Geometric Multiscale Reduction for Autonomous and Controlled Nonlinear Systems

Jake Bouvrie and Mauro Maggioni

Abstract— **Most generic approaches to empirical reduction of dynamical systems, controlled or otherwise, are global in nature. Yet interesting systems often exhibit multiscale structure in time or in space, suggesting that localized reduction techniques which take advantage of this multiscale structure might provide better approximations with lower complexity. We introduce a snapshot-based framework for localized analysis and reduction of nonlinear systems, based on a systematic multiscale decomposition of the statespace induced by the geometry of empirical trajectories. A given system is approximated by a piecewise collection of low-dimensional systems at different scales, each of which is suited to and responsible for a particular region of the statespace. Within this framework, we describe localized, multiscale variants of the proper orthogonal decomposition (POD) and empirical balanced truncation methods for model order reduction of nonlinear systems. The inherent locality of the treatment further motivates control strategies involving collections of simple, local controllers and raises decentralized control possibilities. We illustrate the localized POD approach in the context of a high-dimensional fluid mechanics problem involving incompressible flow over a bluff body.**

I. Introduction

The analysis and control of large-scale dynamical systems has emerged as a ubiquitous challenge across a range of fields. Increased computing power has made the simulation of previously untouchable systems, such as those arising in molecular dynamics and control of fluids, a realistic possibility, but one that is still fraught with difficulty. Without some form of dimension reduction, simulation, analysis and controller design is often intractable. And yet many dynamical systems naturally enjoy *multiscale structure*, both in time and in space. Taken together, these considerations underscore and motivate the need for model reduction techniques that take advantage of this multiscale structure whenever possible. The benefits are clear: localized, multi-resolution approximations have the potential to offer better accuracy at lower complexity, leading to faster simulations, simpler controllers and revealing analyses or interpretations.

We introduce a snapshot-based framework for localized analysis and reduction of nonlinear systems, based on a systematic multiscale decomposition of the statespace induced by the geometry of empirical system trajectories. A given system is approximated by a collection of low-dimensional systems at different scales, each of which is suited to and responsible for a particular region of the statespace. The number of low-dimensional systems, their dimensions and their regions of responsibility are all chosen adaptively, given a prescribed precision $\epsilon > 0$. Within this framework, we describe localized variants of the proper orthogonal decomposition (POD) and empirical balanced truncation methods for model order reduction of nonlinear systems.

Given sample measurements from one or more simulated state trajectories, a multiscale tree decomposition of the subspace containing the trajectories is first constructed using a modified version of the technique proposed in [1]. To each node of the tree we associate a subset of the samples. Next, approximating subspaces are estimated from the samples at each node. How these subspaces are computed depends on the analysis that is desired: a localized POD-based reduction method is obtained by taking the SVD and applying an affine Galerkin projection that is valid locally; local empirical balanced truncation is made possible by way of a local observability subspace estimation step combined with an affine Petrov-Galerkin projection defined by a local balancing transformation and its inverse. A system may then be simulated, controlled or analyzed by switching between or blending the reduced, local systems as the global trajectory evolves throughout the statespace. The basis vectors encoding the local approximating subspaces may be collectively viewed as a multiscale, data-dependent dictionary designed to efficiently encode the behavior of a nonlinear system.

The advantages of this approach may be divided into two categories, one analytical and the other computational. From an analytical perspective, the framework described here can potentially decompose complex phenomena into a multi-resolution hierarchy of simple parts which may be analyzed separately. To this end we describe a localized variant of dynamic mode decomposition (DMD) for understanding the behavior of nonlinear systems at different stages of a simulation without having to manually decide which time intervals to consider. From a computational perspective, a localized approximation scheme can indeed be expected to provide better approximation properties at lower complexity, since the approximating subspaces do not need to capture global phenomena, but only need to be good locally. This then opens the door to control strategies involving a collection of simple, local controllers and raises interesting decentralized control possibilities which could leverage parallelization and sensor/actuator locality.

Due to space limitations we will forgo a detailed review of the existing reduction techniques upon which we build. Detailed expositions are readily available; some good references for POD and balanced truncation of linear systems include [2], [3]. Reference [4] discusses what are now standard

J. Bouvrie is with the Department of Mathematics, Duke University, Box 90320, Durham, NC 27708, USA jvb@math.duke.edu.

M. Maggioni is with the Departments of Mathematics and Computer Science, Duke University, Box 90320, Durham, NC 27708, USA mauro@math.duke.edu.

Research supported by DARPA FA8650-11-1-7150 SUB#7-3130298, MSEE FA8650-11-1-7150; Washington State U. SUB#113054 G002745; NSF IIS-08-03293, DMS-08-47388; ONR N00014-07-1-0625.

Version: September 9, 2012

extensions of these methods to nonlinear systems, and [5], [6], [7] discuss DMD. We begin by describing the multiscale tree decomposition for snapshot data in Section II. Local reduction and spectral analysis algorithms are developed in Section III, with Section IV focusing specifically on the question of efficient simulation and controller design with local approximating dynamical systems of low dimension. Finally we conclude in Section V with an illustrative application of the localized POD method proposed here to a high-dimensional fluid mechanics problem involving a 2-D incompressible flow over a bluff body. The simulations reveal that a localized reduction approach yields greater accuracy with a simpler model.

A. Prior Work

Localized reduction and analysis of dynamical systems often proceeds on the basis of a priori domain knowledge and manual intervention [8]. Local modeling and dimensionality reduction for static datasets is a well explored space ([9], [10] are two popular algorithms), but to our knowledge there has been no attempt to extend these ideas to dynamical systems. In [11] a local POD scheme is proposed, however localization is achieved by binning snapshots in time according to a heuristic and computationally costly procedure that attempts to successively minimize reconstruction error by applying POD to different simulation intervals. Geometry of the system trajectories is not taken into account and multiscale structure is not explicitly explored in time, or explored at all in space. Various efforts have included wavelet analysis of snapshots to capture spatio-temporally localized phenomena, however the resulting reduction methods are often tailored to the specific problem (see e.g. [12]) and do not support balancing for controlled dynamical systems.

II. Geometric Data Tree Construction

The construction of a multiscale tree $\mathcal{T}$ organizes a dataset of N samples $\{x_i\}_{i=1}^N$ into a spatially hierarchical structure. As the geometry of a continuous system will evolve more or less smoothly over time, a spatial analysis will also tend to identify geometrically self-similar temporal extents. Each "node" in the tree corresponds to a subset of the samples, and children beneath a given node partition the data points belonging to that node. Samples are not assigned to nodes arbitrarily; geometric properties of the dataset are used to partition the data into clusters. A cluster may be further split into smaller clusters depending on whether more or less resolution is needed to approximate points in a particular region of the statespace to the prescribed accuracy.

