Nonlinear Model Reduction for CFD Problems Using Local Reduced Order Bases

Kyle Washabaugh[*] David Amsallem[†] Matthew Zahr[‡] and Charbel Farhat[§]
Stanford University, Stanford, CA, 94305-3035, USA

A model reduction framework based on the concept of local reduced-order bases is presented. The offline phase of the method builds the local reduced-order bases using an unsupervised learning algorithm. In the online phase of the method, the choice of the local basis is based on the current state of the system. Inexpensive rank-one updates to the local bases are performed during the online phase for increased accuracy. Applications to nonlinear CFD simulations show that the method is effective in producing small and accurate reduced order models.

I. Introduction

Although computational fluid dynamics (CFD) models are typically high-dimensional, the trajectories of these models are often confined to low-dimensional affine subspaces. For this reason, CFD models are an ideal candidate for Model Order Reduction (MOR) methods.

In most MOR methods, the dimension of the system is reduced by projecting the equations of the high-dimensional model (HDM) onto the low-dimensional subspace spanned by its trajectories. Using this approach, it is possible to greatly reduce the number of degrees of freedom in the model while retaining the accuracy of the original HDM.[1–6]

Typically, this low-order projection subspace is represented by a reduced-order basis (ROB), and the state of the HDM is formed as a linear combination of these basis vectors. The resulting reduced-order model (ROM) can produce accurate responses within the projection subspace; however, a fundamental trade-off is that the ROM will never be able to explore portions of the state-space outside this subspace. This means that the performance of the ROM is ultimately decided by the quality of the ROB.

When a single ROB is used to reduce the dimension of an HDM, it must capture the dynamics of the HDM along the entire trajectory of interest. This can result in a very large ROB, and thus an inefficient reduced order model – an issue that is especially troublesome for design applications, where the ROB needs to capture the dynamics of the HDM at multiple design points.

The MOR method introduced in [7] alleviates this problem through the use of multiple *local* ROBs. In this approach, a local ROB is selected at each time step of the ROM simulation based on the current state of the system. Ideally, each local ROB captures only the local dynamics at a given point of the state-space, resulting in small and accurate models. Such a concept is particularly well-suited for the proper orthogonal decomposition (POD) method in which the ROB is built from snapshots of the system taken at various locations of the state-space.

This local ROB approach takes advantage of the fact that the state-space of a nonlinear dynamical system can often be partitioned into distinct characteristic regimes. In the context of CFD simulations, these state-space partitions could distinguish, for example, between solutions that are dominated by laminar vs. turbulent flow, subsonic vs. supersonic flow, or transient vs. limit-cycle behavior. As the state of the

[*]Graduate Student, Department of Aeronautics and Astronautics, William F. Durand Building, Room 028, Stanford University, Stanford, CA 94305-3035; AIAA Member
[†]Engineering Research Associate, Department of Aeronautics and Astronautics, William F. Durand Building, Room 028A, Stanford University, Stanford, CA 94305-3035; AIAA Member
[‡]Graduate Student, Institute of Computational and Mathematical Engineering, William F. Durand Building, Room 028, Stanford University, Stanford, CA 94305-3035
[§]Vivian Church Hoff Professor of Aircraft Structures, Department of Aeronautics and Astronautics, William F. Durand Building, Room 257, Stanford University, Stanford, CA 94305-3035; AIAA Fellow

system transitions from one regime to another during a ROM simulation, an appropriate local ROB can be chosen that best captures the physics of the current system, and omits information that is not immediately relevant.

In this work, the local ROB method originally presented in [7] is extended to a three-dimensional CFD problem, and the details of constructing accurate local ROBs are discussed at length. To this effect, this paper is organized as follows: The underlying principles of the local ROB method are presented in Section II, the details of the offline and online portions of the method are discussed in Sections III and IV, and applications to two nonlinear CFD systems are presented in Section V.

II. Model Reduction Based on Local Reduced-Order Bases

Consider a set of nonlinear Ordinary Differential Equations (ODEs) arising, for instance, from the discretization in space of a space-time Partial Differential Equation (PDE):

$$\begin{aligned} \frac{d\mathbf{w}(t)}{dt} &= \mathbf{f}(\mathbf{w}(t), t, \boldsymbol{\mu}) \\ \mathbf{w}(0) &= \mathbf{w}_0, \end{aligned} \tag{1}$$

where $t \geq 0$ denotes time, $\mathbf{w}(t) \in \mathbb{R}^n$ denotes the fluid state vector of dimension n, $\boldsymbol{\mu} \in \mathbb{R}^d$ denotes a vector of parameters defining the operating point of the system of interest, and $\mathbf{f} : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^n$ is the nonlinear flux function containing the semi-discrete counterparts of the convective and diffusive fluxes.

When Eq. (1) is solved by an implicit time-integrator, the state $\mathbf{w}^{(i)}$ at time $t^{(i)}, 0 \leq i \leq N_t$, can be computed as the solution to a system of discrete nonlinear equations, which are represented here as a nonlinear residual

$$\mathbf{r}^{(i)}(\mathbf{w}^{(i)}, \boldsymbol{\mu}) = \mathbf{0}. \tag{2}$$

When an iterative procedure such as the Newton-Raphson method is used to solve this nonlinear system, $\mathbf{w}^{(i,l)}$ denotes the computed solution at the l-th iteration of the i-th time step, $t^{(i)}$.

The main assumption made in reduced-order modeling is that the state solution $\mathbf{w}$ belongs to an affine subspace of $\mathbb{R}^n$, the dimension k of that subspace being typically orders of magnitude smaller than n. When a single ROB $\mathbf{V} \in \mathbb{R}^{n \times k}$ is used to reduce the system, this one ROB must capture all of the relevant physics of the HDM (2). For a complex system this can result in a very large basis, and thus a slow ROM.

As suggested in [7], this shortcoming of global ROBs can be overcome by selecting an appropriate *local* ROB at each time step of the ROM simulation. Since each local ROB needs only to capture a subset of the physics exhibited by the HDM, this allows the use of smaller ROBs for a given accuracy.

At a given time iteration i, it is therefore proposed to search for a solution $\mathbf{w}^{(i)}$ under the form

$$\mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} + \mathbf{V}(\mathbf{w}^{(i-1)}) \Delta \mathbf{w}^{(i)}_{k(\mathbf{w}^{(i-1)})} \tag{3}$$

where $\mathbf{V}(\mathbf{w}^{(i-1)}) \in \mathbb{R}^{n \times k(\mathbf{w}^{(i-1)})}$ denotes a local ROB that is chosen based on the state $\mathbf{w}^{(i-1)}$, and $\Delta \mathbf{w}^{(i)}_{k(\mathbf{w}^{(i-1)})}$ denotes the vector of unknowns that contains the increments of the generalized coordinates in the basis $\mathbf{V}(\mathbf{w}^{(i-1)})$. The notation $k(\mathbf{w}^{(i-1)})$ emphasizes that each local basis may have a different size.

In practice, a set of $N_{\mathbf{V}}$ local reduced-order bases is pre-computed and each basis $\mathbf{V}(\mathbf{w}^{(i-1)})$ is chosen within this set. In the remainder of this paper, $\{\mathcal{I}_j\}_{j=1}^{N_{\mathbf{V}}}$ denotes the set of indices $i \in \{1, \cdots, N_t\}$ such that $\mathbf{V}(\mathbf{w}^{(i-1)}) = \mathbf{V}_j$ and $k(\mathbf{w}^{(i-1)}) = k_j$. Using this notation, if $i \in \mathcal{I}_j$, Eq. (3) can be rewritten as

$$\mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} + \mathbf{V}_j \Delta \mathbf{w}^{(i)}_{k_j}. \tag{4}$$

Substituting Eq. (4) into the nonlinear system Eq. (2) results in the following system of n nonlinear equations in terms of k_j variables $\Delta \mathbf{w}^{(i)}_{k_j}$,

$$\mathbf{r}^{(i)}(\mathbf{w}^{(i-1)} + \mathbf{V}_j \Delta \mathbf{w}^{(i)}_{k_j}, \boldsymbol{\mu}) = \mathbf{0}. \tag{5}$$

Once an appropriate local ROB has been chosen, the above system of nonlinear equations can be solved in a least squares sense by considering the following problem,

$$\min_{\Delta \mathbf{w}_{k_j}^{(i)} \in \mathbb{R}^{k_j}} \left\| \mathbf{r}^{(i)}\left(\mathbf{w}^{(i-1)} + \mathbf{V}_j \Delta \mathbf{w}_{k_j}^{(i)}, \boldsymbol{\mu}\right) \right\|_2 . \tag{6}$$

The next two sections of this paper are concerned with the details of solving Eq. (6); Section III discusses the offline construction of the local ROBs, $\{\mathbf{V}_j\}_{j=1}^{N_\mathbf{V}}$, and Section IV discusses the online solution to Eq. (6) using an iterative Gauss-Newton procedure.[6,8]

III. Offline Phase: Construction of the Local ROB Database

In practice, the set of reduced bases $\{\mathbf{V}_j\}_{j=1}^{N_\mathbf{V}}$ is built using a POD algorithm based on the method of snapshots.[9] These snapshots $\{\mathbf{y}^s\}_{s=1}^{N_{\text{snaps}}}$ are typically based on pre-computed states $\{\mathbf{w}^s\}_{s=1}^{N_{\text{snaps}}}$ of the HDM (2), where $N_{\text{snaps}} \leq N_t$.

