Oden Institute REPORT 19-18

December 2019

Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems

by

Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, Karen Willcox

Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin
Austin, Texas 78712

Reference: Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, Karen Willcox, "Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems," Oden Institute REPORT 19-18, Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, December 2019.

Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems

Elizabeth Qian[a], Boris Kramer[b], Benjamin Peherstorfer[c], Karen Willcox[d]

[a] *Center for Computational Engineering, Massachusetts Institute of Technology*
[b] *Department of Mechanical and Aerospace Engineering, University of California San Diego*
[c] *Courant Institute of Mathematical Sciences, New York University*
[d] *Oden Institute for Computational Engineering and Sciences, University of Texas at Austin*

Abstract

We present *Lift & Learn*, a physics-informed method for learning low-dimensional models for large-scale dynamical systems. The method exploits knowledge of a system's governing equations to identify a coordinate transformation in which the system dynamics have quadratic structure. This transformation is called a lifting map because it often adds auxiliary variables to the system state. The lifting map is applied to data obtained by evaluating a model for the original nonlinear system. This lifted data is projected onto its leading principal components, and low-dimensional linear and quadratic matrix operators are fit to the lifted reduced data using a least-squares operator inference procedure. Analysis of our method shows that the Lift & Learn models are able to capture the system physics in the lifted coordinates at least as accurately as traditional intrusive model reduction approaches. This preservation of system physics makes the Lift & Learn models robust to changes in inputs. Numerical experiments on the FitzHugh-Nagumo neuron activation model and the compressible Euler equations demonstrate the generalizability of our model.

Keywords: data-driven model reduction, scientific machine learning, dynamical systems, partial differential equations, lifting map

1. Introduction

The derivation of low-dimensional models for high-dimensional dynamical systems from data is an important task that makes feasible many-query computational analyses like uncertainty propagation, optimization, and control. Traditional model reduction methods rely on full knowledge of the system governing equations as well as the ability to intrusively manipulate solver codes. In contrast, classical machine learning methods fit models to data while treating the solver as a black box, ignoring knowledge of the problem physics. In this paper, we propose a new hybrid machine learning-model reduction method called Lift & Learn, in which knowledge of the system governing equations is exploited to identify a set of lifted coordinates in which a low-dimensional model can be learned.

In projection-based model reduction, the system governing equations are projected onto a low-dimensional approximation subspace to obtain a reduced model. The basis for the reduced space is computed from data; a common choice is the Proper Orthogonal Decomposition (POD) basis, comprised of the leading principal components of the data [1, 2, 3, 4]. However, the projection process is intrusive, requiring access to the codes that implement the high-dimensional operators of the original equations. One way to avoid the intrusive projection is to learn a map from input parameters to coefficients of the reduced basis, e.g., via gappy POD [5, 6], or using radial basis functions [7], a neural network [8, 9, 10] or a nearest-neighbors method [10]. However, these approaches are agnostic to the dynamics of the system in that they do not model the evolution of the system state over time.

To learn models for the dynamics of a system, the work in [11] uses compressed sensing methods to learn sparse, high-dimensional matrix operators from data. Techniques from compressed sensing are also used in [12, 13] to fit terms of governing equations from a dictionary of possible nonlinear terms. However, these approaches do not reduce the dimension of the system. To learn a reduced model, dynamic mode decomposition (DMD) [14, 15] projects data onto a reduced basis and then fits to the reduced data a linear operator. Similarly, the operator inference approach of [16] fits linear and polynomial matrix operators to reduced data. However, if the true underlying dynamics of the system are non-polynomial, then the DMD and operator inference models may be inaccurate.

Several communities have explored using variable transformations to expose structure in a nonlinear system. In particular, Koopman operator the-

ory, which states that every nonlinear dynamical system can be exactly described by an infinite-dimensional linear operator that acts on scalar observables of the system [17], has been used to extend DMD to fit linear models for nonlinear systems in observable space defined by a dictionary [18], kernel [19], or dictionary learning method [20]. In contrast, *lifting* transformations [21] are maps derived for specific governing equations that yield a finite-dimensional coordinate representation in which the system dynamics are quadratic [22]. Lifting is exploited for model reduction in [22, 23, 24, 25], where the quadratic operators of a high-dimensional lifted model are projected onto a reduced space to obtain a quadratic reduced model. However, in many settings, it is impossible or impractical to explicitly derive a high-dimensional lifted model, so the only available model is the original nonlinear one.