We will follow the development in [1] to construct a suitable multiscale decomposition of a metric measure space $(\mathcal{M}, \rho, \mu)$ endowed with (Borel) probability measure μ and metric ρ. In practice $(\mathcal{M}, \rho, \mu)$ will be a finite discrete metric space, μ will simply be the counting measure and $\mathcal{M} \subset \mathbb{R}^D$. Let $B_r(x)$ denote the ρ-ball of radius $r > 0$ centered at $x \in \mathcal{M}$.

Definition 1 *([1], Def. 1) A* **tree decomposition** $\mathcal{T}$ *of a D-dimensional metric measure space $(\mathcal{M}, \rho, \mu)$ is a family of open sets in $\mathcal{M}$, $\{\mathsf{C}_{j,k}\}_{k \in \mathcal{K}_j, j \in \mathbb{Z}}$, called* dyadic cells*, such that*

(i) *for every $j \in \mathbb{Z}$, $\mu\big(\mathcal{M} \setminus \cup_{k \in \mathcal{K}_j} \mathsf{C}_{j,k}\big) = 0$;*
(ii) *for $j' \geq j$ and $k' \in \mathcal{K}_{j'}$, either $\mathsf{C}_{j',k'} \subseteq \mathsf{C}_{j,k}$ or $\mu\big(\mathsf{C}_{j',k'} \cap \mathsf{C}_{j,k}\big) = 0$;*
(iii) *for $j < j'$ and $k' \in \mathcal{K}_{j'}$, there exists a unique $k \in \mathcal{K}_j$ such that $\mathsf{C}_{j',k'} \subseteq \mathsf{C}_{j,k}$;*
(iv) *each $\mathsf{C}_{j,k}$ contains a point $c_{j,k}$ such that $B_{c_1 2^{-j}}(c_{j,k}) \subseteq \mathsf{C}_{j,k} \subseteq B_{2^{-j}}(c_{j,k})$, for a constant c_1 depending on the intrinsic geometric properties of $\mathcal{M}$.*

Given a dataset $\{x_i\}_i$, a tree satisfying (or approximately satisfying) the properties above may be constructed in a number of ways, for instance by recursive spectral partitioning, iterated k-means clustering, or cover trees [13]. For the simulations described here we applied the METIS [14] partitioning algorithm to the k-nearest neighbor weight matrix with neighbor weights $W_{ij} = \exp(-\|x_i - x_j\|_2^2/\sigma_i\sigma_j)$, where σ_i is taken to be the distance between x_i and its $\lfloor k/2 \rfloor$-th nearest neighbor. If the snapshots $\{x_i\}_i$ are sampled from a manifold $\mathcal{M}$, the combinatorial Laplacian $L = \mathrm{diag}(W\mathbf{1}) - W$ can be shown to approximate the Laplace-Beltrami operator $\Delta_{\mathcal{M}}$ on $\mathcal{M}$.

METIS produces a dyadic tree containing the collection of cells $\{\mathsf{C}_{j,k}\}$ down to a scale where the cells contain a prespecified maximum number points. The tree is then pruned by eliminating leaf nodes until a principal components approximation of the data at a node, assuming dimension less than or equal to the parent's approximating dimension, fails to adhere to a prescribed L_2 approximation precision $\epsilon > 0$. See [1, Sec. 7.1] for details explaining the pruning procedure. Different branches of the tree may therefore reach to different scales. We will let $j = J$ denote the finest resulting scale (where there are at least the minimum number of points passed as a parameter to METIS), and $j = 0$ denote the coarsest scale (which includes all points in the dataset). The number of neighbors k used to construct W as well as the parent subspace dimensions and precision parameter ϵ used for pruning typically must be chosen based on the particular application. The cost of constructing a tree is $\mathcal{O}(DN(d_\iota^2 + \log N))$, where d_ι is the approximate intrinsic dimension of the data and of the planes used for approximation.

In the discussion below, we will denote by $\mathsf{Leaf}(\mathcal{T})$ the set of scale/cell index pairs (j,k) such that $\{\mathsf{C}_{j,k}\}_{(j,k) \in \mathsf{Leaf}}$ are the leaf node cells of $\mathcal{T}$. When the tree in question is clear from the context, we will simply write Leaf.

III. Local Reduction

In this section we propose localized, multiscale variants of the proper orthogonal decomposition (POD) and empirical balanced truncation methods for reduction of nonlinear systems. A localized variant of the dynamic mode decomposition (DMD) approach to system analysis is also briefly described. Local POD may be applied to any system, however it is best suited to autonomous and locally approximately normal systems. For dynamical control systems, the local empirical balanced truncation algorithm we will describe takes into account the input-output behavior of the system, and is preferred. For analysis of autonomous systems suspected to be locally non-normal, we suggest the DMD-based technique.

In the sections that follow we will consider a general nonlinear system of the form

$$\begin{cases} \dot{\mathbf{X}} &= f(\mathbf{X}, \mathbf{u}) \\ \mathbf{y} &= h(\mathbf{X}) \end{cases}$$

with state $\mathbf{X} \in \mathbb{R}^D$, input $\mathbf{u} \in \mathbb{R}^p$ (omitted for autonomous systems), output $\mathbf{y} \in \mathbb{R}^q$, and continuous dynamics $f : \mathbb{R}^D \times \mathbb{R}^p \to \mathbb{R}^D$ and output function $h : \mathbb{R}^D \to \mathbb{R}^q$. The vectors $\{\mathbf{e}_i\}_i$ will refer to the canonical basis vectors of a Euclidean space determined by the context in which they are used.

A. Local POD for Controlled and Autonomous Systems

Perhaps the simplest method for reducing the order of nonlinear dynamical systems and systems described by PDEs is the proper orthogonal decomposition (POD) method (see e.g. [3] for a detailed introduction). Given a collection of trajectory samples, $\{x_i \in \mathbb{R}^D\}_{i=1}^N$, POD returns a low-dimensional subspace which best approximates the subspace containing the state trajectories in an L_2 sense. For linear systems, the approximation is optimal.

Given a multiscale decomposition $\mathcal{T}$ of a snapshot dataset $\{x_i\}_i$ as described in Section II, we can systematically *localize* POD-based reduction as follows:

1) Apply POD to the samples belonging to leaf node dyadic cells (separately).
2) Define local, reduced dynamical systems at each leaf node via appropriate Galerkin projections.
3) Combine local dynamical systems by, e.g. averaging or a switching rule.