The proposed method for constructing a database of local ROBs consists of three steps. First, the states $\{\mathbf{w}^s\}_{s=1}^{N_{\text{snaps}}}$ are clustered into $N_\mathbf{V}$ subsets using an unsupervised learning algorithm, such as the k-means algorithm. Second, the clusters are made to overlap with one another by sharing a small number of states between neighboring clusters. Third, the snapshots $\{\mathbf{y}^s\}_{s=1}^{N_{\text{snaps}}}$ are formed from the state clusters and the individual local ROBs are computed. These three steps are detailed below.

A. Clustering of Pre-Computed Solutions

In this work, the k-means algorithm is used to partition the pre-computed states $\{\mathbf{w}^s\}_{s=1}^{N_{\text{snaps}}} \subset \{\mathbf{w}^{(i)}\}_{i=0}^{N_t}$ into $N_\mathbf{V}$ clusters.[7] This choice of algorithm is shown to work well for the applications in Section V; however, any other clustering algorithm could be substituted here with only minor changes to the method.

More important than the choice of algorithm is the choice of distance metric used to compare states. The choice of distance metric is fundamental to the local ROM method, and affects both the offline and online results. In this work, the distance metric used for both the partitioning of the pre-computed states as well as the online choice of ROB is chosen as

$$d\left(\mathbf{w}^i, \mathbf{w}^j\right) = \left\| \mathbf{w}^i - \mathbf{w}^j \right\|_2 . \tag{7}$$

REMARK: This definition of distance has been shown to work well for a simple one dimensional fluid problem and a fluid-structure-electric interaction problem;[7] however, it may not be the best choice for all problems. For systems that exhibit strongly periodic behavior, it may be more appropriate to choose a distance metric that relies on the frequency content of the states, or possibly an output quantity of interest, such as lift or drag for aeronautical applications.

The process of clustering the pre-computed states is summarized in Algorithm 1. To illustrate the algorithm, a set of states as well the corresponding clusters and state-space partitioning are depicted in Fig. 1 (a)-(b).

B. Overlap of Snapshot Clusters

As suggested in [7], cluster overlap can be introduced by adding near-by states to each cluster. This ensures that there are no "gaps" between clusters, reducing the error when the state trajectory transitions from one cluster to another. In this paper, a new method is proposed for introducing overlap.

The proposed method is a two-step approach: the connectivity of the clusters is first established, and a specified number of snapshots are then shared between neighboring clusters. In this section, it is assumed that the clusters have been generated with a k-means algorithm; substituting a different clustering algorithm would require slight modifications to the approach presented below.

Here, two clusters are said to be connected if their respective centroids are the two closest to any of the pre-computed states $\{\mathbf{w}^s\}_{s=1}^{N_{\text{snaps}}}$. With this definition, the cluster connectivity can be established by iterating through the pre-computed states, identifying the two closest cluster centroids to each, and marking the corresponding pair of clusters as neighbors. This corresponds to lines 1 to 3 in Algorithm 2, and is depicted in Fig. 1 (c), where states are colored according to the second nearest cluster center.

Algorithm 1 Clustering of pre-computed states into $N_{\mathbf{V}}$ subsets

Input: Number of desired clusters, $N_{\mathbf{V}} \geq 1$

Output: State vector clusters $\{\mathcal{W}_j\}_{j=1}^{N_{\mathbf{V}}}$; centroids of state vector clusters $\{\overline{\mathbf{w}}_j\}_{j=1}^{N_{\mathbf{V}}}$

1: Compute the full-order solutions $\{\mathbf{w}^{(i)}\}_{i=0}^{N_t}$.

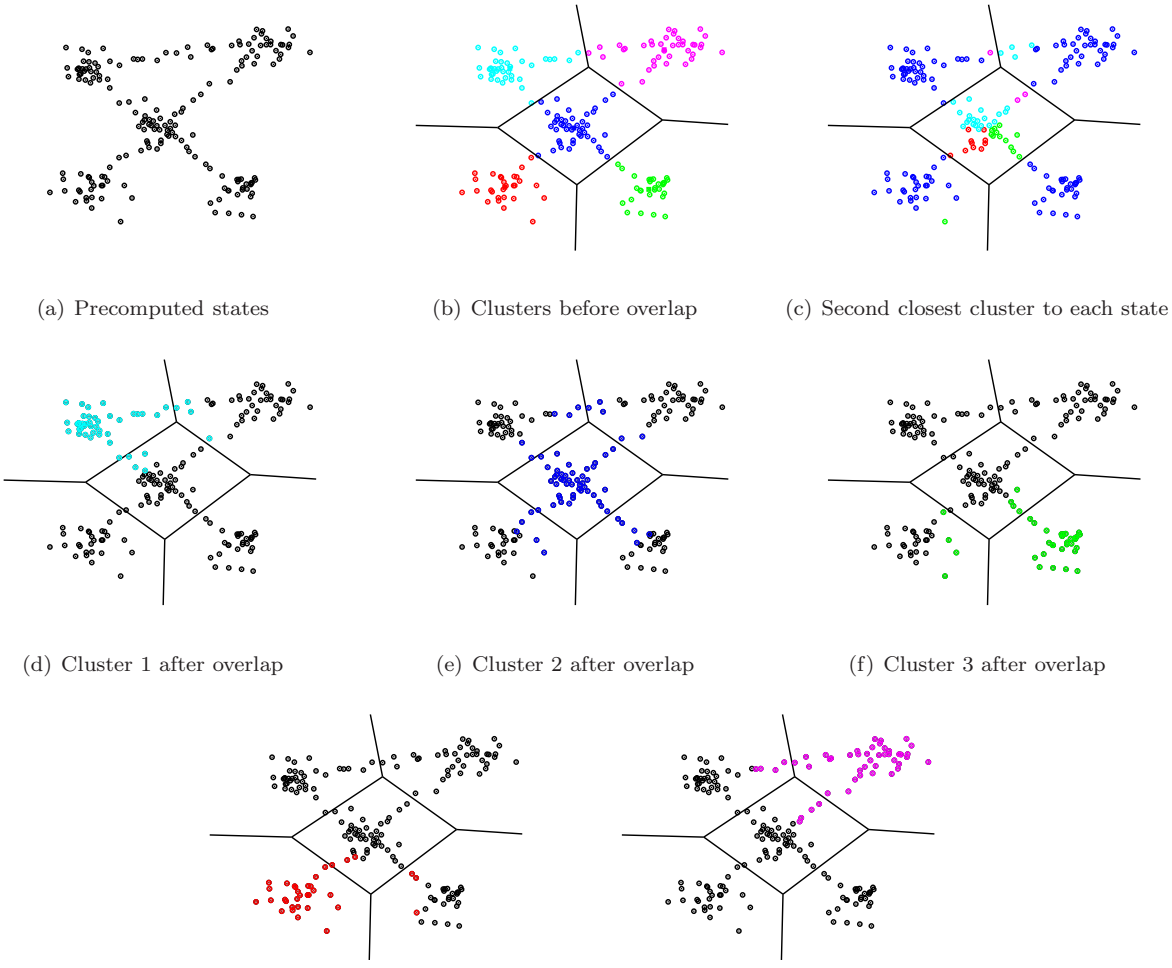
2: Select $N_{\text{snaps}} \geq N_{\mathbf{V}}$ snapshots $\{\mathbf{w}^s\}_{s=1}^{N_{\text{snaps}}} \subset \{\mathbf{w}^{(i)}\}_{i=0}^{N_t}$.

3: Store the state vectors in a matrix $\mathbf{W} = \left[\mathbf{w}^1, \cdots, \mathbf{w}^{N_{\text{snaps}}}\right]$.

4: Partition the state vectors into $N_{\mathbf{V}}$ clusters $\{\mathcal{W}_j\}_{j=1}^{N_{\mathbf{V}}}$ based on the chosen distance metric. Then

$$\bigcup_{j=1}^{N_{\mathbf{V}}} \mathcal{W}_j = \left\{\mathbf{w}^1, \cdots, \mathbf{w}^{N_{\text{snaps}}}\right\} \tag{8}$$

5: Compute the centroid $\overline{\mathbf{w}}_j$ of each cluster $\{\mathcal{W}_j\}_{j=1}^{N_{\mathbf{V}}}$.



(a) Precomputed states (b) Clusters before overlap (c) Second closest cluster to each state

(d) Cluster 1 after overlap (e) Cluster 2 after overlap (f) Cluster 3 after overlap

(g) Cluster 4 after overlap (h) Cluster 5 after overlap

Figure 1. Snapshot clustering procedure, $N_{\mathbf{V}} = 5$

The advantage of this method is that any two clusters with states near their common boundary are likely to be identified as neighbors. Consequently, if the trajectory of the training simulation crosses a cluster boundary, it is likely that the two clusters sharing this boundary will be identified as neighbors. In practice, this method is somewhat conservative; two clusters that are near one another in state space but unrelated in the training trajectory are not always identified as neighbors. This proves to be a desirable feature when attempting to reproduce the training trajectory with the ROM, but it may be necessary to share states from all adjacent clusters for parametric applications.