In *Lift & Learn*, we use the available nonlinear model to learn quadratic reduced model operators for the lifted system without requiring a high-dimensional lifted model to be available. We collect state trajectory data by evaluating the original nonlinear model, lift the data, project the lifted data onto a low-dimensional basis, and fit reduced quadratic operators to the data using the operator inference procedure of [16]. In this way, we learn a quadratic model that respects the physics of the original nonlinear system. Our contributions are thus:

1. the use of lifting to explicitly parametrize the reduced model in a form that can be learned using the operator inference approach of [16],
2. the use of learning to non-intrusively obtain lifted reduced models from data generated by the original nonlinear model, enabling their use even in settings where lifted models are unavailable, and
3. the exploitation of the preservation of system physics to prove guarantees about the lifted model fit to data.

Our approach fits reduced quadratic operators to the data in state space. In the situation where frequency-domain data is available, the work in [26] fits quadratic operators to data in frequency space.

Section 2 describes projection-based model reduction for nonlinear systems. Section 3 defines the lifting map and its properties and introduces our Lift & Learn model learning method. Section 4 derives a bound on the mismatch between the data and the Lift & Learn model dynamics. In Section 5 we apply our method to two examples: the FitzHugh-Nagumo prototypical neuron activation model and the Euler fluid dynamics equations. Our

numerical results demonstrate the learned models' reliability and ability to generalize outside their training sets.

2. Projection-based model reduction

Let $\Omega \in \mathbb{R}^d$ denote a physical domain and let $[0, T_f]$ be a time domain for some final time $T_f > 0$. The nonlinear partial differential equation (PDE)

$$\frac{\partial \vec{s}}{\partial t} = \vec{f}\,(\vec{s}) \tag{1}$$

defines a dynamical system for the d_s-dimensional vector state field

$$\vec{s}(x,t) = \begin{pmatrix} s_1(x,t) \\ \vdots \\ s_{d_s}(x,t) \end{pmatrix}, \tag{2}$$

where $s_j : \Omega \times [0, T_f) \to \mathcal{S}_j \subset \mathbb{R}$, for $j = 1, 2, \ldots, d_s$, and where

$$\vec{f}(\vec{s}) = \begin{pmatrix} f_1(\vec{s}) \\ \vdots \\ f_{d_s}(\vec{s}) \end{pmatrix} \tag{3}$$

is a nonlinear function that maps the state field to its time derivative. We assume that any spatial derivatives of $\vec{s}$ appearing in $\vec{f}$ are well-defined. Denote by $\mathcal{S}$ the d_s-dimensional product space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_{d_s}$, so that $\vec{s}(x,t) \in \mathcal{S}$.

We consider a semidiscrete model where the spatially discretized state vector $\mathbf{s}(t) \in \mathbb{R}^{nd_s}$ corresponds to the values of the d_s state variables at some collection of spatial points $\{x_l \in \Omega\}_{l=1}^n$, e.g., in a finite difference discretization or a finite element setting with a nodal basis. Let $\mathcal{S}^n$ denote the product domain $\mathcal{S}^n = \Pi_{l=1}^n \mathcal{S}$. Then, the semi-discrete full model is given by a system of nd_s ordinary differential equations:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{f}(\mathbf{s}), \tag{4}$$

where $\mathbf{f} : \mathcal{S}^n \to \mathcal{S}^n$ is a Lipschitz continuous function that discretizes $\vec{f}$.

