi_e\}_{e\in\mathcal{E}}$ is the set of "weights" of the cubature rule. These weights are restricted to be positive for reasons that are explained in [6] and [9].

Similarly, ECSW approximates the projected Jacobian, which is needed by Newton's method to solve the nonlinear system of reduced-order equations $\widetilde{\mathbf{r}}_n(\mathbf{y}(t^m;\boldsymbol{\mu}),t^m;\boldsymbol{\mu})=0$, as follows

$$\widetilde{\mathbf{J}}_n(\mathbf{y}(t^m;\boldsymbol{\mu});\boldsymbol{\mu})=\sum_{e\in\widetilde{\mathcal{E}}\subset\mathcal{E}}\xi_e\mathbf{W}^T\mathbf{L}_e^T\mathbf{J}^e(\mathbf{L}_{e+}(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu}));\boldsymbol{\mu})\mathbf{L}_{e+}\mathbf{V}$$

where $\mathbf{J}^e(\mathbf{L}_{e+}(\mathbf{u}_0+\mathbf{V}\mathbf{y}(t^m;\boldsymbol{\mu}));\boldsymbol{\mu})\in\mathbb{R}^{d_e\times d_{e+}}$ is the Jacobian of $\mathbf{r}^e$ with respect to $\mathbf{y}$.

Most importantly, ECSW determines the set of cubature points $\widetilde{\mathcal{E}}$ and the corresponding set of weights $\Xi_{\widetilde{\mathcal{E}}}=\left\{\xi_e\mid e\in\widetilde{\mathcal{E}}\right\}$ using a machine learning approach based on the loss function

$$\|\mathbf{C}\boldsymbol{\zeta}-\mathbf{d}\|_2$$

where $\left(\mathbf{C}\in\mathbb{R}^{N_s\times N_e},\mathbf{d}\in\mathbb{R}^{N_e}\right)$ is the pair of training matrix and training vector obtained by requiring that for the generalized coordinates of all collected solution snapshots, the cubature rule (15) is *exact* for $\boldsymbol{\zeta}=\mathbf{1}$, where $\mathbf{1}$ denotes the N_e-dimensional vector of ones. Specifically, ECSW determines $\widetilde{\mathcal{E}}$ and $\Xi_{\widetilde{\mathcal{E}}}$ by solving the following non-negative least-squares (NNLS) problem

$$\begin{aligned}&\text{minimize }\|\mathbf{C}\boldsymbol{\zeta}-\mathbf{d}\|_2^2\\&\text{subject to }\boldsymbol{\zeta}\geq 0\end{aligned}$$

equipped with the threshold-based early termination criterion

$$\|\mathbf{C}\boldsymbol{\zeta} - \mathbf{d}\|_2 \leq \varepsilon \|\mathbf{d}\|_2 \tag{17}$$

where $\varepsilon > 0$ is a specified small tolerance.

Now, note that a sparse right-ROB $\mathbf{V}$ accelerates the hyperreduction approximation (15) in the same manner it accelerates any other PROM-based computation. Hence, a space-local right ROB $\mathbf{V}_{\text{loc}}$ is beneficial to hyperreduction in the same way it is beneficial to PMOR. Furthermore, the use of a space-local right ROB in the cubature rule (15) does not incur any modification to the mesh sampling *procedure* underlying the ECSW method, or for that matter, any other hyperreduction procedure. However, it motivates its application to the nonoverlapping subdomains $\{\Omega_i\}_{i=1}^{N_c}$ in an embarrassingly parallel fashion and the construction of one reduced mesh "component" for each separate subdomain. In this way, the tolerance ε can be tailored to each specific subdomain Ω_i, according to the feature richness/poorness of the solution in this subdomain, and more importantly, its evolution with $\boldsymbol{\mu} \in \mathcal{P}$.

5. Applications

The proposed concept of a space-local ROB for accelerating PROM-based computations through sparsity was implemented in the compressible flow solver AERO-F [35, 36] and integrated in its nonlinear PMOR capabilities. It is illustrated here by treating the following two CFD applications:

1. The prediction of an unsteady fluid flow around a NACA 0012 airfoil that exhibits flow separation and vortex shedding. In this case, the flow is modeled using the Reynolds-averaged Navier-Stokes (RANS) equations equipped with the Spalart-Allmaras turbulence model [37]. The problem is two-dimensional but the flow computations are performed in three dimensions because AERO-F is a three-dimensional flow solver. PMOR is performed using the traditional affine approximation and LSPG but without hyperreduction, in order to focus exlusively on the discussion of the performance of the novel elements presented in this paper. Consequently, the wall-clock time performance of the space-local PROM is contrasted only with that of its traditional counterpart: indeed, in the absence of hyperreduction, no significant speedup factor can be expected with respect to the performance of the HDM. For this academic application, both FFT-k-means and far-field/near-field approaches are considered for decomposing the computational fluid domain into two nonoverlapping subdomains.
2. The prediction of the unsteady, turbulent wake flow behind the Ahmed body – which is a benchmark CFD problem in the automotive industry. In this case: the flow is modeled using the detached eddy simulation (DES) equations [38]; the HDM is of a much higher dimension than in the previous case; LSPG is performed using sparse state-local right ROBs obtained by combining both concepts of state-local and space-local right ROBs; the resulting nonlinear reduced-order governing equations do not exhibit a polynomial dependence on the fluid state variable and therefore hyperreduction is performed using ECSW. For this industrial-grade application, the far-field/near-field approach is considered for decomposing the computational fluid domain into two nonoverlapping subdomains.

In both applications outlined above, the fluid is air and is modeled as a perfect gas; the HDM (1) is constructed by discretizing the three-dimensional inviscid fluxes using a second-order FV upwind scheme based on the approximate Riemann solver and the three-dimensional viscous fluxes and source term are approximated by a piecewise linear FE method; time-discretization is performed using the three-point implicit backward difference formula equipped with a Newton-Krylov solver and the additive Schwartz preconditioned GMRES algorithm [39]; and the accuracy delivered by the nonlinear PROMs is measured using a relative root mean squared error (RMSE) metric as follows

$$\mathbb{RE}_{\mathbb{Q}} = \frac{\sqrt{\sum_{t \in T_Q} (Q(t) - \widetilde{Q}(t))^2}}{\sqrt{\sum_{t \in T_Q} Q(t)^2}} \tag{18}$$

In (18) above, $Q(t)$ is the QoI, $\widetilde{Q}(t)$ denotes its PROM-based approximation, and T_Q is the set of evenly-spaced times-teps at which the QoIs are computed.

All simulations are performed in double precision arithmetic on a Linux cluster, where each compute node is equipped with two 12-core Intel Xeon Gold 5118 processors running at 2.30 GHz and 192 GB of memory.