After the cluster connectivity has been established, the overlap algorithm then iterates through the clusters and identifies which states should be added from each neighboring cluster. In this work, each cluster is enlarged by a set percentage (typically $r = 10\%$) by adding an equal number of states from each of its neighbors. The algorithm shares the states that are nearest to the boundary between the cluster and its neighbor. This step corresponds to lines 4 to 13 in Algorithm 2 and its result is depicted in Fig. 1 (d)-(h).

Note that the cluster centers are not updated when overlap is introduced. The cluster centers define the partitioning of the state-space, and the intent of adding overlap is to improve the performance of the local ROBs near the boundaries of each cluster, not to alter the partitioning of the state-space.

Algorithm 2 Introduce overlap into state clusters

Input: State clusters $\{\mathcal{W}_j\}_{j=1}^{N_\mathbf{V}}$; state cluster centroids $\{\overline{\mathbf{w}}_j\}_{j=1}^{N_\mathbf{V}}$; threshold r
Output: Overlapping state clusters $\{\widehat{\mathcal{W}}_j\}_{j=1}^{N_\mathbf{V}}$

1: **for** $s = 1, \cdots, N_{\text{snaps}}$ **do**
2: Identify the two nearest clusters centroids to state $\mathbf{w}^s$ and mark these clusters as neighbors
3: **end for**
4: **for** $j = 1, \cdots, N_\mathbf{V}$ **do**
5: Let $N_{\text{snaps},j}$ denote the number of states in cluster j, and $N_{\text{neighbors},j}$ the number of its neighboring clusters
6: $\widehat{\mathcal{W}}_j = \mathcal{W}_j$
7: Calculate the number $N_{\text{add},j}$ of states to add from each neighbor of $\mathcal{W}_j$ as $N_{\text{add},j} = \text{ceil}\left(\frac{N_{\text{snaps},j} \times r}{N_{\text{neighbors},j}}\right)$
8: **for** $i = 1, \cdots, N_\mathbf{V}$ **do**
9: **if** i is a neighbor to j **then**
10: Find the $N_{\text{add},j}$ states nearest to their common boundary and add these to $\widehat{\mathcal{W}}_j$
11: **end if**
12: **end for**
13: **end for**

C. Local ROB Construction

After forming the overlapping state clusters $\{\overline{\mathbf{w}}_j\}_{j=1}^{N_\mathbf{V}}$, the set of reduced bases $\{\mathbf{V}_j\}_{j=1}^{N_\mathbf{V}}$ is built using a POD algorithm based on the method of snapshots.[9] These snapshots $\mathcal{Y}_j = \{\mathbf{y}_j^s\}_{s=1}^{N_{\text{snaps},j}}$ are typically formed by subtracting reference states $\{\mathbf{w}_{\text{ref},j}\}_{j=1}^{N_\mathbf{V}}$ from the raw states $\widehat{\mathcal{W}}_j = \{\mathbf{w}_j^s\}_{s=1}^{N_{\text{snaps},j}}$. The offline procedure for constructing a database of local ROBs by POD is summarized in Algorithm 3.

In practice, the choice of the reference states $\{\mathbf{w}_{\text{ref},j}\}_{j=1}^{N_\mathbf{V}}$ has an impact on the performance of the resulting ROBs. In the following discussion, it is argued that a particular choice of reference condition is the most appropriate for local ROBs.

To gain insight into this matter of snapshot reference conditions for offline ROB construction, consider the subsequent online ROM simulation using the pre-computed local ROBs. At any time iteration i before the ROM switches from the first local ROB $\mathbf{V}_1$ to the second local ROB $\mathbf{V}_2$, $\mathbf{w}^{(i)}$ can be expressed in terms of the initial solution, $\mathbf{w}^{(0)}$, as follows

$$\mathbf{w}^{(i)} = \mathbf{w}^{(0)} + \mathbf{V}_1 \sum_{p=1}^{i} \Delta \mathbf{w}_{k_1}^{(p)}. \tag{10}$$

Algorithm 3 Construction of a set of local reduced bases

Input: Number $N_{\mathbf{V}} \geq 1$ of pre-computed bases, corresponding sets of state clusters $\{\widehat{\mathcal{W}}_j\}_{j=1}^{N_{\mathbf{V}}}$, reference states $\{\mathbf{w}_{\text{ref},j}\}_{j=1}^{N_{\mathbf{V}}}$

Output: Local POD bases $\{\mathbf{V}_j\}_{j=1}^{N_{\mathbf{V}}}$, corresponding singular values and right vectors $\{\mathbf{\Sigma}_j\}_{j=1}^{N_{\mathbf{V}}}$ and $\{\mathbf{Z}_j\}_{j=1}^{N_{\mathbf{V}}}$

1: **for** $j = 1, \cdots, N_{\mathbf{V}}$ **do**

2: Subtract the reference state $\mathbf{w}_{\text{ref},j}$ from the states $\widehat{\mathcal{W}}_j = \{\mathbf{w}_j^s\}_{s=1}^{N_{\text{snaps},j}}$ to form the snapshots $\mathcal{Y}_j = \{\mathbf{y}_j^s\}_{s=1}^{N_{\text{snaps},j}}$.

3: Store the snapshots $\mathcal{Y}_j$ in a matrix $\mathbf{Y}_j$.

4: Compute a singular value decomposition

$$\mathbf{Y}_j = \mathbf{U}\mathbf{\Sigma}\mathbf{Z}^T. \tag{9}$$

5: Choose a dimension $k_j \leq N_{\text{snaps},j}$ for the j-th reduced basis

6: Truncate the first k_j left components to obtain the reduced-order basis $\mathbf{V}_j = \mathbf{U}(:, 1:k_j)$, singular values $\mathbf{\Sigma}_j = \mathbf{\Sigma}(1:k_j, 1:k_j)$, and right singular vectors $\mathbf{Z}_j = \mathbf{Z}(:, 1:k_j)$, such that $\mathbf{Y}_j \approx \mathbf{V}_j\mathbf{\Sigma}_j\mathbf{Z}_j^T$.

7: **end for**

For the purposes of this discussion, it is helpful to slightly rearrange Eq. (10) as

$$\mathbf{w}^{(i)} - \mathbf{w}^{(0)} = \mathbf{V}_1 \sum_{p=1}^{i} \Delta\mathbf{w}_{k_1}^{(p)}. \tag{11}$$

Examining Eq. (11), the quantities $\mathbf{w}^{(i)} - \mathbf{w}^{(0)}$ are constrained to lie in the subspace spanned by $\mathbf{V}_1$. It follows that, for this ROM to be accurate, $\mathbf{V}_1$ must be a good approximation of the subspace spanned by the corresponding iterates of the HDM. If the basis $\mathbf{V}_1$ is formed using Algorithm 3 without truncation, that is $k_j = N_{\text{snaps},j}$, then there are several choices of snapshot reference condition that could be appropriate; however, when truncation is present, it is postulated that $\mathbf{V}_1$ will best approximate this subspace if constructed from snapshots of the type $\mathbf{y}_1^s = \mathbf{w}_1^s - \mathbf{w}^{(0)}$. In practice, it has been observed[10,11] that, for global ROMs, this choice does indeed lead to more accurate reduced models than the common choice $\mathbf{y}_1^s = \mathbf{w}_1^s$ or the choice $\mathbf{y}_1^s = \mathbf{w}_1^s - \mathbf{w}_1^{s-1}$ presented in [6].

Consider now a later time step i from this same online local ROM simulation after the second local ROB $\mathbf{V}_2$ has been selected. Here, m denotes the iteration at which this second local ROB was first selected. At this time step, the counterpart to Eq. (10) is then

$$\mathbf{w}^{(i)} = \mathbf{w}^{(0)} + \mathbf{V}_1 \sum_{1 \leq p \leq m-1} \Delta\mathbf{w}_{k_1}^{(p)} + \mathbf{V}_2 \sum_{m \leq p \leq i} \Delta\mathbf{w}_{k_2}^{(p)}. \tag{12}$$

Since $\mathbf{w}^{(m-1)} = \mathbf{w}^{(0)} + \mathbf{V}_1 \sum_{1 \leq p \leq m-1} \Delta\mathbf{w}_{k_1}^{(p)}$, the counterpart of Eq. (11) is then

$$\mathbf{w}^{(i)} - \mathbf{w}^{(m-1)} = \mathbf{V}_2 \sum_{m \leq p \leq i} \Delta\mathbf{w}_{k_2}^{(p)}. \tag{13}$$

Appealing to the same logic as before, it appears that the second local ROB $\mathbf{V}_2$ should be constructed from snapshots of the type $\mathbf{y}_2^s = \mathbf{w}_2^s - \mathbf{w}^{(m-1)}$.

Generalizing this observation, it is expected that a given local ROB $\mathbf{V}_j$ will perform well online if constructed from snapshots of the type $\mathbf{y}_j^s = \mathbf{w}_j^s - \mathbf{w}_{\text{switch},j}$, where $\mathbf{w}_{\text{switch},j}$ denotes the state of the system when the ROM switches to basis $\mathbf{V}_j$. Unfortunately, this reference condition $\mathbf{w}_{\text{switch},j}$ is not known at the time of the ROB construction, and additionally, if a basis is used more than once by the simulation it would need to be referenced with two different states.