Each of these steps will be explained in turn. For each leaf-node cell $\mathsf{C}_{j,k}, (j,k) \in \mathsf{Leaf}$, we define the local mean $c_{j,k} = |\mathsf{C}_{j,k}|^{-1} \sum_{x \in \mathsf{C}_{j,k}} x$ and covariance

$$\mathrm{cov}_{j,k} = |\mathsf{C}_{j,k}|^{-1} \sum_{x \in \mathsf{C}_{j,k}} (x - c_{j,k})(x - c_{j,k})^*. \qquad (1)$$

Taking the SVD

$$\mathrm{cov}_{j,k} = U_{j,k} \Sigma_{j,k}^2 U_{j,k}^*, \qquad (2)$$

let $P_{j,k}$ denote the matrix whose rows are the columns of $U_{j,k}$ corresponding to the $r < D$ largest singular values. A locally-defined Galerkin projection is obtained by considering the dynamics of $\mathbf{X}' = \mathbf{X} - c_{j,k}$ and projecting down to the subspace spanned by the r orthonormal rows of $P_{j,k}$. The reduced-order dynamical system valid in the vicinity of $c_{j,k}$ is thus given by

$$\begin{cases} \dot{\mathbf{x}}_{j,k} = P_{j,k} f\big(P_{j,k}^* \mathbf{x}_{j,k} + c_{j,k}\big) \\ \mathbf{y} = h\big(P_{j,k}^* \mathbf{x}_{j,k} + c_{j,k}\big) \end{cases} \qquad (3)$$

for each $(j,k) \in \mathsf{Leaf}$. We note that the local systems need not have the same dimension. In practice, the dimension of a leaf node system is chosen adaptively on the basis of the tree precision parameter ϵ as described in Section II.

By building the tree $\mathcal{T}$ and estimating the (affine) subspaces spanned by the rows of the $P_{j,k}$, we are approximating the statespace with an arrangement of low-dimensional planes which pass through the respective centers $c_{j,k}$. The dynamical systems (3) attempt to approximate the original system by projecting the vector field f onto these planes. We will discuss in Section IV below how to construct a model of the system that is valid globally, by combining the dynamics of the local reduced systems.

B. Local Empirical Balanced Truncation

The POD method renders a set of $r < D$ modes which attempt to capture the input-to-state behavior of a controlled system, but entirely ignores state-to-output properties. Balanced truncation (see [3], [2] for details) is a method that attempts to find a low-dimensional approximating subspace by choosing directions which are simultaneously the most controllable (capturing input-to-state energy) and the most observable (capturing state-to-output energy). An empirical, snapshot based variant of balanced truncation (also known as balanced POD) for linear systems was first studied in [15], and was subsequently extended to nonlinear systems in [4]. The approach proposed in [4] essentially carries out the linear balancing theory and applies a Petrov-Galerkin projection to a nonlinear system as if it were linear, however empirical estimation of the observability and controllability gramians is different from the method of snapshots for linear systems. While snapshots are typically computed from primal and adjoint systems by matrix multiplications in the linear setting [16], for nonlinear systems true state/output trajectories are simulated from impulsive input (as the primal simulation), and from impulsive initial conditions (as the "adjoint" simulation). We will largely follow the balanced truncation process for nonlinear systems proposed in [4], but adapted to the localized setting considered here by way of some important modifications.

The local empirical balanced truncation methodology we propose consists of the following steps:

1) Controllability Tree Construction and Local POD: First, perform local POD following the approach described in Section III-A: Simulate the system with $\mathbf{u}(t) = \mathbf{e}_i \delta(t)$ for $i = 1, \ldots, p$ and $\mathbf{X}(0) = 0$, collect N_c state trajectory samples, and build a controllability tree $\mathcal{T}_c$ to precision ϵ following the process described in Section II. Then, for each leaf node in the tree, compute the partial isometries $P_{j,k}$ as described in Section III-A. Temporarily use more POD modes than is necessary for the final reduced system so that the observability simulations described immediately below can reliably estimate the observable subspace.

2) Observability Subspace Estimation: Let $\{\phi_{j,k}^{(i)}\}_{i=1}^r$ denote the top r orthonormal basis vectors spanning the POD subspace associated to node (j,k), computed above according to (1)-(2) (note that r may be different for each $(j,k) \in \mathsf{Leaf}$). These are the rows of $P_{j,k}$. For each leaf node $(j,k) \in \mathsf{Leaf}(\mathcal{T}_c)$, run r simulations of the original high-dimensional system from initial conditions $\mathbf{X}(0) = \phi_{j,k}^{(i)} + c_{j,k}, i = 1, \ldots, r$, respectively, setting $\mathbf{u}(t) = 0$. These simulations assess observability *locally*, and so should be stopped shortly after the state trajectory leaves the relevant region of responsibility defined by the centers $\{c_{j,k}\}$ (see Definition 2 and surrounding discussion below). Global observability properties are therefore measured by way of $r \cdot |\mathsf{Leaf}|$ *short* simulations.

Estimation of the observability gramian in [4] requires at least D simulations, corresponding to the initial conditions

$\mathbf{X}(0) = \mathbf{e}_i\delta(t), i = 1,\ldots,D$. For large systems (which motivate reduction in the first place), this can be prohibitively expensive. Fluid control problems can easily involve a 10^{3-6} dimensional state variable, for example. Computational considerations aside, if an r-dimensional local POD subspace can well-approximate the dynamics locally, then one can expect that the output trajectories should approximately lie within a subspace of dimension no greater than r. This observation justifies simulating the system starting only from the controllability POD modes $\{\phi^{(i)}_{j,k}\}_i$, and is similar in spirit to output-projection approximations applied in the context of linear systems [16]. On the other hand, from a geometric point of view, we would like to perform a localized balancing and so wish to assess controllability and observability only locally as well. We therefore need to simulate the system locally near each affine subspace spanned by the $\{\phi^{(i)}_{j,k}\}_i$, and capture observability properties within the relevant region of responsibility of the statespace.