5.1. NACA 0012 airfoil: vortex-shedding due to flow separation

5.1.1. High-dimensional CFD model

For this RANS application, the chord length of the airfoil is set to $c = 1$ m; and the free-stream Mach number, free-stream pressure, free-stream density, and angle of attack are set to $M_\infty = 0.4$, $p_\infty = 10^5$ Pa, $\rho_\infty = 1.3$ kg/m^3, and $\alpha = 10°$, respectively. The corresponding Reynolds number based on the chord length is $Re = 4.47 \times 10^4$. These flow conditions promote flow separation and vortex shedding in a realtively small region aft of the top surface of the airfoil, which makes this application a good demonstration of the approach proposed in this paper for accelerating nonlinear PMOR.

The computational domain Ω is chosen as a disk of diameter $20\,c = 20$ m. It is discretized using a one-element-thick, unstructured, three-dimensional mesh with 165,460 nodes and 474,362 tetrahedral elements. Figure 4 shows the computational mesh in the vicinity of the airfoil. Symmetry boundary conditions are applied on the spanwise faces of the domain to ensure that the computed flow is two-dimensional. A no-slip adiabatic wall boundary condition is applied on the wall boundary of Ω; and the Steger-Warming flux splitting method [40] is applied on the far-field boundaries of Ω to enforce the far-field boundary conditions. The resulting semi-discrete HDM has the dimension $N = 992,760$ (five conservative fluid variables and one turbulence modeling variable per node). The unsteady flow computation is initialized from the results of a quasi-steady simulation performed using the same boundary conditions and in a sufficiently long time-interval to reach the onset of a statistically stationary flow. Then, time-integration is performed until $T_f = 0.025$ s using the constant time-step size $\Delta t = 5 \times 10^{-5}$ s and therefore 500 time-steps.

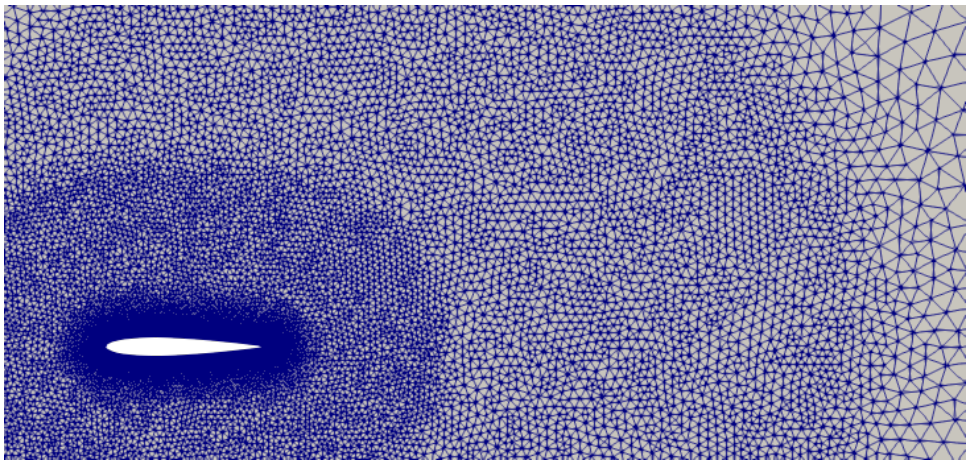

Figure 4: Discretization of a computational fluid domain around a NACA 0012 airfoil.

The HDM-based unsteady simulation is carried out on a single core of the aforementioned Linux cluster: it requires 5.5 hrs wall-clock time. The top-left part of Figure 5 displays the contour plots of the local Mach number computed at $T_f = 0.025$ s. It reveals the presence of flow separation features on the top surface of the airfoil and vortex shedding behind it.

Figure 6 displays in black lines the time-histories of four computed QoIs: the lift and drag coefficients; and the flow velocity (magnitude) and pressure at a probe located 0.11 chord-lengths downstream of the airfoil and 0.13 chord-lengths above it (in a coordinate system where one axis is aligned with the chord). The probe data are normalized by their values at the free-stream.

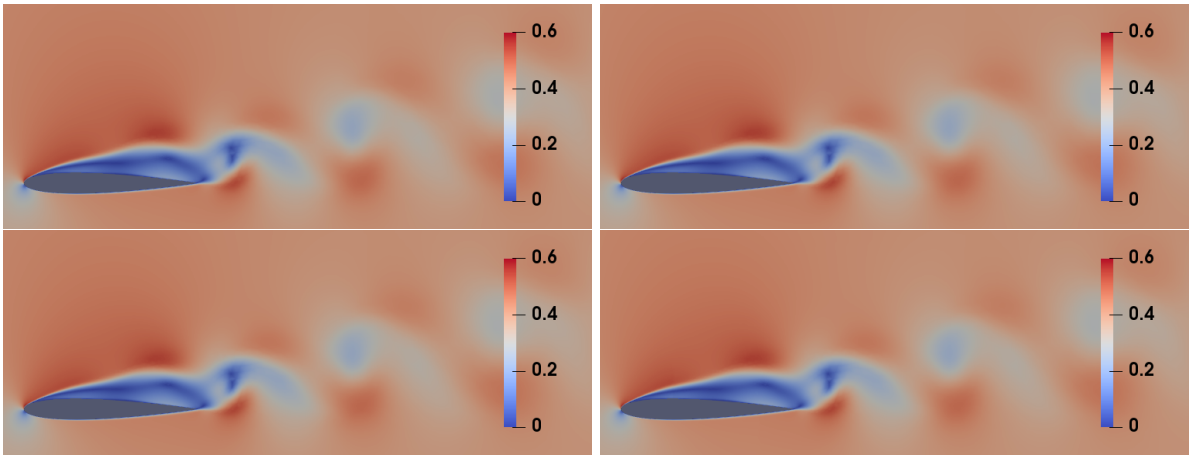

Figure 5: Contour plots of the local Mach number computed around a NACA 0012 airfoil at $T_f = 0.025$ s using: the HDM (top-left); a traditional LSPG PROM based on a dense right ROB (top-right); a counterpart LSPG PROM based on a space-local sparse right ROB constructed using the automated FFT-k-means domain decomposition approach (bottom-left); and a counterpart LSPG PROM based on a space-local sparse right ROB constructed using the manual near-field/far-field domain decomposition approach (bottom-right).