The novel approach proposed in this paper is to first construct a database of local ROBs offline using some reference states $\{\mathbf{w}_{\text{ref},j}\}_{j=1}^{N_{\mathbf{V}}}$, and to then perform inexpensive rank-one updates to the local ROBs online when $\mathbf{w}_{\text{switch},j}$ is known. In this paper, the initial condition of the online simulation is used as the reference state $\mathbf{w}_{\text{ref},j}$ for all clusters $j = 1, \cdots, N_{\mathbf{V}}$, ensuring that the first local ROB does not need to be updated

online; however, for parametric applications it could prove beneficial to use the centroids of the clusters as the reference state, since the cluster centers should be closer to $\mathbf{w}_{\text{switch},j}$ than the initial condition. This represents a topic for future investigation.

After constructing the local ROB database using Algorithm 3 with reference snapshots $\{\mathbf{w}_{\text{ref},j}\}_{j=1}^{N_{\mathbf{V}}}$,

$$\text{range}(\mathbf{V}_j) \subset \text{span}\left(\{\mathbf{w}_j^s - \mathbf{w}_{\text{ref},j}\}_{s=1}^{N_{\text{snaps},j}}\right) = \text{range}\left(\mathbf{Y}_j\right), \tag{14}$$

where the columns of $\mathbf{Y}_j$ contain the snapshots $\mathbf{w}_j^s - \mathbf{w}_{\text{ref},j}$. During the online ROM simulation, when a new basis $\mathbf{V}_j$ is selected, a local ROB update is then performed using the state $\mathbf{w}_{\text{switch},j}$. The goal of the proposed approach is to compute a modified local ROB $\widetilde{\mathbf{V}}_j$ such that

$$\text{range}(\widetilde{\mathbf{V}}_j) \subset \text{span}\left(\{\mathbf{w}_j^s - \mathbf{w}_{\text{switch},j}\}_{s=1}^{N_{\text{snaps},j}}\right) = \text{range}\left(\widetilde{\mathbf{Y}}_j\right), \tag{15}$$

where the columns of $\widetilde{\mathbf{Y}}_j$ contain the snapshots $\mathbf{w}_j^s - \mathbf{w}_{\text{switch},j}$. $\widetilde{\mathbf{Y}}_j$ and $\mathbf{Y}_j$ are therefore related by the identity

$$\widetilde{\mathbf{Y}}_j = \mathbf{Y}_j + \left(\mathbf{w}_{\text{ref},j} - \mathbf{w}_{\text{switch},j}\right)\mathbf{1}^T, \tag{16}$$

where ^{T} denotes the transpose operator and $\mathbf{1}$ is a vector of ones which has the dimension of the number of snapshots in $\mathbf{Y}_j$. Hence, $\widetilde{\mathbf{Y}}_j$ is a rank-one update matrix to $\mathbf{Y}_j$. Since the basis $\widetilde{\mathbf{V}}_j$ is constructed by a POD that involves a Singular Value Decomposition (SVD) of the matrix $\widetilde{\mathbf{Y}}_j$, the fast algorithm for updating an SVD developed in [12] is used for a fast update of the local ROB $\widetilde{\mathbf{V}}_j$. This is described in Algorithm 4.

Algorithm 4 Online Update to Local ROB $\mathbf{V}_j$, $j \in \{1, \cdots, N_{\mathbf{V}}\}$

Input: Original SVD $\mathbf{Y}_j \approx \mathbf{V}_j \mathbf{\Sigma}_j \mathbf{Z}_j^T$, $\mathbf{V}_j \in \mathbb{R}^{n \times k_j}$, $\mathbf{\Sigma}_j \in \mathbb{R}^{k_j \times k_j}$, $\mathbf{Q}_j = \mathbf{Z}_j^T \mathbf{1} \in \mathbb{R}^{k_j}$, $q_j = \|\mathbf{1} - \mathbf{Z}_j \mathbf{Q}_j\|_2$, original reference state $\mathbf{w}_{\text{ref},j}$, online state $\mathbf{w}_{\text{switch},j}$

Output: Updated local ROB $\widetilde{\mathbf{V}}_j$

1: Compute $\mathbf{a} = \mathbf{w}_{\text{ref},j} - \mathbf{w}_{\text{switch},j} \in \mathbb{R}^n$
2: Compute $\mathbf{m} = \mathbf{V}_j^T \mathbf{a} \in \mathbb{R}^{k_j}$
3: $\mathbf{p} = \mathbf{a} - \mathbf{V}_j \mathbf{m} \in \mathbb{R}^n$
4: $R_a = \|\mathbf{p}\|_2$
5: $\mathbf{p} = \mathbf{p}/R_a$
6: $\mathbf{K} = \begin{bmatrix} \mathbf{\Sigma}_j & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \begin{bmatrix} \mathbf{m} \\ R_a \end{bmatrix} \begin{bmatrix} \mathbf{Q}_j \\ q_j \end{bmatrix}^T \in \mathbb{R}^{(k_j+1)\times(k_j+1)}$
7: Compute the SVD of $\mathbf{K} = \mathbf{C}\,\mathbf{S}\,\mathbf{D}^T$, where $\mathbf{C}, \mathbf{S}, \mathbf{D} \in \mathbb{R}^{(k_j+1)\times(k_j+1)}$
8: Compute $\bar{\mathbf{V}}_j = \begin{bmatrix} \mathbf{V}_j & \mathbf{p} \end{bmatrix} \mathbf{C} \in \mathbb{R}^{n \times (k_j+1)}$
9: Let $\widetilde{\mathbf{V}}_j = \bar{\mathbf{V}}_j(:, 1 : k_j)$

REMARK 1: Algorithm 4 performs an exact SVD update on a low rank approximation to the snapshot matrix $\mathbf{Y}_j$. As a result, the modified basis $\widetilde{\mathbf{V}}_j$ is an approximation to the exact basis that would have been computed using the true snapshot matrix $\widetilde{\mathbf{Y}}_j$. The error introduced due to this approximation is studied for the one-dimensional Burgers' equation in Section V.A.

REMARK 2: One of the main benefits of using local ROBs is that the bases can be of varying sizes; however, picking each of these sizes by hand quickly becomes impractical as the number of bases grows. In this work, the local ROBs sizes are chosen based on the decay of the singular values, and are constrained to fall between an upper and lower bound. For the applications in this paper, this method yielded appropriately sized local ROBs without requiring significant assistance.

IV. Online Phase: Simulation of the Reduced-Order Model Using Local Bases

After constructing the local ROBs $\{\mathbf{V}_j\}_{j=1}^{N_{\mathbf{V}}}$ offline, these bases are used to solve a sequence of minimization problems of the form (6) during the online ROM simulation. At every time step, the appropriate ROB is determined by computing the distance of the current state to the cluster centers and selecting the local ROB that corresponds to the closest cluster center.

The approach proposed here is identical to the approach of [7], except that every time a new ROB is selected it is updated online using Algorithm 4. This procedure is summarized in Algorithm 5.

Algorithm 5 Solution of Eq. (6) by the Gauss-Newton procedure

Input: Previous solution $\mathbf{w}^{(i-1)}$, set of local ROBs $\{\mathbf{V}_j\}_{j=1}^{N_\mathrm{V}}$, previous ROB $\widetilde{\mathbf{V}}$

Outputs: Solution $\mathbf{w}^{(i)}$

1: Choose a local model j based on the distances between the cluster centroids and the current state, $\mathbf{w}^{(i-1)}$

2: If the ROB j differs from the one used at the previous iteration $i-1$, update the ROB $\mathbf{V}_j$ using Algorithm 4 and obtain the modified ROB $\widetilde{\mathbf{V}} = \widetilde{\mathbf{V}}_j$, otherwise use the modified ROB $\widetilde{\mathbf{V}}$ used at the previous iteration

3: Let $\mathbf{w}^{(i,0)} = \mathbf{w}^{(i-1)}$ and $\Delta\mathbf{w}_{k_j}^{(i,0)} = \mathbf{0}$

4: **for** $l = 0, \cdots,$ (until convergence) **do**

5: Compute $\mathbf{r}^{(i,l)} = \mathbf{r}^{(i)}(\mathbf{w}^{(i,l)}, \boldsymbol{\mu})$.

6: Evaluate $\mathbf{J}^{(i,l)} = \dfrac{\partial \mathbf{r}^{(i)}}{\partial \mathbf{w}}(\mathbf{w}^{(i,l)}, \boldsymbol{\mu})$

7: Compute $\mathbf{W}_j^{(i,l)} = \mathbf{J}^{(i,l)}\widetilde{\mathbf{V}}$

8: Compute the thin QR decomposition $\mathbf{W}_j^{(i,l)} = \mathbf{Q}^{(i,l)}\mathbf{R}^{(i,l)}$

9: Solve $\mathbf{R}^{(i,l)}\mathbf{p}^{(i,l)} = -\mathbf{Q}^{(i,l)T}\mathbf{r}^{(i,l)}$

10: Compute a step $\alpha^{(i,l)}$ by a line-search procedure (or use the Newton step and set $\alpha^{(i,l)} = 1$)

11: Update $\Delta\mathbf{w}_{k_j}^{(i,l+1)} = \Delta\mathbf{w}_{k_j}^{(i,l)} + \alpha^{(i,l)}\mathbf{p}^{(i,l)}$

12: Update $\mathbf{w}^{(i,l+1)} = \mathbf{w}^{(i,I)} + \widetilde{\mathbf{V}}\Delta\mathbf{w}_{k_j}^{(i,l+1)}$

13: **end for**

14: Let $\mathbf{w}^{(i)} = \mathbf{w}^{(i,l)}$

REMARK 1: Since the distances from the current state to each of the cluster centers are computed at every time step, it is important that these distance calculations be computationally efficient. It is shown in [7] that, using the distance metric defined in Eq. (7), it is possible to compute distances with a complexity that does not depend on the dimension n of the large-scale underlying model. Similarly, in the proposed approach with updated local ROBs, it can be shown that the same low complexity can be achieved.