Let $y^{(i)}_{j,k}(\alpha,\beta)$ denote the response of output ($\mathbf{y}$) coordinate $\alpha \in \{1,\ldots,q\}$ at time $t_\beta, \beta \in \{1,\ldots,N_o\}$ simulated using $\mathbf{X}(0) = \phi^{(i)}_{j,k} + c_{j,k}$ as the initial condition, and define the snapshot vector $d_{j,k}(\alpha,\beta) = \left(y^{(i)}_{j,k}(\alpha,\beta)\right)_{i=1}^{r} \in \mathbb{R}^r$. We can think of these simulations as yielding the sampled output measurement collections $\{d_{j,k}(\alpha,\beta)\}_{\alpha,\beta}$ for each $(j,k) \in \mathsf{Leaf}$. The simulation procedure above leads to a rank-r approximation of the true *local* observability gramian $W^o_{j,k}$ given by

$$W^o_{j,k} \approx (P^*_{j,k}Y_{j,k})(P^*_{j,k}Y_{j,k})^* = P^*_{j,k}\widetilde{W}^o_{j,k}P_{j,k}$$

where $Y_{j,k}$ is the $(r \times N_o q)$ matrix storing the snapshots $\{d_{j,k}(\alpha,\beta)\}_{\alpha,\beta}$ and $\widetilde{W}^o_{j,k} = Y_{j,k}Y^*_{j,k}$ is the $(r \times r)$ reduced observability gramian local to leaf node (j,k) of $\mathcal{T}_c$. *We will overload the notation $W^o_{j,k}$ and use it to refer to the local low-rank approximation*. For purposes of computing a balancing transformation (discussed below), one need not (and should not) explicitly form the large $(D \times D)$ matrix $W^o_{j,k}$ unless $D < N_o q$. We will proceed under the assumption that $D \gg N_o q$.

Finally, as an organizational step, define the cell $\mathsf{D}_{j,k} = \{d_{j,k}(\alpha,\beta)\}_{\alpha,\beta}$ and associate this cell with node (j,k) of $\mathcal{T}_c$. Conceptually, one can alternatively think of organizing the cells $\mathsf{D}_{j,k}$ into an observability tree $\mathcal{T}_o$ with structure identical to that of the controllability tree $\mathcal{T}_c$, and invoking the canonical matching between the two trees.

3) Local Balancing and Truncation: Next, apply empirical balancing and truncation separately at each leaf node given the respective controllability and observability snapshot pairs $(\mathsf{C}_{j,k}, \mathsf{D}_{j,k})$. The local balancing transformations $T_{j,k}$, one for each $(j,k) \in \mathsf{Leaf}(\mathcal{T}_c)$, are computed as follows. Let

$$H_{j,k} = Y^*_{j,k}P_{j,k}X_{j,k}$$

denote the $(N_o q \times N_c)$ Hankel matrix of inner-products local to node (j,k), where $Y_{j,k}$ is the matrix of observability snapshots in $\mathsf{D}_{j,k}$ as defined above, and $X_{j,k}$ is the $(d \times N_c)$ matrix of controllability snapshots in $\mathsf{C}_{j,k}$. Compute the (economy sized) SVD of $H_{j,k}$, $H_{j,k} = U_{j,k}\Sigma_{j,k}V^*_{j,k}$, organized so that the singular values are sorted in descending order by magnitude and stored along the main diagonal of $\Sigma_{j,k}$. Let $\tilde{U}_{j,k}, \tilde{V}_{j,k}$ denote the submatrices of $U_{j,k}, V_{j,k}$ which include the first $d < D$ columns, respectively, and let $\tilde{\Sigma}_{j,k}$ denote the top-left $(d \times d)$ block of $\Sigma_{j,k}$. Then the truncated balancing coordinate transformation is given by the pair

$$T^*_{j,k} = P^*_{j,k}Y_{j,k}\tilde{U}_{j,k}\tilde{\Sigma}^{-1/2}_{j,k}, \qquad T^{-1}_{j,k} = X_{j,k}\tilde{V}_{j,k}\tilde{\Sigma}^{-1/2}_{j,k}.$$

A balanced, reduced order system valid in a local region of the statespace near $c_{j,k}$ is obtained by performing a Petrov-Galerkin projection onto the d directions corresponding to the largest singular values of $H_{j,k}$, giving

$$\begin{cases} \dot{\mathbf{x}}_{j,k} = T_{j,k}f\big(T^{-1}_{j,k}\mathbf{x}_{j,k} + c_{j,k}\big) \\ \quad \mathbf{y} = h\big(T^{-1}_{j,k}\mathbf{x}_{j,k} + c_{j,k}\big) \end{cases} \tag{4}$$

for each $(j,k) \in \mathsf{Leaf}$.

We note that in the local POD and balancing algorithms described above, time enters via the locality of the plane projection operators. By contrast, global approaches ignore time.

C. Dynamic Mode Decomposition and Non-Normal Approximations

The reduction methods discussed above make a strong *normality assumption*. Sampled trajectories are analyzed as if they were generated from a linear system by way of the singular value decomposition, which is really only useful when the system may be locally well-approximated by a linearization $\dot{\mathbf{X}} = A\mathbf{X}$ involving a *normal* matrix A. For many systems, particularly nonlinear systems, this may not be the case, and non-normal matrices can generate rich dynamics not predicted by the spectrum (see [17] for a detailed discussion of non-normal matrices).

This observation motivates an alternative approach which does not use the SVD. *Dynamic mode decomposition* (DMD) [5], [7], [6] is a method that analyzes empirical trajectory data by applying a variant of the Arnoldi iteration for non-symmetric matrices [18]. The Arnoldi algorithm is an iterative algorithm for finding eigenvalues of a matrix A requiring only a black box capable of computing the product Ax. DMD feeds Arnoldi sequential samples generated by a discrete-time nonlinear system, $x_{k+1} = f(x_k)$ (in place of $x_{k+1} = Ax_k$), and can be viewed as computing eigenvalues and eigenvectors of a non-symmetric linear approximation to the (unknown) nonlinear system. Here, time enters explicitly.

Dynamic mode decomposition has recently been applied to understand unsteady fluid flow problems [6], [8], but this and other problem domains require *a priori* knowledge specifying appropriate time intervals to study (see e.g. [8], Sec. 6.3). The modes returned by DMD may not make sense if long trajectories are analyzed, and equilibrium, transient or cycle behaviors become conflated. Similarly, short trajectories many not include enough information to deliver revealing information.

Our contribution here is to simply suggest that 1) Applying DMD is a good idea for gaining insight into the behavior of nonlinear systems, and 2) Applying DMD locally, as determined by a geometric multiscale decomposition, can resolve some of the difficulty in deciding how and where to apply DMD. We do not give a detailed description of

the DMD algorithm or present experimental validation, but remark that a naive localization of DMD can be obtained by following the approach described in Section III-A, replacing POD with DMD.

The tree construction technique described in Section II decomposes the statespace into geometrically-meaningful regions. For continuous systems describing physical processes it is likely that sequential trajectory snippets will be placed into the leaf node cells $\mathsf{C}_{j,k}$[1], in which case these snippets can be fed directly to DMD (assuming one keeps track of the time attached to each observation). If samples are missing, then the map f will need to be applied as necessary to fill in temporal gaps.

IV. Simulation and Controller Design

With a collection of locally defined dynamical systems on hand, it remains to specify how to combine the local systems into a globally defined system. A given system may be modeled globally by activating a local system whenever the trajectory is in or near the area of the statespace for which the local system is "responsible". We define these areas as follows.