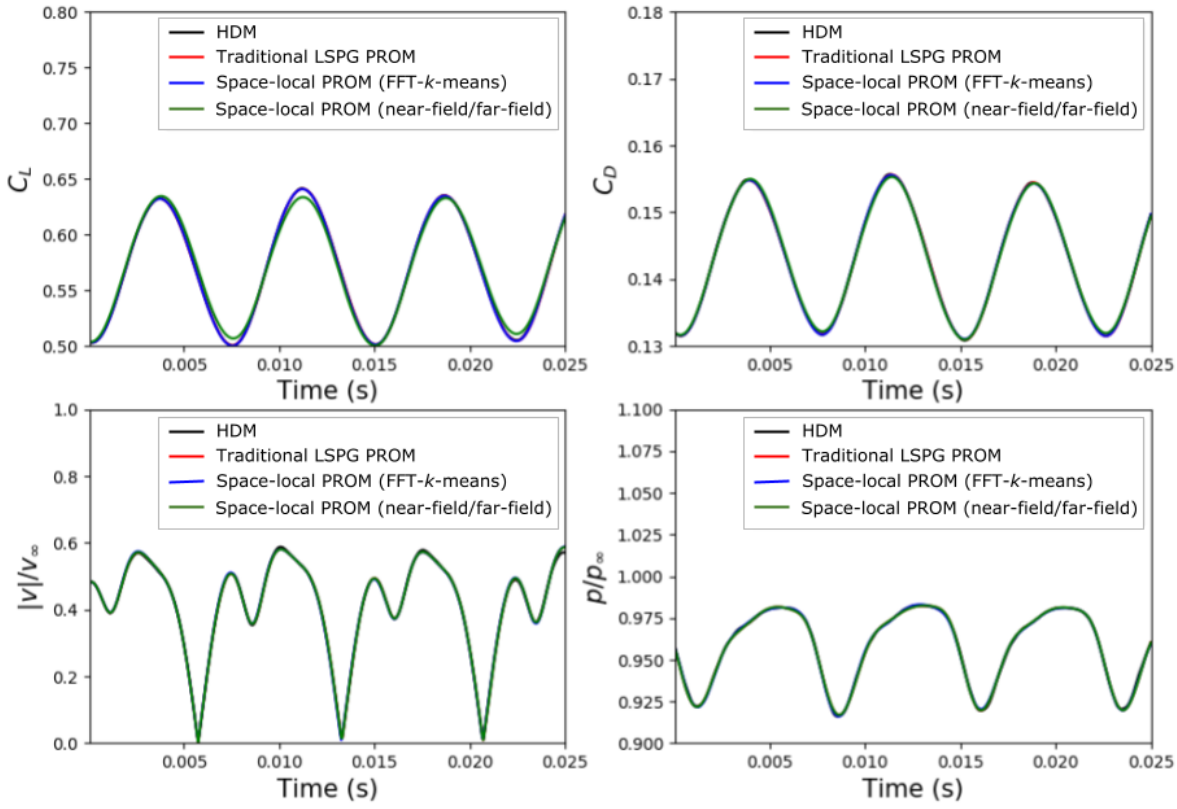

Figure 6: Time-histories of four QoIs numerically predicted using various computational models (NACA 0012): lift coefficient (top-left); drag coefficient (top-right); normalized flow velocity at a probe (bottom-left); and normalized pressure at the same probe (bottom-right).

5.1.2. Space-local PROMs

Both nonoverlapping domain decomposition approaches described in Section 3 are applied to construct two space-local LSPG PROMs of the same dimension. A similarly accurate traditional LSPG

PROM is also built for the purpose of performance comparisons. In all cases, 500 solution snapshots are collected at the sampling frequency implied by $\Delta t_s = 5 \times 10^{-5}$ s in the time-interval $t \in [0, 0.025]$ s – that is, at every time-step of the HDM-based simulation.

The three LSPG PROMs are constructed as follows:

- The traditional one based on a conventional, POD-based, dense right ROB is constructed using the singular value energy criterion (10) with $\tau = 99.99\%$, which leads to a PROM of dimension $n = 17$.
- The first space-local PROM is constructed using the FFT-k-means algorithm (Algorithm 2) to decompose the computational fluid domain into two nonoverlapping subdomains and Algorithm 4 to expand by five layers of mesh nodes that subdomain expected to contain flow features and denoted here by Ω_1. From the comparison of the top part of Figure 7 with any part of Figure 5, the reader can observe that in this case, Algorithm 2 alone – and certainly the combination of Algorithm 2 and Algorithm 4 – yields a nonoverlapping domain decomposition where Ω_1 contains both regions of flow separation and vortex shedding. Then, a right ROB is constructed in each subdomain using the singular value energy criterion (10) with $\tau = 99.99\%$, which leads to: a right ROB in Ω_1 of dimension $n_1 = 16$; a right ROB in Ω_2 of dimension $n_2 = 12$; and therefore a space-local sparse right ROB of dimension $n_{\text{loc}} = 28$.
- The second space-local PROM is constructed by applying the manual near-field/far-field heuristic to decompose the computational fluid domain into two nonoverlapping subdomains Ω_{NF} and Ω_{FF} (see the bottom part of Figure 7). Again, the comparison of any part of Figure 5 with the bottom part of Figure 7 shows that Ω_{NF} contains both regions of flow separation and vortex shedding. Then, a right ROB is constructed in each subdomain using separate singular value energy criteria (10) that account for the feature richness/poorness of the solution: $\tau_{\text{NF}} = 99.99\%$ in Ω_{NF}; and $\tau_{\text{FF}} = 99\%$ in Ω_{FF}. This leads to: a right ROB in Ω_{NF} of dimension $n_1 = 17$; a right ROB in Ω_{FF} of dimension $n_2 = 4$ only; and therefore a space-local sparse right ROB of dimension $n_{\text{loc}} = 21$. The ability to tune the truncation criterion (10) to each subdomain allows the computational effort to focus in this case on the vortex-shedding region where the solution is most important and complex.

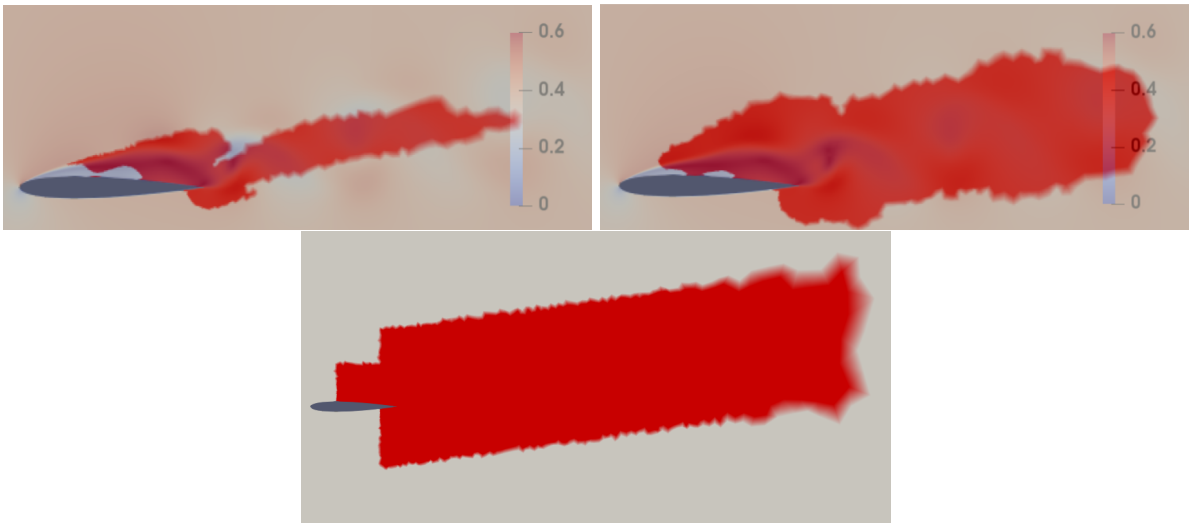

Figure 7: Decomposition of the computational fluid domain around the NACA 0012 airfoil into two nonoverlapping subdomains using: Algorithm 2, with Ω_1 shown in red and Ω_2 shown in grey (top-left); Algorithm 2 and Algorithm 4, with Ω_1 shown in red and Ω_2 shown in grey (top-right); and the near-field/far-field heuristic with Ω_{NF} shown in red and Ω_{FF} shown in grey.