REMARK 2: In Eq. (6), the number of unknowns has been reduced through the introduction of a local ROB for the state. In spite of this, the computational cost associated with solving this system of equations via Algorithm 5 is still comparable to the cost of solving the high-fidelity model as it requires the evaluation of the full-order residual and Jacobian $\mathbf{r}^{(i)}(\mathbf{w}^{(i,l)}, \boldsymbol{\mu})$ and $\mathbf{J}^{(i)}(\mathbf{w}^{(i,l)}, \boldsymbol{\mu})$ at each Newton iteration. It has been shown in [7] that the computational cost of Algorithm 5 can be greatly reduced using hyper-reduction, that is, by introducing an additional level of approximation. The local ROB update procedure outlined in Algorithm 4 in Section III.C requires full state vectors and therefore has a complexity that scales with the dimension n of the underlying HDM. As a result, it cannot be extended to the online phase of the local hyper-reduction as is. Future work will focus on adapting this approach to hyper-reduction.

V. Applications

A. One dimensional Burgers' Equation

1. *High-Dimensional Model*

The first system of interest originates from the inviscid Burgers' equation

$$\begin{aligned} &\frac{\partial W}{\partial t} + W\frac{\partial W}{\partial x} = g(x,t), \;\; x \in [0,1], \;\; 0 \le t, \\ &W(x,0) = 1, \;\; W(0,t) = \sqrt{5}, \end{aligned} \tag{17}$$

where x denotes the space variable, $W(x,t)$ is the quantity of interest, and $g(x,t) = 0.02e^{0.02x}$ is a source term. This initial-boundary-value-problem (IBVP) is discretized in space using Godunov's Finite Volume method, resulting in a set of nonlinear ODEs of dimension $n = 10000$. Solutions of the HDM at six different times are depicted in Fig. 2.

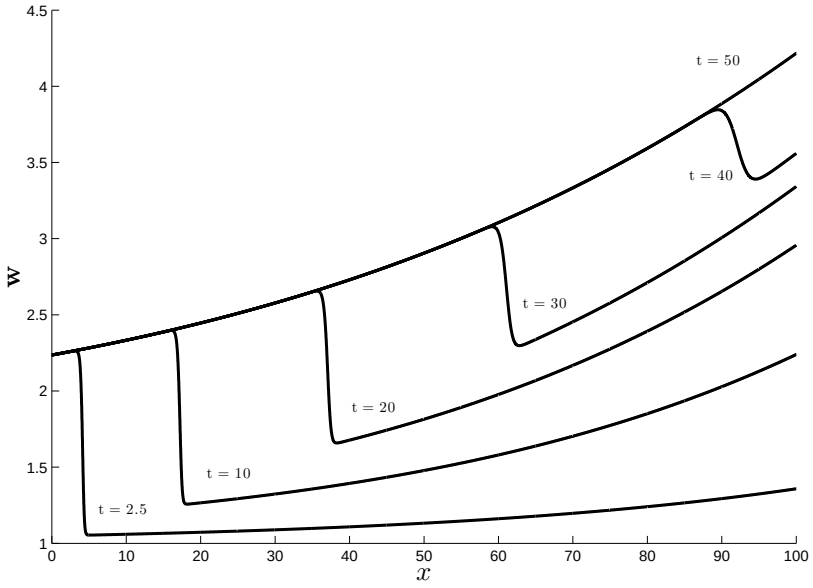


Figure 2. HDM solution of the inviscid Burgers' equation

2. *Clustering Results*

In this section, the particular case of $N_{\mathbf{V}} = 4$ has been selected to demonstrate the performance of the clustering and overlap algorithms. These results are representative of those obtained for other values of $N_{\mathbf{V}}$. In Fig. 3 (a), the $N_{\mathbf{V}} = 4$ state clusters are shown both before and after overlap is introduced using a threshold of ($r = 10\%$). Note that any two overlapping clusters are also linked in the training simulation.

The cluster centroids are depicted in Fig. 3 (b). This example demonstrates that the cluster centroids are not necessarily physical states of the system – here, the centroids are characterized by smooth, blended shocks that are clearly non-physical.

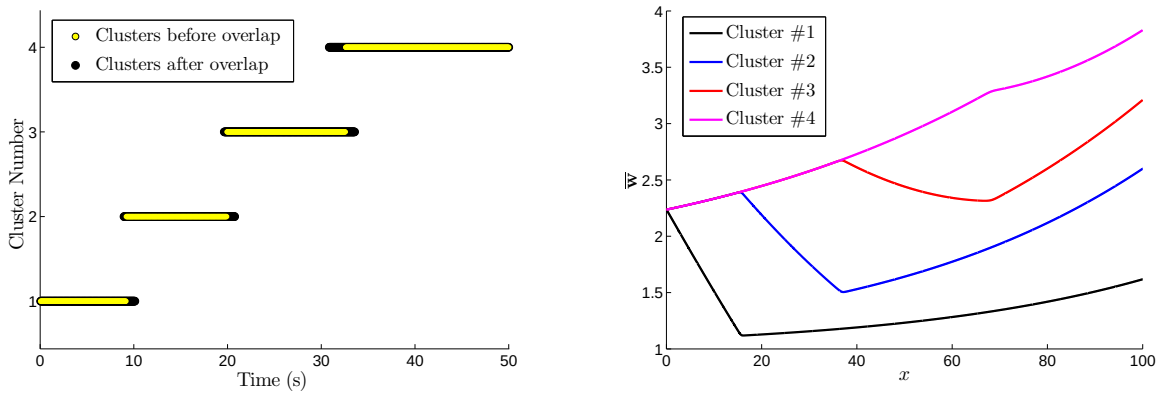
3. *ROM Performance*

The first set of numerical experiments examine the issue of snapshot reference condition discussed in Section III.C. For these experiments, local ROBs $\{\mathbf{V}_j\}_{j=1}^{N_{\mathbf{V}}}$ were constructed from the $N_{\mathbf{V}} = 4$ clusters shown in Fig. 3 using the following snapshot reference conditions,

1. $\mathbf{y}^s_{\text{init},j} = \mathbf{w}^s_j - \mathbf{w}^{(0)}$ (Initial state),
2. $\mathbf{y}^s_{\text{prev},j} = \mathbf{w}^s_j - \mathbf{w}^{s-1}_j$ (Previous state),
3. $\mathbf{y}^s_{\text{none},j} = \mathbf{w}^s_j$ (No reference),
4. $\mathbf{y}^s_{\text{switch},j} = \mathbf{w}^s_j - \mathbf{w}_{\text{switch},j}$ (State when ROM switches to $\mathbf{V}_j$).

For comparison, reference condition 4 was implemented using both the proposed online updates and an exact method. For the online updates, ROBs that were initially referenced with $\mathbf{w}^{(0)}$ were updated using $\mathbf{w}_{\text{switch},j}$ as outlined in Algorithm 4. For the exact method, the SVD was computed online using Algorithm 3 with $\mathbf{w}_{\text{ref},j} = \mathbf{w}_{\text{switch},j}$. Clearly, this exact approach is too computationally expensive to be realistic in an online framework, and it is reported here solely for assessing the accuracy of the proposed online update approach.

After constructing the local ROBs corresponding to the various snapshot choices, local ROM simulations were then performed. Error versus time for each of the snapshot choices is reported in Fig. 4. Examining



(a) Overlapping clusters

(b) Cluster centroids

Figure 3. Clustering results, $N_{\mathbf{V}} = 4$

these results, the snapshot methods $\mathbf{y}^s_{\text{prev},j}$ and $\mathbf{y}^s_{\text{none},j}$ both performed very poorly, with relative errors exceeding 10%. In fact, these two simulations were so inaccurate that the online ROM simulations did not even use all of the precomputed ROBs; the ROM constructed with $\mathbf{y}^s_{\text{prev},j}$ used three of the four local ROBs, and the ROM constructed with $\mathbf{y}^s_{\text{none},j}$ only used two. The ROM simulation using snapshot method $\mathbf{y}^s_{\text{init},j}$ performed better, with relative errors of 6%, but as expected, the relative error increased as the simulation progressed. In contrast, both ROMs constructed with $\mathbf{y}^s_{\text{switch},j}$ performed very well, with relative errors of less than 4%. This study supports the earlier claim that the local ROBs should be constructed using the snapshots $\mathbf{y}^s_{\text{switch},j}$, and indicates that the proposed online update algorithm closely approximates the exact ROBs built from these snapshots.