Definition 2 *The statespace* **regions of responsibility** *are defined as the Voronoi regions of the statespace induced by the centers $\{c_{j,k}\}_{(j,k)\in\mathsf{Leaf}}$ under the metric ρ. In other words, a particular local system with index (j',k') is responsible for a state $s\in\mathbb{R}^D$ if s is ρ-closer to $c_{j',k'}$ than any other leaf node center.*

A global system may be most easily defined by applying a switching rule, wherein only one local system is active at any point in time, and the active system is swapped out for another whenever a responsibility region boundary is crossed. While simple and computationally inexpensive, the switching rule can potentially lead to non-smooth transitions near boundaries, when an old model is dropped and a new set of projections are applied. One solution to this problem is to blend several reduced systems in the vicinity of a boundary in proportion to the distance from the current state to a given center $c_{j,k}$. We consider each of these possibilities, switching and blending, in turn.

In the following we will adopt a more generic notation and assume that the local, reduced dynamical systems are respectively parameterized by the tuples $\{(\Phi_i,\Psi_i,c_i,f)\}_{i=1}^{|\mathsf{Leaf}|}$ so that we may define the maps

$$f_i(x)\triangleq\Phi_i f\big(\Psi_i^* x+c_i\big) \tag{5}$$

$$h_i(x)\triangleq h\big(\Psi_i^* x+c_i\big) \tag{6}$$

with $f_i:\mathbb{R}^{d_i}\to\mathbb{R}^{d_i}, d_i=\dim(\text{range }\Phi_i)$, and write

$$\begin{cases}\dot{\mathbf{x}}_i(t)=f_i(\mathbf{x}_i)\\ \mathbf{y}(t)=h_i(\mathbf{x}_i)\end{cases} \tag{7}$$

for the reduced local system associated to leaf node i of the data tree. For POD, $\Phi_i=\Psi_i=P_i$, while for balanced systems $\Phi_i=T_i,\Psi_i=T_i^{-*}$, where we have replaced the subscript index notation (j,k) with i for brevity.

[1]We found that this is indeed the case for the fluid flow problem examined in Section V.

Before continuing, we make an important observation regarding (7). The systems (7) are responsible for localized regions of the statespace, characterized to a prescribed degree of accuracy by low-dimensional planes. As such, we argue that it is reasonable to attempt to approximate the low dimensional dynamics (5) locally on these planes, by fitting simple functions that can be quickly evaluated (e.g. a linear operator, neural network, or kernel function approximation scheme). In the case of fluid simulations, for example, this could avoid the solution of large linear systems at each time step, and lead to substantial computational savings. The local, geometric nature of the piecewise decomposition of the statespace and dynamics may plausibly support higher accuracy approximations consisting of simpler localized models, as compared to global approximation approaches.

A. Switching Between Local Systems

A switching rule is the simplest possible means for combining local systems: a new model is swapped in whenever a responsibility region boundary is crossed. The new system picks up where the previous one left off via its initial condition. Without loss of generality, we can consider the discrete collection of time instances $t_0<\cdots<t_n<t_{n+1}<\cdots<t_T$ at which the simulation computes the globally defined trajectory $\mathbf{x}(t)$. Suppose the current (or initial) local system has index i_0 so that $\mathbf{x}(t)=\mathbf{x}_{i_0}(t)$, where $\mathbf{x}_{i_0}(t)$ is a solution to (7), from time $t=t_0$ until time $t_{i_0,i_1}\in(t_0,t_T)$ when the boundary between system i_0 and some other system $i_1\neq i_0$ is crossed. That is,

$$\operatorname*{argmin}_j \rho\big(\Psi_{i_0}^*\mathbf{x}(t)+c_{i_0},c_j\big)=\begin{cases}i_0, & t_0\le t<t_{i_0,i_1}\\ i_1, & t=t_{i_0,i_1}.\end{cases}$$

Let $\mathbf{X}(t_0)\in\mathbb{R}^D$ denote a pre-specified initial condition for the original system, and suppose $t_{i_1,i_2}>t_{i_0,i_1}$ is the time of a second, subsequent boundary crossing. The global reduced system for $t_0\le t\le t_{i_1,i_2}$ may be defined as

$$\dot{\mathbf{x}}(t)=\begin{cases}f_{i_0}(\mathbf{x}),\ \mathbf{x}(t_0)=\Phi_{i_0}(\mathbf{X}(t_0)-c_{i_0}),\\ \qquad\qquad\qquad\text{for } t_0<t\le t_{i_0,i_1}\\ f_{i_1}(\mathbf{x}),\ \mathbf{x}(t_{i_0,i_1})\leftarrow\Phi_{i_1}\big(\Psi_{i_0}^*\mathbf{x}(t_{i_0,i_1})+c_{i_0}-c_{i_1}\big),\\ \qquad\qquad\qquad\text{for } t_{i_0,i_1}<t\le t_{i_1,i_2}.\end{cases}$$

The last state computed by system i_0, $\mathbf{x}(t_{i_0,i_1})$, serves as the initial condition for system i_1 only after projecting $\mathbf{x}(t_{i_0,i_1})$ onto plane i_1 via the affine transformation $\Phi_{i_1}\big(\Psi_{i_0}^*\mathbf{x}(t_{i_0,i_1})+c_{i_0}-c_{i_1}\big)$. The process above then continues for further boundary crossings, replacing the indices i_0,i_1,i_2 as appropriate.

B. Blending Local Systems

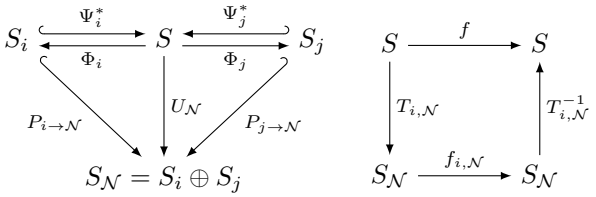
Transitions between local models can be smoothed by blending several systems in a neighborhood of a given boundary. However, trajectories and dynamics of the reduced systems are expressed in differing coordinate systems specific to the respective approximating planes, and cannot be averaged directly. This difficulty can be resolved by blending in a common space just large enough to include the planes associated to the collection of local systems participating in the blend. In the following, we will consider a blend

involving two local systems for simplicity, however the development readily applies more generally to an arbitrary number of systems.