Figure 6 shows that all nonlinear LSPG PROMs outlined above deliver similar excellent levels of accuracy, for all considered QoIs. The relative error results reported in Table 2 confirm this observation.

LSPG PROM type	$\mathbb{RE}_{C_L}$ (%)	$\mathbb{RE}_{C_D}$ (%)	$\mathbb{RE}_{v}$ (%)	$\mathbb{RE}_{p}$ (%)
Traditional (dense right ROB)	0.08	0.06	0.8	0.07
Space-local (FFT-k-means)	0.08	0.08	1.0	0.07
Space-local (near-field/far-field)	0.70	0.20	0.8	0.08

Table 1: Unsteady flow problem around a NACA 0012 (RANS): accuracy of the constructed nonlinear LSPG PROMs.

Table 2 presents relevant dimensionality and sparsity data. It shows that while each constructed space-local PROM has a higher dimension than the similarly accurate traditional PROM, the associated space-local right ROB has almost 50% fewer nonzero entries per dof ($n\gamma$) and therefore a strong potential for accelerating the PROM-based computations. This potential is confirmed by the results discussed in the next section.

LSPG PROM type	n	n_1	n_2	γ	$n\gamma$
Traditional (dense)	17	–	–	1.000	17.00
Space-local (FFT-k-means)	28	16	12	0.477	13.38
Space-local (near-field/far-field)	21	16	4	0.483	10.15

Table 2: Unsteady flow problem around a NACA 0012 (RANS): dimensions and sparsity of the constructed nonlinear LSPG PROMs.

5.1.3. Wall-clock performance results

All PROM-based simulations are also carried out on a single core of the same Linux cluster: their wall-clock time performance results are reported in Table 3.

The simulation based on the traditional nonlinear LSPG PROM constructed using a traditional dense right ROB consumes 3.97 hrs wall-clock time and therefore performs 1.39 times faster than its HDM-based counterpart. This is because without hyperreduction, no substantial speedup relative to HDM-based computations can be achieved for such a nonlinear problem. As stated earlier, the main interest for this example is in the speedup factor that can be delivered by a space-local PROM relative to a traditional counterpart.

The simulation based on the similarly accurate space-local PROM constructed using the FFT-k-means algorithm for nonoverlapping domain decomposition completes in 3.54 hrs. Hence, it delivers a speedup factor of 1.11 relative to its counterpart based on the traditional PROM. That based on the similarly accurate space-local PROM constructed using the near-field/far-field heuristic for nonoverlapping domain decomposition completes in 3.13 hrs. Thus, it achieves a speedup factor of 1.27 relative to the same reference. Larger speedup factors are achieved in the second case by tuning the heuristic (since it is performed manually), but the emphasis here is on demonstrating the potential for a speedup factor of a simple, hands-off nonoverlapping domain decomposition approach.

LSPG PROM type	Wall-clock time (hrs)	Wall-clock speedup factor
Traditional (dense)	3.97	–
Space-local (FFT-k-means)	3.54	1.11
Space-local (near-field/far-field)	3.13	1.27

Table 3: Unsteady flow problem around a NACA 0012 (RANS): wall-clock time performance of the various nonlinear LSPG PROMs.

5.2. Turbulent flow in the near-wake of an Ahmed body

5.2.1. High-dimensional CFD model

The Ahmed body [41] is a simplified vehicle geometry. Predicting the turbulent flow past this body – and specifically, the highly separated turbulent flow behind it – is a benchmark problem in the automotive industry. The specific instance adopted here has a 20° rear slant angle.

For this application, all computations are performed in nondimensional form, for the free-stream velocity $v_\infty = 60$, the angle of attack $\alpha = 0°$, and the Reynolds number based on the vehicle length $Re = 4.29 \times 10^6$. As stated earlier, turbulence is modeled for this application using a DES approach based on the one-equation Spalart-Allmaras turbulence model.

The chosen computational fluid domain Ω is shown in Figure 8. Specifically, the body is placed in a box of a length of 7.5 body lengths (with the front of the body 2.4 body lengths away from the inlet boundary), a width of 2.5 body widths, and a height of 4.1 body heights. Due to symmetry, only half of the body is modeled: the other half is replaced by a symmetry plane. Hence, Ω is delimited by inlet and outlet boundaries, a symmetry plane, and a ground plane.

The computational fluid domain is discretized by an unstructured mesh with 2,890,434 mesh nodes and 17,017,090 tetrahedral elements. Adiabatic no-slip boundary conditions are applied on the wall boundary of the body and on the ground. Reichardt's law of the wall is used to model the flow immediately adjacent to these surfaces. The far-field boundary conditions are enforced on the inlet and outlet boundaries using the Steger-Warming flux splitting method. The resulting semi-discrete HDM has the dimension $N = 17,342,604$ (five conservative fluid variables and one turbulence-modeling variable per mesh node). The unsteady flow simulation is initialized from the results of a quasi-steady simulation performed using the same boundary conditions. Then, time-integration is performed in the time-interval $t \in [0, 0.2]$, using the constant time-step size $\Delta t = 8 \times 10^{-5}$. Hence, this time-interval includes some startup transients as well as a period of statistically stationary turbulent flow.

The HDM-based simulation is performed on 120 cores of the aforementioned Linux cluster: it requires 29.10 hrs wall-clock time to complete and therefore 3,490.43 hrs of CPU time. A snapshot of the predicted flow is displayed in Figure 9 in the form of vorticity magnitude iso-surfaces colored by the local Mach number.

For this application, the chosen QoIs are: the lift and drag coefficients of the body; and the streamwise and vertical components of the flow velocity vector at a point in Ω located in the symmetry plane, half of a body-length downstream of the body, and half of a body-height above the ground. The time-histories of these QoIs are displayed in Figure 10.