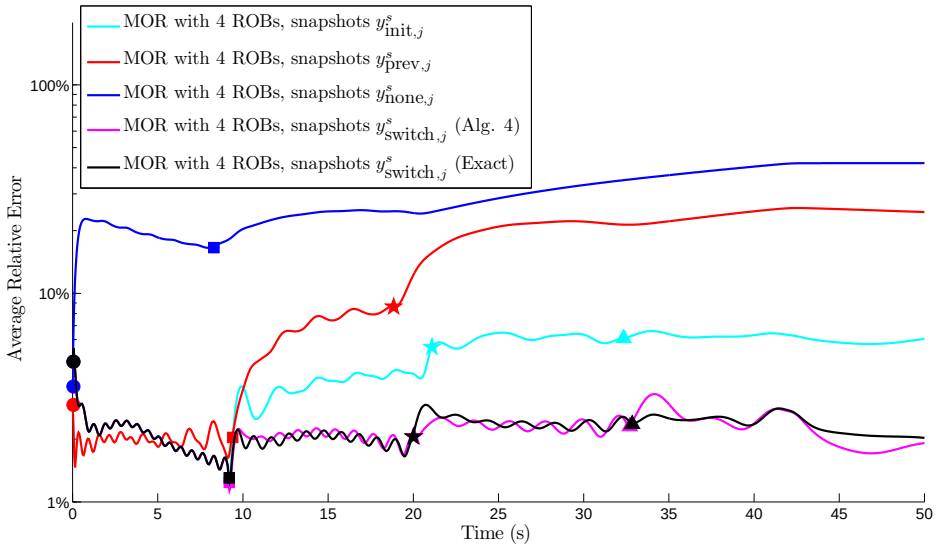


Figure 4. Performance of the nonlinear MOR method with $N_{\mathbf{V}} = 4$ and $k_j = \{16, 11, 7, 4\}$ as a function of snapshot method; time iterates corresponding to a local ROB switch indicated as ◯ for ROB #1, □ for ROB #2, ☆ for ROB #3, and △ for ROB #4

In Fig. 5, solutions computed with $\mathbf{y}^s_{\text{switch},j}$ (Online Updates) and $\mathbf{y}^s_{\text{init},j}$ are presented for comparison. These two ROM simulations use the same local ROBs; the only difference being that the $\mathbf{y}^s_{\text{switch},j}$ simulation performs online updates to the local ROBs using Algorithm 4. Examining the results in Fig. 5, the simulation with the online updates is able to achieve better accuracy than the simulation without updates – especially at the end of the simulation.

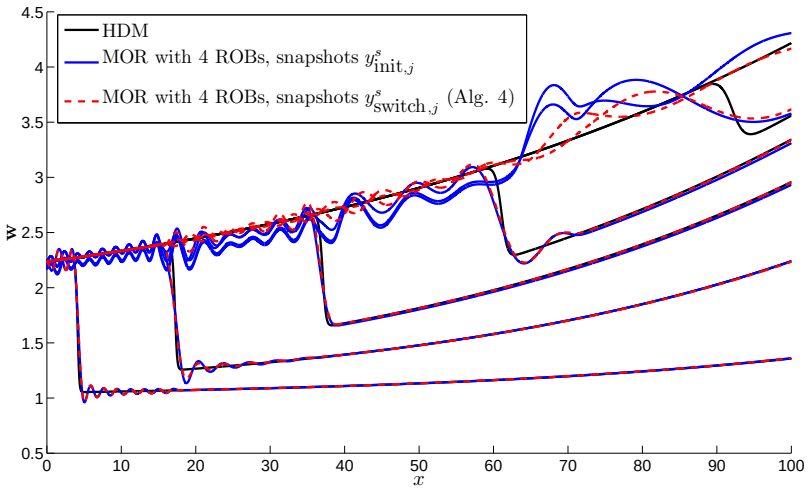


Figure 5. Performance of the local ROMs with $k_j = \{20, 14, 10, 4\}$ and $N_{\mathbf{V}} = 4$ for snapshot methods $\mathbf{y}^s_{\mathbf{init},j}$ and $\mathbf{y}^s_{\mathbf{switch},j}$ (Algorithm 4)

In Fig. 6, the computational cost and relative errors for simulations using online updates are reported for $N_{\mathbf{V}} = 1, 2, 3, 4$. These results show that the use of local ROBs results in faster ROMs for a given accuracy, but that there appears to be an upper limit on the performance benefits that can be gained by increasing $N_{\mathbf{V}}$. Note that without hyper-reduction, these ROM simulations still scale with the large dimension of the HDM, and as expected these simulations typically require more CPU time than the HDM. In [7], the authors show that hyper-reduction can be applied to the local ROM framework, resulting in significant speedups. Future work will focus on adapting the online updates proposed here to the local hyper-reduction method presented in [7].

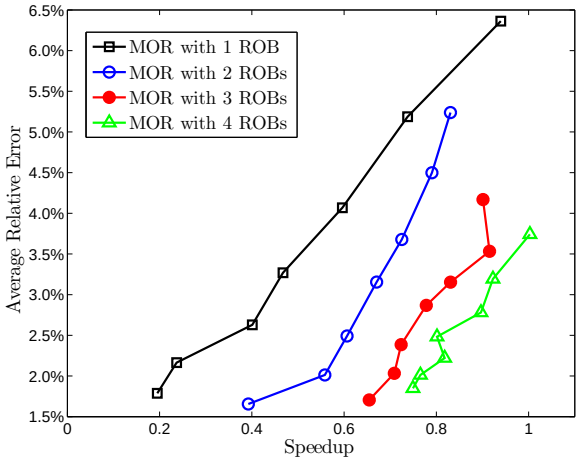


Figure 6. Performance of the nonlinear MOR method using online updates as a function of $N_{\mathbf{V}}$

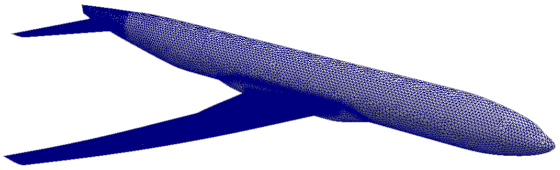
B. Acceleration Study of a Transport Aircraft

1. High-Dimensional Model

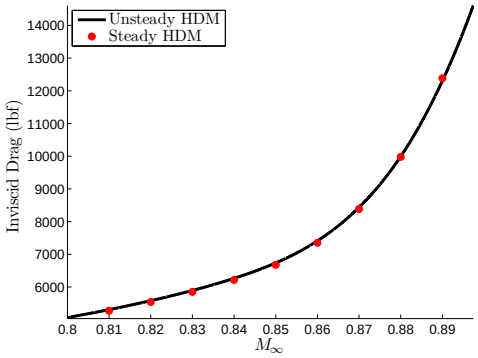
The second system that is considered is a transport aircraft subjected to a constant acceleration. The geometry of interest is NASA's Common Research Model (CRM).[13] The fluid mesh used for this study consists of 3,252,078 unstructured tetrahedra, with 617,864 fluid nodes. In this simulation, the Euler equations are solved using a node-based method, resulting in 3,089,320 degrees of freedom. The surface mesh of the model is depicted in Fig. 7 (a).

During the unsteady simulation, the aircraft angle of attack is held fixed at zero degrees, and the free stream Mach number is varied from $M_\infty = 0.8$ to $M_\infty = 0.9$ over 12.5 seconds using a single step ALE formulation.[14,15] The resulting acceleration of approximately 2.5 m/s^2 is fairly gradual, and as shown in Fig. 7 (b), it does not appear to introduce substantial unsteadiness into the simulation.

In an effort to make compressibility effects readily apparent, drag has been chosen as the output of interest. Note that since this is an inviscid simulation, the predicted drag is only a portion of the total drag.



(a) CRM surface mesh



(b) Steady and unsteady HDM results

Figure 7. High-dimensional model

2. Clustering Results

For this system, ROMs based on $N_{\mathbf{V}} = 1$, $N_{\mathbf{V}} = 2$, and $N_{\mathbf{V}} = 5$ local ROBs are considered. The k-means clustering results for $N_{\mathbf{V}} = 2$ and $N_{\mathbf{V}} = 5$ are shown in Fig. 8. Note that the overlap algorithm is performing as intended, adding snapshots to each cluster from its nearest neighbors in the simulation trajectory.

The centroids of these clusters are presented in Fig. 9 for $N_{\mathbf{V}} = 5$ and Fig. 10 for $N_{\mathbf{V}} = 2$. Again, these centroids are not physical states of the system, and upon close inspection the flow features are slightly blurred. This effect is more pronounced for $N_{\mathbf{V}} = 2$ since each cluster contains a greater variety of states. After clustering, the local ROBs are constructed using Algorithm 3 with $\mathbf{w}_{\text{ref},j} = \mathbf{w}^{(0)},\ j = 1, \cdots, N_{\mathbf{V}}$.

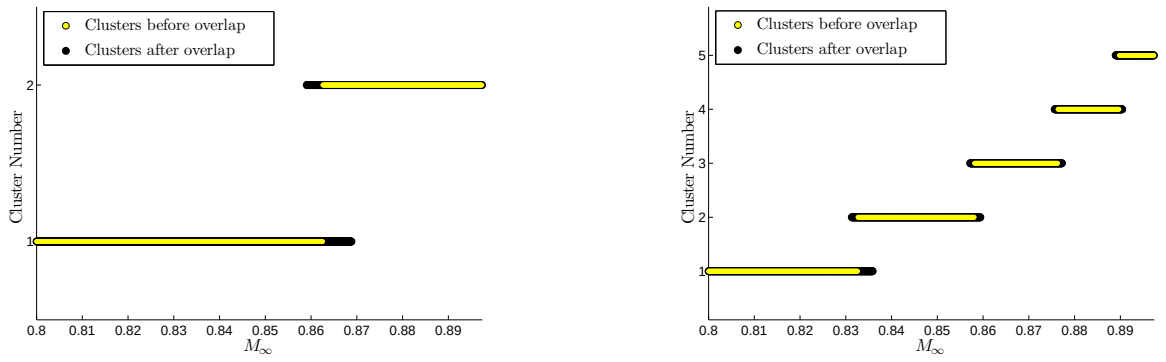
3. ROM Performance

The drag histories for several global ROM simulations are presented in Fig. 11 (a). Global ROMs with more than 50 basis vectors were able to reproduce the HDM results with less than 0.6% relative error.