Let $S \subseteq \mathbb{R}^D$ denote the high dimensional statespace, let $S_i \supseteq \text{range}\Phi_i$ denote the d_i dimensional subspace containing the statespace of the local dynamical system i, and let the blending neighborhood $\mathcal{N} = \mathcal{N}(t)$ consist of two arbitrary (but distinct) local systems i and j; $\mathcal{N} = \{i, j\}$. A plane in S describing the common subspace $S_\mathcal{N} = S_i \oplus S_j$ may be found via the SVD. Let

$$U_\mathcal{N}^* \Sigma_\mathcal{N} V_\mathcal{N} = \text{svd}\big([(\Phi_i^* + c_i \mathbf{1}_{d_i}^\top)\ (\Phi_j^* + c_j \mathbf{1}_{d_j}^\top)]\big). \qquad (8)$$

The common plane has dimension $d_\mathcal{N} \leq d_i + d_j$, center $c_\mathcal{N} = U_\mathcal{N}^* \mathbf{e}_1$, and orthogonal basis vectors $\{U_\mathcal{N}^* \mathbf{e}_k\}_{k=2}^{d_\mathcal{N}}$. Temporarily omitting the role of the plane centers c_i for simplicity, the following diagrams describe the situation:



The map $P_{i\to\mathcal{N}}$ is obtained from the diagram on the left as $P_{i\to\mathcal{N}} = U_\mathcal{N} \Psi_i^*$. Subsequently, its "reverse" $P_{\mathcal{N}\to i}$ is given by $P_{\mathcal{N}\to i} = \Phi_i U_\mathcal{N}^*$. A similar argument applies to $P_{j\to\mathcal{N}}$.

We can define a global reduced system valid for $t > 0$ as

$$\begin{cases} \dot{\mathbf{x}}(t) = \sum_{i\in\mathcal{N}} w_i(\mathbf{x}, t) P_{i\to\mathcal{N}} f_i(P_{\mathcal{N}\to i}\mathbf{x}) \\ \mathbf{y}(t) = \sum_{i\in\mathcal{N}} w_i(\mathbf{x}, t) h_i(P_{\mathcal{N}\to i}\mathbf{x}) \end{cases} \qquad (9)$$

where the weights w_i satisfy $w_i \geq 0, \sum_{i\in\mathcal{N}} w_i = 1$ for all $\mathbf{x}, t$. The right-hand diagram above exhibits the relationship of the blended system (9) to the original system on S. Letting $T_{i,\mathcal{N}} \triangleq U_\mathcal{N} \Psi_i^* \Phi_i$ and $T_{i,\mathcal{N}}^{-1} \triangleq \Psi_i^* \Phi_i U_\mathcal{N}^*$, we may write the contribution of system i to the reduced dynamics in $S_\mathcal{N}$ as $f_{i,\mathcal{N}}(x) = T_{i,\mathcal{N}} f(T_{i,\mathcal{N}}^{-1} x)$ using Equation (5).

The initial condition of (9) is $\mathbf{x}(0) = U_\mathcal{N}\mathbf{X}(0)$ if $\mathbf{X}(0)$ is a given initial condition for the high-dimensional system. Note that this gives the correct initial conditions for the local systems $\{f_i\}_{i\in\mathcal{N}}$ since $\mathbf{x}_i(0) = P_{\mathcal{N}\to i}\mathbf{x}(0) = \Phi_i U_\mathcal{N}^* U_\mathcal{N} \mathbf{X}(0) = \Phi_i \mathbf{X}(0)$ as the columns of Φ_i^* are already in the range of the projector $U_\mathcal{N}^* U_\mathcal{N}$ from (8) (still omitting the centers). When a new local model f_k is added to the blending neighborhood $\mathcal{N}$, its initial condition is similarly obtained from the current state of the system $\mathbf{x}(t) \in S_\mathcal{N}$ as $P_{\mathcal{N}\to k}\mathbf{x}(t)$. Local models may be added or dropped as needed as long as the operators $\{P_{i\to\mathcal{N}}\}_{i\in\mathcal{N}}$ are updated as necessary.

The interpolation weights w_i in (9) may be chosen in a variety of ways. Combining local systems on the basis of distance to the plane centers is a reasonable choice. The weights in this case may be defined as follows. Consider a discretization[2] $t_0 < \ldots < t_n < t_{n+1} < \ldots t_T$ of the simulation time interval. Given a collection of weights $w_i(t_n)$ at an arbitrary time t_n, the high-dimensional embedding into S of the low dimensional state $\mathbf{x}(t_n)$ evolving according to (9) is given by

$$\widetilde{\mathbf{X}}(t_n) = \sum_{i\in\mathcal{N}} w_i(t_n) T_{i,\mathcal{N}}^{-1} \mathbf{x}(t_n).$$

The set of weights at the next simulation time step can then be computed as

$$w_i(t_{n+1}) = \frac{K\big(\widetilde{\mathbf{X}}(t_n), c_i\big)}{\sum_{j\in\mathcal{N}} K\big(\widetilde{\mathbf{X}}(t_n), c_j\big)}, \qquad i \in \mathcal{N} \qquad (10)$$

where $K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}_+$ is a non-negative valued, positive definite, symmetric similarity function. The Gaussian kernel $K(x, x') = \exp(-\rho^2(x, x')/\sigma^2)$ is a natural choice (σ may be rather easily chosen based on the size of the local neighborhoods). The initial set of weights $w_i(0)$ may be computed by comparing the initial condition $\mathbf{X}(0)$ to the centers $\{c_i\}_i$ with K.

In practice, one could apply a threshold to the weights (10) so that only a small number of systems contribute to the interpolation, and only near transition regions. Although the global, reduced system is characterized by a collection of $|\mathsf{Leaf}|$ local systems (with in general different dimensions), at any given time the system can be entirely described by either a single low-dimensional system, or if desired, a small number of systems during transitions from one region of responsibility to another; *simulations always remain governed by low-dimensional dynamics*.

In the context of control, one can therefore design simple controllers responsible only for the respective local regions of the statespace associated to each local reduced model, and switch (or blend) between them as needed in practice. The controllers can be simple both because they are applied to low-dimensional systems and because they only need to be effective in a restricted, localized region of the statespace. Furthermore, the control problem is decomposed into nearly independent parts, so that if a complex controller is needed in one part of the statespace, the other controllers responsible for different parts do not necessarily need to be complex. Maintaining a multiscale collection of controllers also opens the door to interesting decentralized control schemes. Global control of a complex system may be effected by applying local control, thereby distributing the computational burden, reducing communication overhead and possibly bringing a controller closer to the relevant sensors and actuators[3].

V. Application to a Fluid Flow Problem

We illustrate the local POD algorithm discussed in section III-A in the context of an uncontrolled, nonlinear fluid flow simulation with statespace dimension $D = 3721$.

A. Problem Specification

We consider the two-dimensional unsteady, incompressible flow of fluid past a stationary, infinite cylinder. This is a well-studied problem in fluid mechanics (see e.g. [19]), and is related to flow past inclined plates and airfoils at various

[2]Chosen so that the movement in time of the system is small compared to the scale of change of the interpolation weights w_i.