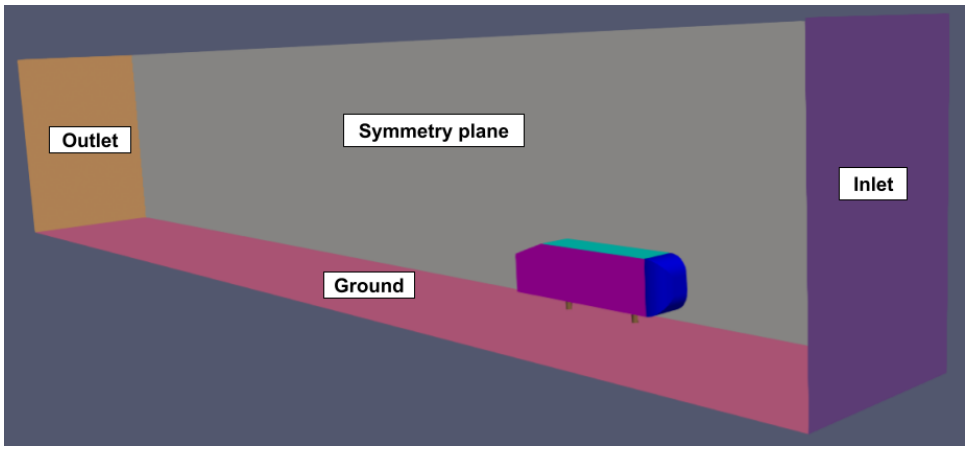

Figure 8: Geometry of the Ahmed body and its computational fluid domain (one half of the body is modeled and symmetry boundary conditions are specified in the symmetry plane).

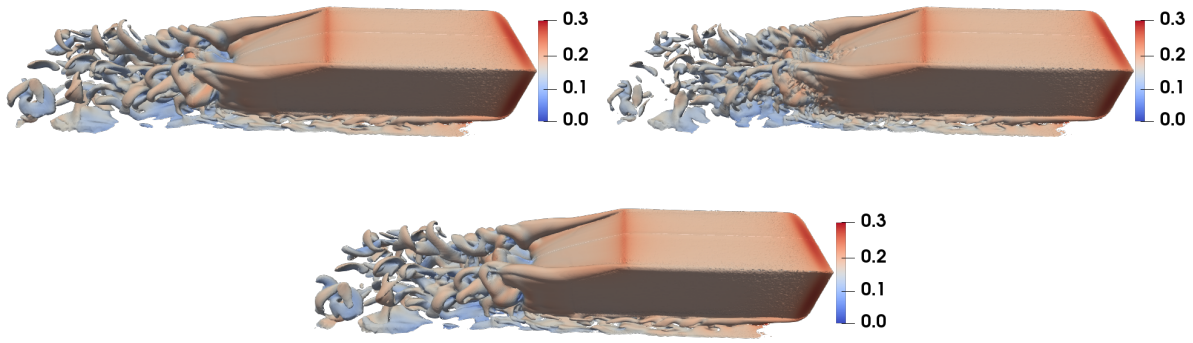

Figure 9: Vorticity magnitude iso-surfaces colored by the local Mach number and computed around the Ahmed body at $t = 0.2$ using: the HDM (top-left); a set of traditional state-local LSPG HPROMs based on traditional dense right ROBs (top-right); and a counterpart set of LSPG HPROMs based on a set of sparse state-local right ROBs built using the near-field/far-field domain decomposition heuristic.

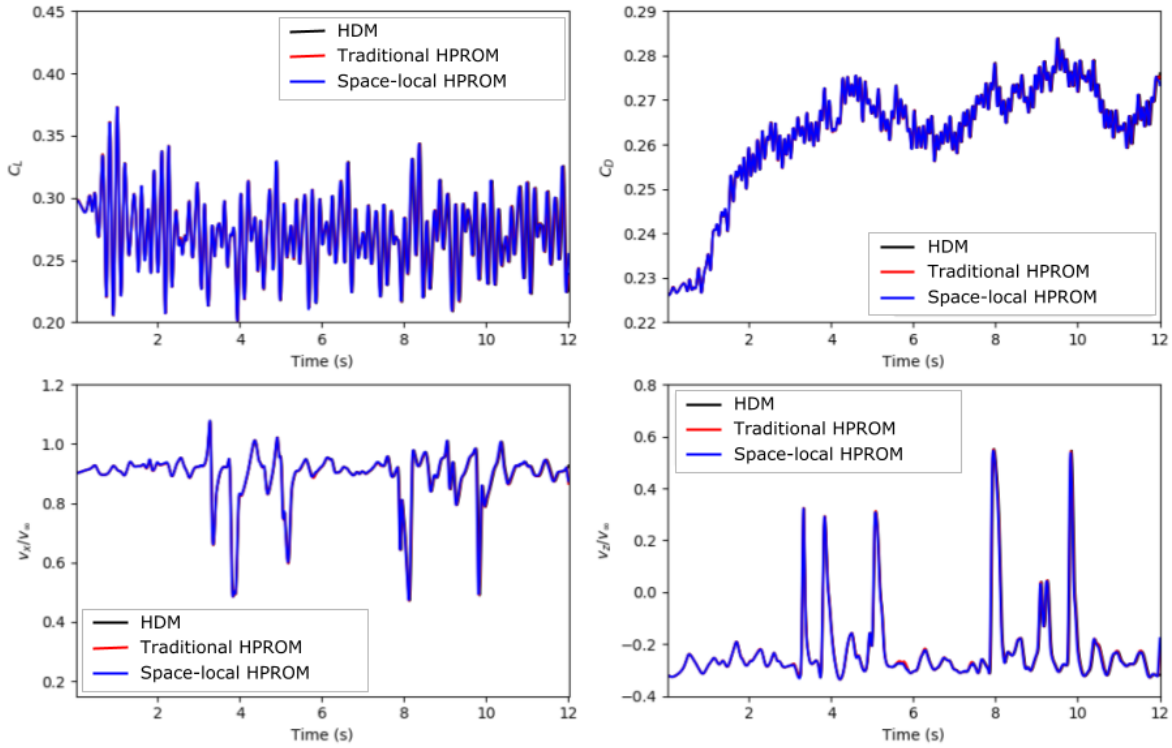

Figure 10: Time-histories of four QoIs numerically predicted using various computational models (Ahmed body): lift coefficient (top-left); drag coefficient (top-right); normalized streamwise component of the flow velocity vector at a probe (bottom-left); and normalized vertical component of the flow velocity vector at the same probe (bottom-right).

5.2.2. Sparse state-local hyperreduced projection-based reduced-order model

The near-field/far-field heuristic described in Section 3.3.1 is applied to construct 10 sparse state-local right ROBs and the corresponding set of local LSPG PROMs, for the DES of the flow past the Ahmed body. It partitions the CFD mesh into the two nonoverlapping subdomains Ω_{NF} and Ω_{FF} shown in Figure 11. The comparison of this figure with any part of Figure 9 shows that Ω_{NF} contains the regions of flow separation and vortex shedding.