Fig. 11 (b) and (c) show the drag histories for local ROMs with $N_{\mathbf{V}} = 2$ and $N_{\mathbf{V}} = 5$. As the number of local ROBs is increased, each individual ROB can be smaller for a given accuracy; however, since there is some redundancy required in the local ROB subspaces, more total basis vectors are required.

Fig. 12 (a) demonstrates the performance of the local ROMs with $N_{\mathbf{V}} = 4$ constructed using several of the snapshot choices discussed in Section V.A.3. Note that the online updates can be performed quickly, and provide a clear performance benefit over the alternative snapshot procedures.

Fig. 12 (b) compares the performance of the local ROMs using ROB updates as $N_{\mathbf{V}}$ is increased. Again, without hyper-reduction all of the ROM simulations ran slower than the HDM; however, performance was



(a) $N_{\mathbf{V}} = 2$ (b) $N_{\mathbf{V}} = 5$

Figure 8. State clusters before and after overlap algorithm

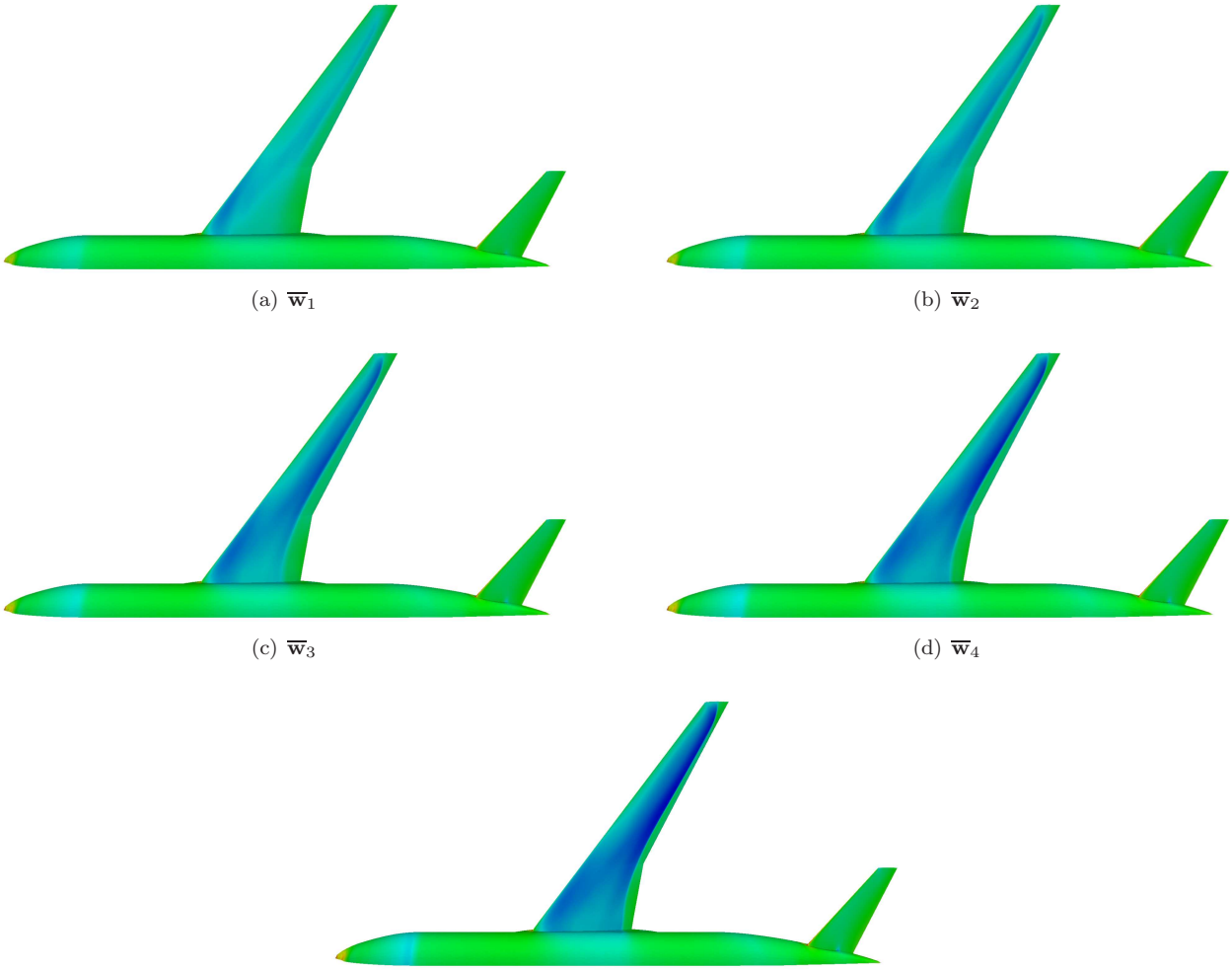
significantly improved through the use of local ROBs. For the most accurate simulations ($\leq 0.08\%$ average relative error in drag), the ROM with $N_{\mathbf{V}} = 2$ ran twice as fast as the global ROM, and the ROM with $N_{\mathbf{V}} = 5$ ran four times as fast as the global ROM.

VI. Conclusions and Future Work

A novel Model Order Reduction (MOR) method based on the concept of local bases is introduced. Like most MOR methods, this method consists of distinct offline and online phases. In the offline phase, precomputed states are partitioned into overlapping clusters using an unsupervised learning algorithm that relies on the concept of cluster connectivity. The local Reduced Order Bases (ROBs) are then constructed from states contained in each cluster using Proper Orthogonal Decomposition (POD). During the online phase, an appropriate local ROB is chosen at each time iteration based on the current state of the system. An inexpensive rank-one update is applied whenever a new local ROB is selected, resulting in search spaces that are tailored for the simulation. Applications to a model fluid problem and a three-dimensional CFD problem reveal that the proposed local MOR method can achieve high levels of accuracy using much smaller ROBs than would be required by a global MOR approach. The proposed method of online local ROB updates is shown to improve the accuracy of local ROM simulations. Future work will focus on adapting the proposed online update method to the local hyper-reduction framework introduced in [7].

Acknowledgments

The authors acknowledge partial support by the Army Research Laboratory through the Army High Performance Computing Research Center under Cooperative Agreement W911NF-07-2-0027, partial support by the Office of Naval Research under Grant N00014-11-1-0707, partial support by The Boeing Company under Contract Sponsor Ref. 45047, and partial support by a research grant from King Abdulaziz City for Science and Technology (KACST). The third author also acknowledges the support by a Department of Energy Computational Science Graduate Fellowship. The content of this publication does not necessarily reflect the position or policy of any of these supporters, and no official endorsement should be inferred.

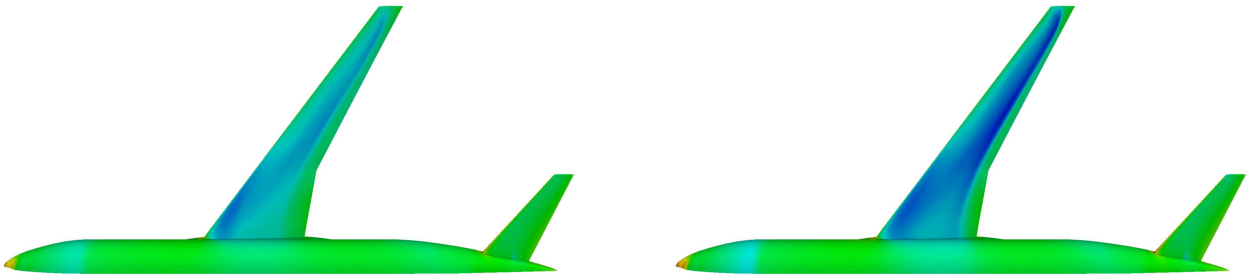


(a) $\overline{\mathbf{w}}_1$ (b) $\overline{\mathbf{w}}_2$

(c) $\overline{\mathbf{w}}_3$ (d) $\overline{\mathbf{w}}_4$

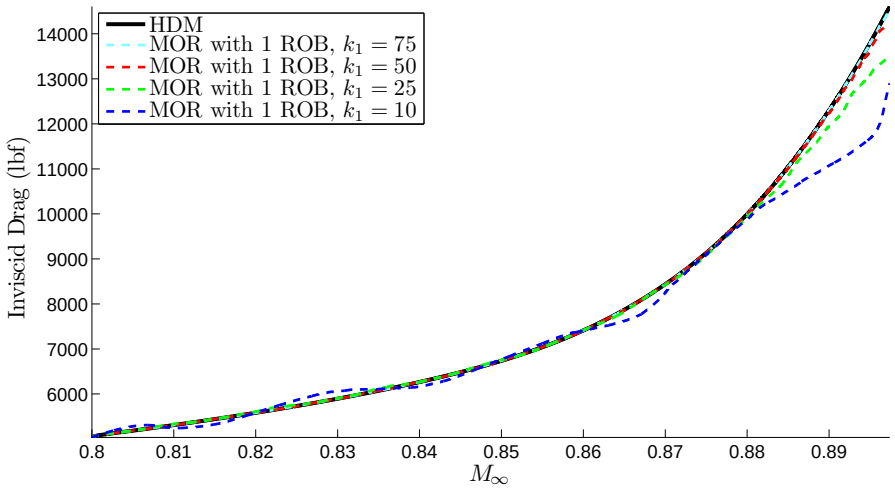
(e) $\overline{\mathbf{w}}_5$

Figure 9. Pressure contours for the cluster centroids ($N_{\mathbf{V}} = 5$)