[3]Consider control of coupled systems, as in flocking, or problems relying heavily on parallel computation where data-locality and easy parallelization with low communication overhead is critical.

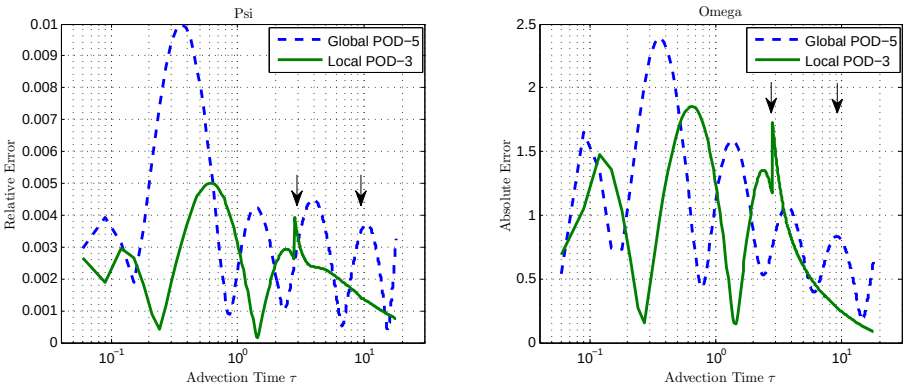


Fig. 1. Evolution of the simulation error for global POD vs. the local POD approach described in Section III-A: stream function ψ error (left) and vorticity ω error (right). Arrows indicate where switching between models occurred in the local POD simulation. See text for details.

angles of attack. Control of fluids flowing past obstacles is an active area of research, with applications in civil engineering and aerodynamics (see [20] for a review which includes applications of POD to fluids). We restrict our attention to a low-Reynolds number regime with $R_e = 40$, non-dimensionalized as $R_e \triangleq Ud_c/\nu$ where U is the free-stream velocity, d_c is the cylinder diameter and ν is the kinematic viscosity. At this Reynolds number the flow is stable and does not exhibit vortex shedding. The velocity $\mathbf{V} = [u(x,y)\ v(x,y)]^\top$ and pressure $p(x,y)$ of the fluid are governed by the unsteady Navier-Stokes equations, which for this problem can be reformulated in terms of vorticity $\boldsymbol{\omega} = \nabla \times \mathbf{V}$, and a stream function ψ defined by the relations $u = \partial_y \psi$ and $v = -\partial_x \psi$. Denoting by ω the z-component of $\boldsymbol{\omega}$, the system of PDEs governing the 2D flow can be expressed as

$$\partial_t \omega + \partial_y \psi \partial_x \omega - \partial_x \psi \partial_y \omega = R_e^{-1} \Delta \omega \qquad (11a)$$

$$\Delta \psi = -\omega. \qquad (11b)$$

See e.g. [21] for a detailed discussion of the stream function-vorticity formulation of 2D flow problems.

B. Simulation Protocol

We followed the general approach described in [21] (Sec. 4.6) to simulate the system (11). A mixture of explicit and implicit finite difference schemes were employed to solve the vorticity equation, and no-slip boundary conditions were enforced on the cylinder surface. The unbounded physical domain was mapped to a finite computational domain via an algebraic transformation. Symmetry of the solution in the absence of vortex shedding allows for simulation of only the top half of the problem, and we used a (61×61) collocated grid with identical spatial discretizations in the x and y-directions. Given the nondimensional advection time $\tau \triangleq tU/d_c$, the integration time step was set to $\delta\tau = 10^{-3}$.

Local POD was performed as follows. We first simulated the system from zero initial conditions for $0 \leq \tau \leq 20$, taking snapshots of the the vorticity $\omega(x,y)$ every 30 integration steps, giving $n = 665$ D-dimensional samples ($D = 3721$). A geometric multi-resolution tree decomposition was then computed, following the process described in Section II, to precision $\epsilon = 0.001$ with non-leaf node dimension (dimension of the PCA approximation) set to 3. This resulted in a tree with 5 nodes, of which 3 were leaf nodes. The dimension of the POD basis at all leaf nodes was also 3 (note that this is determined automatically based on ϵ and the dimension at the parent nodes – see [1] for details). Construction of the tree, including PCA basis vectors at each node, took on the order of a few seconds on a Linux PC equipped with a Xeon 5320 CPU running MATLAB. For comparison purposes, global, vanilla POD, with centering, was also computed on the above snapshot dataset, and the top 5 most energetic modes were selected.

Local reduced systems were then defined exactly as in (3), and simulated for $0 \leq \tau \leq 20$ (again from zero initial conditions) using the simple switching rule defined in Section IV-A. Parasitic oscillation between models near responsibility region boundary transitions was not found to be a problem.

C. Results

Let $\psi_{\text{red}}(\tau) \in \mathbb{R}^D$ denote a reduced model's stream function, embedded into the full statespace $\mathbb{R}^D$ via an appropriate $P^*_{j,k}$ at (advection) time τ, and let $\psi_{\text{true}}(\tau)$ denote the full model simulation stream function. Similarly, define the analogous vorticities $\omega_{\text{red}}(\tau)$, $\omega_{\text{true}}(\tau) \in \mathbb{R}^D$. The relative error

$$e_\psi(\tau) = \frac{\|\psi_{\text{red}}(\tau) - \psi_{\text{true}}(\tau)\|_2}{\|\psi_{\text{true}}(\tau)\|_2}$$

and absolute error

$$e_\omega(\tau) = \|\omega_{\text{red}}(\tau) - \omega_{\text{true}}(\tau)\|_2$$

for each of global (vanilla) POD and local POD were computed at each (respective) simulation time step. Figure 1 shows error signals for the stream functions ψ (left) and vorticity ω (right), over the duration of the simulation. The vorticity error was assessed on an absolute basis due to zero or nearly zero vorticity for the first second or so of this simulation. The time axis is given on a logarithmic scale for improved readability. Arrows mark the points in time where the local POD simulation switched between models (near $\tau = 2.8$ and $\tau = 9.2$). Abrupt switching between local models is the cause of the spike in error seen at $\tau = 2.8$ in the solid/green traces, however it is likely that this could be mitigated by blending the models around transition regions, as described in Section IV-B. Stream functions computed (by

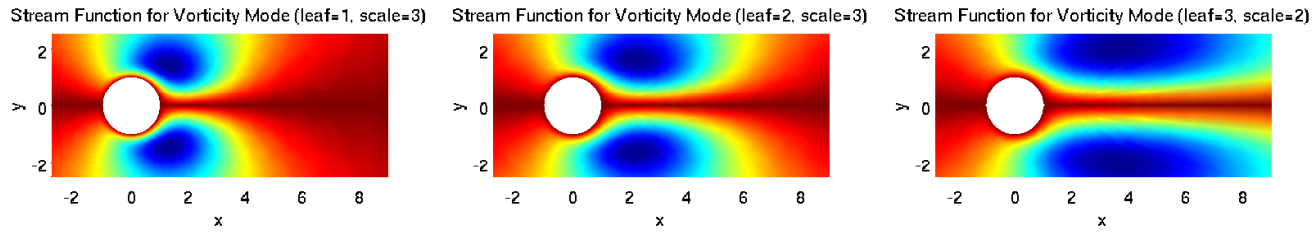


Fig. 2. Stream functions associated to the (respective) top local POD modes from each of the leaf nodes.