A similarly accurate set of 10 traditional state-local right ROBs and the corresponding set of tradi-

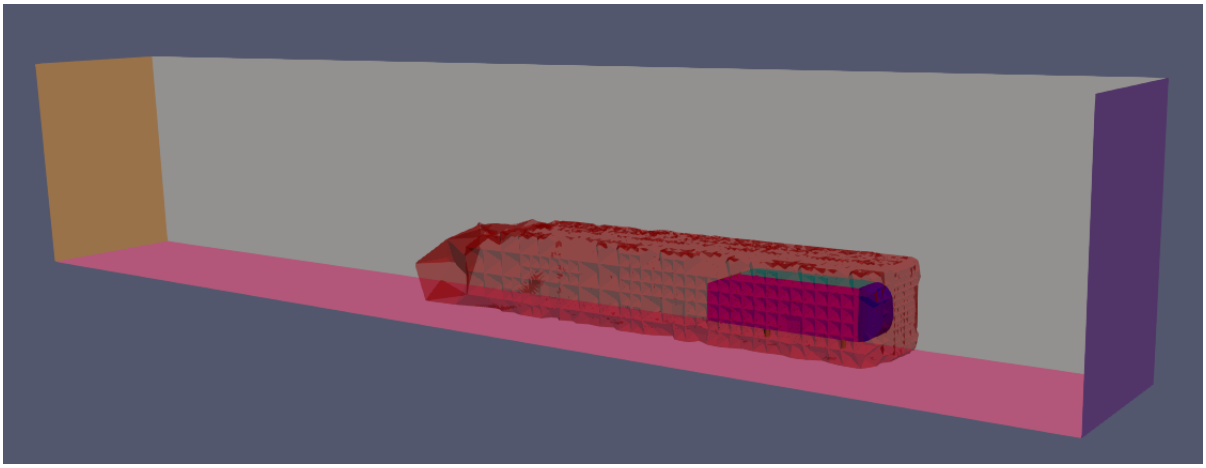

Figure 11: Decomposition of the computational fluid domain around the Ahmed body into two nonoverlapping subdomains using the near-field/far-field heuristic: Ω_{NF} is shown in translucent red and Ω_{FF} is shown in grey.

tional local LSPG PROMs are also built, for the purpose of performance comparisons. In both cases, 1,251 solution snapshots are collected at the sampling frequency implied by $\Delta t_s = 1.6 \times 10^{-4}$ in the time-interval $t \in [0, 0.2]$ – that is, every two time-steps of the HDM-based simulation.

In both cases:

- The snapshots are clustered into 10 clusters using the k-mean algorithm and 10% of the number of snapshots assigned to each cluster are duplicated in its neighbors to create overlapping and promote continuity of the piecewise affine approximation. All 10 state-local right ROBs are constructed using SVD and the singular value energy criterion (10) with $\tau = 99.99\%$. In the case of the sparse state-local right ROB, the construction strategy described in the bullet S1 of Section 3.2 is adopted – that is, basis vectors are added to each component of the space-local ROB in order of decreasing singular value until the singular value energy criterion is met with $\tau = 99.99\%$.

- Hyperreduction is performed using the ECSW method tailored for LSPG [9] and 17 evenly spaced solution snapshots sampled at the frequency implied by $\Delta t_H = 1.25 \times 10^{-3}$ in the time-interval $t \in [0, 2 \times 10^{-1}]$. It transforms every LSPG PROM into a hyperreduced LSPG PROM (HPROM). The reduced mesh built for the set of traditional state-local LSPG HPROMs associated with the set of traditional state-local right ROBs is obtained using the tolerance $\varepsilon = 1 \times 10^{-3}$ in (17). That constructed for the set of state-local LSPG HPROMs associated with the set of sparse state-local right ROBs is obtained using two different tolerances: $\varepsilon_{\text{NF}} = 1 \times 10^{-3}$ in Ω_{NF}, where the solution is expected to be feature-rich; and $\varepsilon_{\text{FF}} = 1 \times 10^{-2}$ in Ω_{FF}, where it is expected to be feature-poor. Figure 12 shows both reduced meshes.

Figure 10 shows that both sets of state-local LSPG HPROMs deliver similar excellent levels of accuracy, for all considered QoIs. The relative error results reported in Table 5 confirm this observation.

LSPG HPROM type	$\mathbb{RE}_{C_L}$ (%)	$\mathbb{RE}_{C_D}$ (%)	$\mathbb{RE}_{v_x}$ (%)	$\mathbb{RE}_{v_z}$ (%)
Traditional (dense)	5.0	0.60	3.1	4.7
Sparse state-local (near-field/far-field)	6.5	0.74	3.9	6.1

Table 4: Turbulent flow problem around the Ahmed body (DES): accuracy of the constructed nonlinear LSPG HPROMs.

Table 5 presents for each set of state-local LSPG HPROMs relevant dimensionality and sparsity data averaged across all its state-local right ROBs. It shows that while on average, a sparse state-local right ROB has an 18% higher dimension than a similarly accurate traditional state-local right ROB, it has about 8% fewer nonzero entries per dof ($n\gamma$). More importantly, the ability to tune the ECSW threshold in (17) to each specific subdomain Ω_i, according to the feature richness/poorness of the solution in this