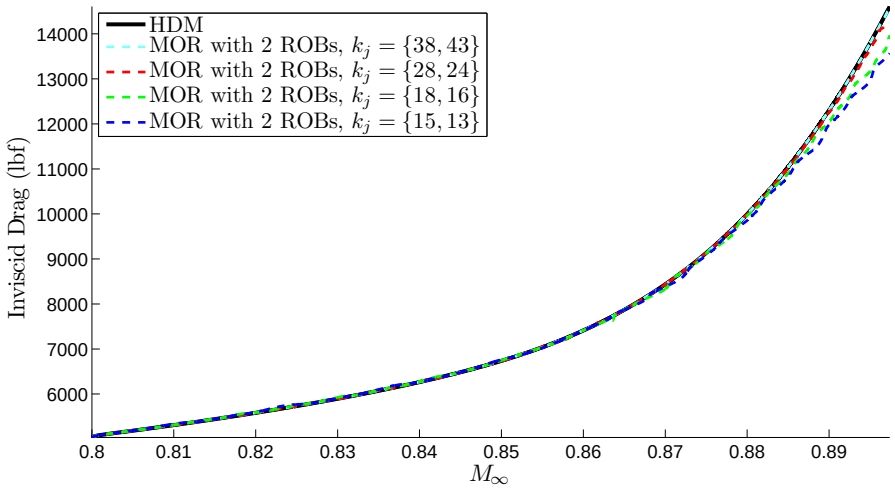


(a) $\overline{\mathbf{w}}_1$ (b) $\overline{\mathbf{w}}_2$

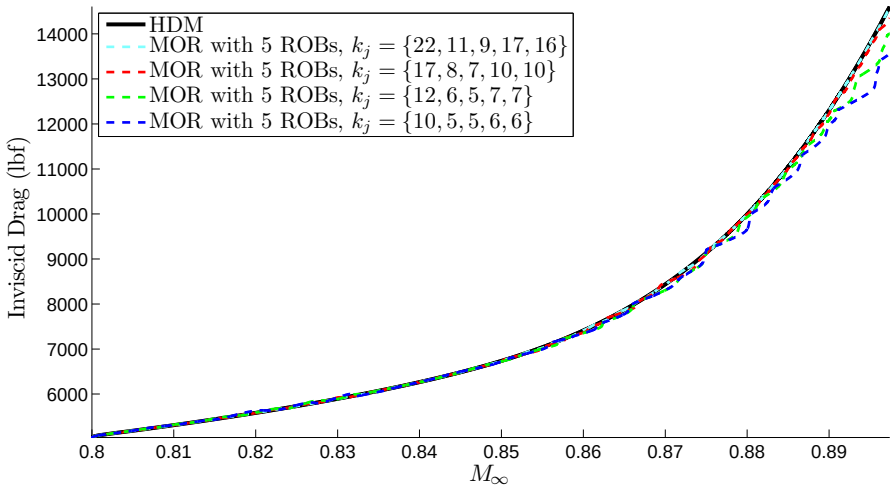
Figure 10. Pressure contours for the cluster centroids ($N_{\mathbf{V}} = 2$)



(a) $N_{\mathbf{V}} = 1$

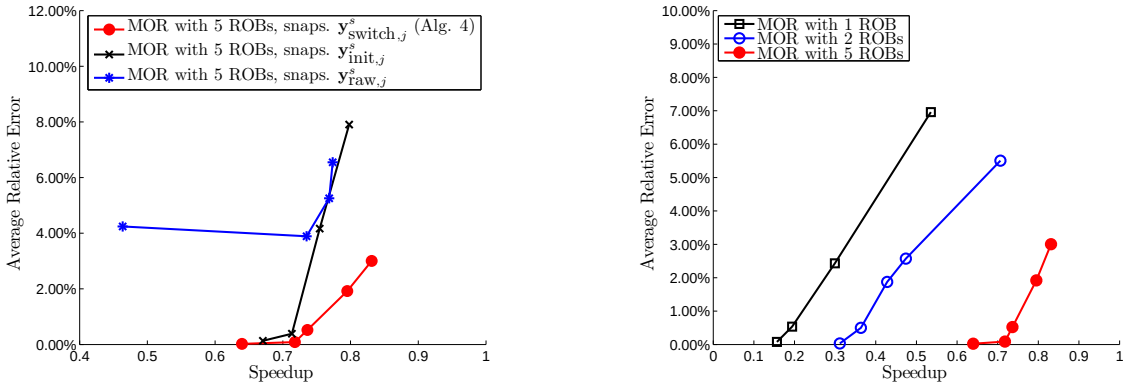


(b) $N_{\mathbf{V}} = 2$



(c) $N_{\mathbf{V}} = 5$

Figure 11. Performance of the nonlinear MOR method as a function $N_{\mathbf{V}}$ and k_j



(a) Performance of MOR method with $N_{\mathbf{V}} = 5$ as a function of snapshot method

(b) Performance of MOR method for $N_{\mathbf{V}} = 1, 2, 5$ using online updates

Figure 12. Performance of nonlinear MOR method as a function of snapshot method with $N_{\mathbf{V}} = 5$, and as a function of $N_{\mathbf{V}}$ using only the online updates

References

[1]Rewienski, M. and White, J., "Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations," *Linear Algebra and its Applications*, Vol. 415, No. 2-3, 2006, pp. 426–454.

[2]Lieu, T. and Farhat, C., "Adaptation of aeroelastic reduced-order models and application to an F-16 configuration," *AIAA Journal*, Vol. 45, No. 6, 2007, pp. 1244–1257.

[3]Amsallem, D. and Farhat, C., "Interpolation method for adapting reduced-order models and application to aeroelasticity," *AIAA Journal*, Vol. 46, No. 7, 2008, pp. 1803–1813.

[4]Amsallem, D., Cortial, J., and Farhat, C., "Toward Real-Time Computational-Fluid-Dynamics-Based Aeroelastic Computations Using a Database of Reduced-Order Information," *AIAA Journal*, Vol. 48, No. 9, 2010, pp. 2029–2037.

[5]Chaturantabut, S. and Sorensen, D., "Nonlinear Model Reduction via Discrete Empirical Interpolation," *SIAM Journal on Scientific Computing*, Vol. 32, No. 5, 2010, pp. 2737–2764.

[6]Carlberg, K., Bou-Mosleh, C., and Farhat, C., "Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations," *International Journal for Numerical Methods in Engineering*, Vol. 86, No. 2, 2011, pp. 155–181.

[7]Amsallem, D., Zahr, M., and Farhat, C., "Nonlinear Model Order Reduction Based on Local Reduced-Order Bases," *International Journal for Numerical Methods in Engineering, in press*, 2012, pp. 1–31.

[8]Bui-Thanh, T., Willcox, K., and Ghattas, O., "Parametric Reduced-Order Models for Probabilistic Analysis of Unsteady Aerodynamic Applications," *AIAA Journal*, Vol. 46, No. 10, 2008, pp. 2520–2529.

[9]Sirovich, L., "Turbulence and the dynamics of coherent structures. Part I: Coherent structures," *Quarterly of applied mathematics*, Vol. 45, No. 3, 1987, pp. 561–571.

[10]Zahr, M., "Comparison of Model Reduction Techniques on High-Fidelity Linear and Nonlinear Electrical, Mechanical, and Biological Systems," *Technical Report, Stanford University*, Sept. 2010, pp. 1–30.

[11]Carlberg, K., Cortial, J., Amsallem, D., Zahr, M., and Farhat, C., "The GNAT nonlinear model reduction method and its application to fluid dynamics problems," *AIAA Paper 2011-3112, 6th AIAA Theoretical Fluid Mechanics Conference*, July 2011, pp. 1–24.

[12]Brand, M., "Fast low-rank modifications of the thin singular value decomposition," *Linear Algebra and its Applications*, Vol. 415, 2006, pp. 20–30.

[13]Vassberg, J. C., DeHaan, M. A., Rivers, S. M., and Wahls, R. A., "Development of a Common Research Model for Applied CFD Validation Studies," *26th AIAA Applied Aerodynamics Conference, August 2008, AIAA-2008-6919*, 2008.

[14]Farhat, C., Harris, C., and Rixen, D., "Expanding a Flutter Envelope Using Accelerated Flight Data: Application to an F-16 Fighter Configuration," *41st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, April 2000, AIAA-2000-1702*, 2000.

[15]Farhat, C., Geuzaine, P., Brown, G., and Harris, C., "Nonlinear Flutter Analysis of an F-16 in Stabilized, Accelerated, and Increased Angle of Attack Configurations," *43rd AIAA/ASME/ASCE/AHS/ASC Structural Dynamics and Materials Conference, April 2002, AIAA-2002-1490*, 2002.