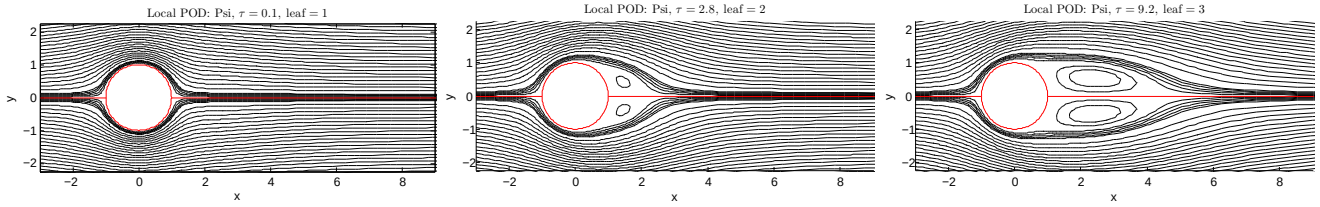


Fig. 3. Streamlines illustrating the local POD flow simulation near the beginning (left), after a transition from leaf node model 1 to 2 (middle), and after a transition from leaf node model 2 to 3 (right).

numerically solving (11b)) from the top local POD vorticity mode at each of the three leaf nodes (respectively) are shown in Figure 2. These modes correspond to clear temporal phases of the flow, suggesting that the tree has decomposed trajectories into temporally meaningful segments. Figure 3 shows contour plots of the local POD simulation's stream function ψ, in left to right order, near the start of the simulation, just after switching from leaf node model 1 to 2, and just after switching from leaf node model 2 to 3.

Although the global POD simulation (Figure 1, blue dashed traces) uses five components, the global POD error is in general seen to be higher than the switching local POD simulation (green solid traces), which at any given moment in time uses three components. Thus, for this particular problem, there is a computational savings of two projections over global POD during simulation. We ignore the cost of constructing the multiresolution sample tree in this comparison, as it is a one-time cost, and is typically small relative to the cost of simulating. As discussed above, the fact that simple local models support accurate simulations provides evidence that one may be able to fit low dimensional approximations to the projected dynamics locally, and simulate at a drastically reduced computational cost. The fact that the local models are of lower dimension than a global POD approximation also suggests that in a fluid control setting, simpler, localized controllers may be possible relative to global approaches. We seek to address these interesting topics in a more comprehensive, future publication.

REFERENCES

[1] W. K. Allard, G. Chen, and M. Maggioni, "Multi-scale geometric methods for data sets ii: Geometric multi-resolution analysis," *Appl. Comput. Harmon. Anal.*, vol. 32, pp. 435–462, 2012.

[2] G. E. Dullerud and F. Paganini, *A Course in Robust Control Theory: a Convex Approach*. Springer, 2000.

[3] A. Antoulas, *Approximation of large-scale dynamical systems*. Cambridge University Press, 2005.

[4] S. J. Lall, J. Marsden, and S. Glavaski, "A subspace approach to balanced truncation for model reduction of nonlinear control systems," *International Journal on Robust and Nonlinear Control*, vol. 12, pp. 519–535, 2002.

[5] I. Mezić, "Spectral properties of dynamical systems, model reduction and decompositions," *Nonlinear Dynam.*, vol. 41, no. 1-3, pp. 309–325, 2005.

[6] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, "Spectral analysis of nonlinear flows," *J. Fluid Mech.*, vol. 641, pp. 115–127, 2009.

[7] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *J. Fluid Mech.*, vol. 656, pp. 5–28, 2010.

[8] K. K. Chen, J. H. Tu, and C. W. Rowley, "Variants of dynamic mode decomposition: Boundary condition, koopman, and fourier analyses," *Journal of Nonlinear Science*, 2012. [published online; DOI: 10.1007/s00332-012-9130-9].

[9] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.

[10] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[11] M.-L. Rapún and J. Vega, "Reduced order models based on local POD plus Galerkin projection," *J. Comput. Phys.*, vol. 229, no. 8, pp. 3046–3063, 2010.

[12] M. Schlegel, B. R. Noack, P. Comte, D. Kolomenskiy, K. Schneider, M. Farge, D. M. Luchtenburg, J. E. Scouten, and G. Tadmor, "Reduced-order modelling of turbulent jets for noise control," in *Notes on Numerical Fluid Mechanics and Multidisciplinary Design (Eds. C. Brun et al.)*, vol. 104, pp. 3–27, Springer, 2009.

[13] S. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in *Proc. ICML*, pp. 97–104, 2006.

[14] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Sci. Comput.*, vol. 20, pp. 359–392, 1999.

[15] B. Moore, "Principal component analysis in linear systems: Controllability, observability and model reduction," *IEEE Tran. Automat. Control*, vol. 26, pp. 17–32, 1981.

[16] C. W. Rowley, "Model reduction for fluids using balanced proper orthogonal decomposition," *Internat. J. Bifur. Chaos Appl. Sci. Engrg.*, vol. 15, pp. 997–1013, 2005.

[17] L. N. Trefethen and M. Embree, *Spectra and Pseudospectra: The Behavior of Nonnormal Matrices and Operators*. Princeton University Press, 2005.

[18] Y. Saad, "Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices," *Linear Algebra Appl.*, vol. 34, pp. 269–295, 1980.

[19] C. H. K. Williamson, "Vortex dynamics in the cylinder wake," in *Annual review of fluid mechanics, Vol. 28*, pp. 477–539, Palo Alto, CA: Annual Reviews, 1996.

[20] H. Choi, W.-P. Jeon, and J. Kim, "Control of flow over a bluff body," in *Annual review of fluid mechanics. Vol. 40*, vol. 40 of *Annu. Rev. Fluid Mech.*, pp. 113–139, Palo Alto, CA: Annual Reviews, 2008.

[21] S. Biringen and C.-Y. Chow, *An Introduction to Computational Fluid Mechanics by Example*. Wiley, 2011.