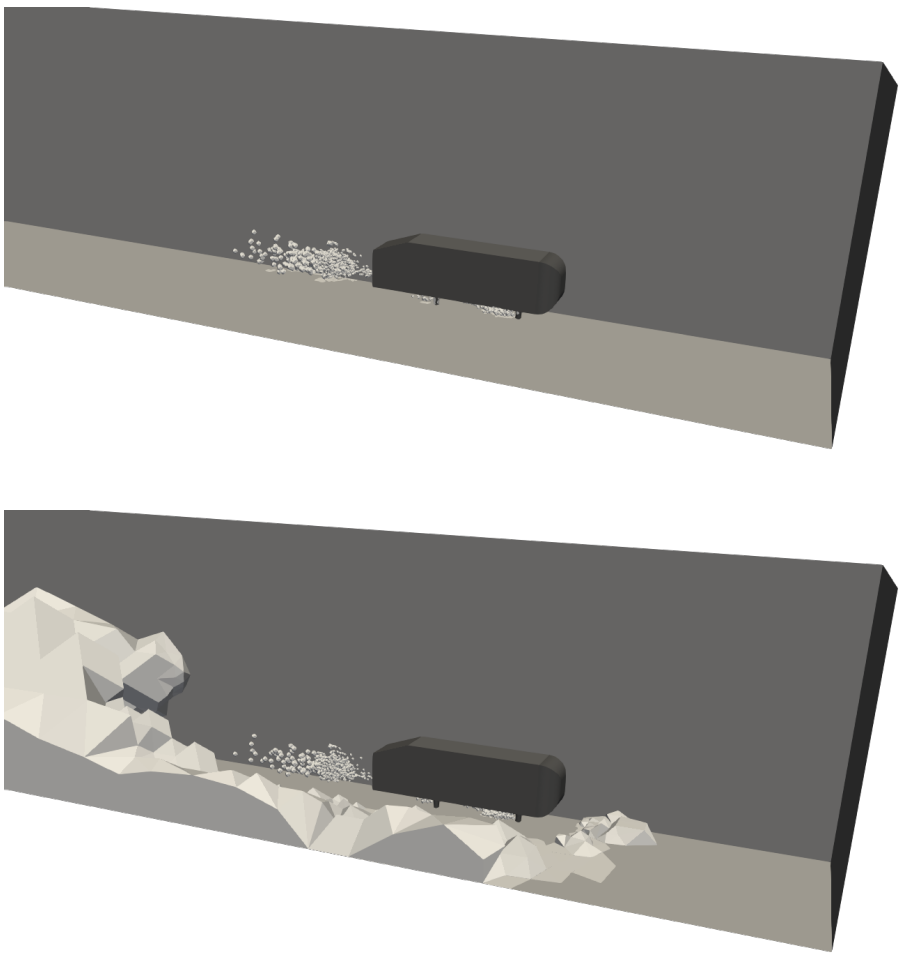

Figure 12: ECSW-generated reduced meshes generated for: the set of traditional state-local LSPG PROMs associated with the set of traditional state-local right ROBs; and the set of state-local LSPG PROMs associated with the set of sparse state-local right ROBs (right).

subdomain, leads in this case to a reduced mesh for the set of state-local LSPG HPROMs associated with the set of sparse state-local right ROBs that is almost twice smaller than its counterpart for the set of traditional state-local LSPG HPROMs. This suggests a strong potential for sparse state-local right ROBs to accelerate HPROM-based computations. This potential is confirmed by the results discussed in the next section.

LSPG HPROM type	n_{avg}	γ_{avg}	$(n\gamma)_{\text{avg}}$	$\widetilde{N}$
Traditional (dense)	113.7	1.000	113.70	3,793
Sparse state-local (near-field/far-field)	133.6	0.783	105.59	1,706

Table 5: Turbulent flow problem around the Ahmed body (DES): dimensions and sparsity of the constructed nonlinear LSPG HPROMs.

5.2.3. Wall-clock performance results

Both state-local HPROM-based simulations are carried out on eight cores of the same Linux cluster: their performance results are reported in Table 6. These results show that the set of traditional state-local HPROMs completes the simulation 22.5 times faster than the HDM wall-clock time and consumes

roughly 338 times less CPU time. The set of state-local HPROMs constructed using sparse state-local right ROBs completes the simulation 34.8 times faster than the HDM – and therefore more than 1.5 times faster than the counterpart set of traditional state-local HPROMs. Hence, the results reported in Table 6 highlight the potential of sparse state-local right ROBs to accelerate HPROM-based computations.

Computational model	Wall-clock time (hrs)	CPU time (hrs)	Wall-clock speedup factor	CPU speedup factor
HDM	29.00	3,490.80	–	–
State-local HPROMs (traditional)	1.29	10.30	22.5	338.2
State-local HPROMs (sparse)	0.84	6.68	34.8	522.4

Table 6: Turbulent flow problem around the Ahmed body (DES): wall-clock time and CPU time performance results for the nonlinear LSPG HPROMs.

6. Conclusions

In this paper, the concept of a space-local right reduced-order basis (ROB) is developed to introduce sparsity in a projection-based reduced-order model (PROM) and accelerate its processing. This concept relies on decomposing the computational domain into nonoverlapping subdomains, computing in each subdomain an independent local ROB, and concatenating all local ROBs to construct a sparse right ROB referred to as a space-local right ROB. This concept is compatible with both concepts of state-local ROBs and hyperreduction, both of which are essential for nonlinear projection-based model order reduction. It is shown in this paper that manual and simple as well as automated and sophisticated domain decomposition strategies can be developed to implement the proposed concept. It is also shown that this concept enhances the computational efficiency of hyperreduction, at least when it is performed using the energy-conserving sampling and weighting (ECSW) method [6, 34, 9]. Most importantly, it is shown that for a large-scale turbulent flow problem with $O(10^6)$ degrees of freedom, the proposed concept of a space-local right ROB and its integration with that of a state-local right ROB accelerate the performance of a traditional nonlinear PROM by a factor 1.5. This is significant considering that a traditional nonlinear PROM can itself accelerate the performance of a high-dimensional model by several orders of magnitude.

7. Acknowledgements

Spenser Anderson acknowledges partial support by the Air Force Office of Scientific Research under grants FA9550-20-1-0358 and FA9550-22-1-0004, and partial support by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate (NDSEG) Fellowship Program. Charbel Farhat acknowledges partial support by the Air Force Office of Scientific Research under grants FA9550-17-1-0182 and FA9550-22-1-0004. This document however does not necessarily reflect the position of any of these institutions and therefore no official endorsement should be inferred.

References

[1] S. Chaturantabut, D. C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM Journal on Scientific Computing 32 (5) (2010) 2737–2764.

[2] D. Amsallem, M. J. Zahr, C. Farhat, Nonlinear model order reduction based on local reduced-order bases, International Journal for Numerical Methods in Engineering 92 (2012) 891–916. doi:10.1002/nme.4371.
URL `http://doi.wiley.com/10.1002/nme.4371`

[3] A. Da Ronch, K. Badcock, Y. Wang, A. Wynn, R. Palacios, Nonlinear model reduction for flexible aircraft control design, in: AIAA Atmospheric Flight Mechanics Conference, 2012, p. 4404.

[4] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares petrov-galerkin projection and compressive tensor approximations, International Journal for Numerical Methods in Engineering 86 (2011) 155–181. doi:10.1002/nme.3050.
URL http://doi.wiley.com/10.1002/nme.3050

[5] K. Carlberg, C. Farhat, J. Cortial, D. Amsallem, The gnat method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows, Journal of Computational Physics 242 (2013) 623–647. doi:10.1016/j.jcp.2013.02.028.
URL http://linkinghub.elsevier.com/retrieve/pii/S0021999113001472

[6] C. Farhat, P. Avery, T. Chapman, J. Cortial, Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency, International Journal for Numerical Methods in Engineering 98 (2014) 625–662. doi:10.1002/nme.4668.
URL http://doi.wiley.com/10.1002/nme.4668

[7] D. Osipov, K. Sun, Adaptive nonlinear model reduction for fast power system simulation, IEEE Transactions on Power Systems 33 (6) (2018) 6746–6754.

[8] S. Grimberg, C. Farhat, N. Youkilis, On the stability of projection-based model order reduction for convection-dominated laminar and turbulent flows, Journal of Computational Physics 419 (2020) 109681. doi:10.1016/J.JCP.2020.109681.

[9] S. Grimberg, C. Farhat, R. Tezaur, C. Bou-Mosleh, Mesh sampling and weighting for the hyperreduction of nonlinear petrov-galerkin reduced-order models with local reduced-order bases, International Journal for Numerical Methods in Engineering 122 (2021) 1846–1874. doi:10.1002/nme.6603.
URL https://onlinelibrary.wiley.com/doi/10.1002/nme.6603

[10] J. Barnett, C. Farhat, Quadratic approximation manifold for mitigating the kolmogorov barrier in nonlinear projection-based model order reduction, Journal of Computational Physics (2022) 111348.

[11] T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza, Model order reduction in fluid dynamics: challenges and perspectives, Reduced Order Methods for modeling and computational reduction (2014) 235–273.

[12] A. Iollo, D. Lombardi, Advection modes by optimal mass transfer, Physical Review E 89 (2014) 022923. doi:10.1103/PhysRevE.89.022923.
URL https://journals.aps.org/pre/abstract/10.1103/PhysRevE.89.022923

[13] F. Bernard, A. Iollo, S. Riffaud, Reduced-order model for the bgk equation based on pod and optimal transport, Journal of Computational Physics 373 (2018) 545–570. doi:10.1016/J.JCP.2018.07.001.

[14] N. Cagniart, Y. Maday, B. Stamm, Model order reduction for problems with large convection effects, in: Contributions to partial differential equations and applications, Springer, 2019, pp. 131–150.

[15] N. J. Nair, M. Balajewicz, Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent shocks, International Journal for Numerical Methods in Engineering 117 (12) (2019) 1234–1262.

[16] F. Black, P. Schulze, B. Unger, Projection-based model reduction with dynamically transformed modes, ESAIM: Mathematical Modelling and Numerical Analysis 54 (6) (2020) 2011–2043.

[17] W. He, P. Avery, C. Farhat, In situ adaptive reduction of nonlinear multiscale structural dynamics models, International Journal for Numerical Methods in Engineering 121 (22) (2020) 4971–4988.

[18] T. Taddei, A registration method for model order reduction: data compression and geometry reduction, SIAM Journal on Scientific Computing 42 (2) (2020) A997–A1027.

[19] A. Pinkus, n-Widths in Approximation Theory, Vol. 7, Springer Berlin Heidelberg, 1985. doi:10.1007/978-3-642-69894-1.

[20] Y. Maday, E. M. Ronquist, The reduced basis element method: Application to a thermal fin problem, http://dx.doi.org/10.1137/S1064827502419932 26 (2006) 240–258. doi:10.1137/S1064827502419932.
URL http://www.siam.org/journals/sisc/26-1/41993.html

[21] L. Iapichino, A. Quarteroni, G. Rozza, Reduced basis method and domain decomposition for elliptic problems in networks and complex parametrized geometries, Computers & Mathematics with Applications 71 (2016) 408–430. doi:10.1016/J.CAMWA.2015.12.001.

[22] C. Hoang, Y. Choi, K. Carlberg, Domain-decomposition least-squares petrov–galerkin (dd-lspg) nonlinear model reduction, Computer Methods in Applied Mechanics and Engineering 384 (2021) 113997. doi:10.1016/J.CMA.2021.113997.

[23] M. Buffoni, H. Telib, A. Iollo, Iterative methods for model reduction by domain decomposition, Computers & Fluids 38 (2009) 1160–1167. doi:10.1016/J.COMPFLUID.2008.11.008.

[24] J. Baiges, R. Codina, S. Idelsohn, A domain decomposition strategy for reduced order models. application to the incompressible navier–stokes equations, Computer Methods in Applied Mechanics and Engineering 267 (2013) 23–42. doi:10.1016/J.CMA.2013.08.001.

[25] A. Corigliano, M. Dossi, S. Mariani, Model order reduction and domain decomposition strategies for the solution of the dynamic elastic–plastic structural problem, Computer Methods in Applied Mechanics and Engineering 290 (2015) 127–155. doi:10.1016/J.CMA.2015.02.021.

[26] S. Riffaud, M. Bergmann, C. Farhat, S. Grimberg, A. Iollo, The dgdd method for reduced-order modeling of conservation laws, Journal of Computational Physics 437 (2021) 110336. doi:10.1016/J.JCP.2021.110336.

[27] K. Carlberg, M. Barone, H. Antil, Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction, Journal of Computational Physics 330 (2017) 693–734. doi:10.1016/J.JCP.2016.10.033.

[28] A. C. Antoulas, Approximation of large-scale dynamical systems, Approximation of Large-Scale Dynamical Systems (1 2005). doi:10.1137/1.9780898718713.
URL https://epubs.siam.org/page/terms

[29] D. Amsallem, C. Farhat, Stabilization of projection-based reduced-order models, International Journal for Numerical Methods in Engineering 91 (2012) 358–377. doi:10.1002/nme.4274.

[30] L. Sirovich, Turbulence and the dynamics of coherent structures. i. coherent structures, Quarterly of Applied Mathematics 45 (1987) 561–571. doi:10.1090/QAM/910462.
URL https://www.ams.org/qam/1987-45-03/S0033-569X-1987-0910462-6/

[31] G. Berkooz, , P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, http://dx.doi.org/10.1146/annurev.fl.25.010193.002543 25 (2003) 539–575. doi:10.1146/ANNUREV.FL.25.010193.002543.
URL https://www.annualreviews.org/doi/abs/10.1146/annurev.fl.25.010193.002543

[32] C. Eckart, G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1936 1:3 1 (1936) 211–218. doi:10.1007/BF02288367.
URL https://link.springer.com/article/10.1007/BF02288367

[33] D. C.-L. Fong, M. Saunders, Lsmr: An iterative algorithm for sparse least-squares problems, SIAM Journal on Scientific Computing 33 (5) (2011) 2950–2971.

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

[35] P. Geuzaine, G. Brown, C. Harris, C. Farhat, Aeroelastic dynamic analysis of a full f-16 configuration for various flight conditions, AIAA journal 41 (3) (2003) 363–371.

[36] C. Farhat, P. Geuzaine, G. Brown, Application of a three-field nonlinear fluid–structure formulation to the prediction of the aeroelastic parameters of an f-16 fighter, Computers & Fluids 32 (1) (2003) 3–29.

[37] P. R. Spalart, S. R. Allmaras, One-equation turbulence model for aerodynamic flows, Recherche aerospatiale (1994) 5–21doi:10.2514/6.1992-439.

[38] M. Strelets, Detached eddy simulation of massively separated flows, American Institute of Aeronautics and Astronautics Inc., 2001. doi:10.2514/6.2001-879.

[39] X.-C. Cai, C. Farhat, M. Sarkis, A minimum overlap restricted additive schwarz preconditioner and applications in 3d flow simulations, Contemporary Mathematics 218 (1998).

[40] J. L. Steger, R. Warming, Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods, Journal of computational physics 40 (2) (1981) 263–293.

[41] S. Ahmed, G. Ramm, G. Faltin, Some salient features of the time-averaged ground vehicle wake, in: SAE International Congress and Exposition, SAE International, 1984. doi:https://doi.org/10.4271/840300.
URL `https://doi.org/10.4271/840300`