Projection-based model reduction seeks a low-dimensional approximation to eq. (4) to achieve computational speed-ups. Denote by $\mathbf{s}_k$ the state snapshot at time t_k, i.e., the solution of eq. (4) at time t_k. The state snapshot matrix, $\mathbf{S}$, collects K snapshots as

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 & \cdots & \mathbf{s}_K \end{bmatrix} \in \mathbb{R}^{nd_s \times K}. \tag{5}$$

Note that $\mathbf{S}$ can contain states from multiple full model evaluations, e.g., from different initial conditions. The singular value decomposition (SVD) of $\mathbf{S}$ is given by

$$\mathbf{S} = \mathbf{\Phi}\mathbf{\Xi}\mathbf{\Psi}^\top \tag{6}$$

where $\mathbf{\Phi} \in \mathbb{R}^{nd_s \times nd_s}$, $\mathbf{\Xi} \in \mathbb{R}^{nd_s \times nd_s}$, and $\mathbf{\Psi} \in \mathbb{R}^{nd_s \times K}$. The Proper Orthogonal Decomposition (POD) basis of size r is denoted by $\mathbf{\Phi}_r$ and defined by the leading r columns of $\mathbf{\Phi}$. The POD state approximation is $\mathbf{s} \approx \mathbf{\Phi}_r \hat{\mathbf{s}}$, where $\hat{\mathbf{s}} \in \mathbb{R}^r$ is the reduced state. The POD reduced model is defined by Galerkin projection:

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \mathbf{\Phi}_r^\top \mathbf{f}\left(\mathbf{\Phi}_r \hat{\mathbf{s}}\right). \tag{7}$$

When $\mathbf{f}$ has no particular structure, solving eq. (7) requires evaluating the nd_s-dimensional map $\mathbf{f}$, which is expensive. When the PDE contains only polynomial nonlinearities in the state, however, the POD-Galerkin model preserves this structure and can be efficiently evaluated without recourse to high-dimensional quantities. That is, let

$$\frac{\partial \vec{s}}{\partial t} = \vec{a}(\vec{s}) + \vec{h}(\vec{s}) \tag{8}$$

where $\vec{a}(\cdot)$ and $\vec{h}(\cdot)$ are linear and quadratic operators, respectively. The discretized full model can then be written using matrix operators as follows:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{A}\mathbf{s} + \mathbf{H}(\mathbf{s} \otimes \mathbf{s}), \tag{9}$$

where $\mathbf{A} \in \mathbb{R}^{nd_s \times nd_s}$ is a linear matrix operator, $\mathbf{H} \in \mathbb{R}^{nd_s \times n^2 d_s^2}$ is a matricized tensor operator, and $\otimes$ denotes the Kronecker product. The POD-Galerkin reduced model is then given by:

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{s}} + \hat{\mathbf{H}}(\hat{\mathbf{s}} \otimes \hat{\mathbf{s}}), \tag{10}$$

where

$$\hat{\mathbf{A}} = \mathbf{\Phi}_r^\top \mathbf{A} \mathbf{\Phi}_r, \quad \hat{\mathbf{H}} = \mathbf{\Phi}_r^\top \mathbf{H}(\mathbf{\Phi}_r \otimes \mathbf{\Phi}_r) \tag{11}$$

are reduced matrix operators. The cost of evaluating eq. (10) depends only on the reduced dimension r. However, in many cases, including in our setting, the high-dimensional operators $\mathbf{A}$ and $\mathbf{H}$ are not available, so the reduced matrix operators cannot be computed as in eq. (11) and must be obtained through other means.

3. Lift & Learn: Reliable, generalizable predictive models for nonlinear PDEs

Lift & Learn is a method for learning quadratic reduced models for dynamical systems governed by nonlinear PDEs. The method exposes structure in nonlinear PDEs by identifying a lifting transformation in which the PDE admits a quadratic representation. Non-quadratic state data is obtained by evaluating the original nonlinear model (eq. (4)) and the lifting transformation is applied to this data. Quadratic reduced model operators are then fit to the transformed data. The result is an efficiently evaluable quadratic reduced model for the original nonlinear PDE.

Section 3.1 introduces lifting transformations and describes their properties. Section 3.2 describes how lifted reduced state data are obtained from a full model simulation in the original non-polynomial system variables. Section 3.3 introduces the operator inference procedure of [16] used to learn the reduced model from the lifted data. Section 3.4 summarizes the Lift & Learn method.

3.1. Exposing structure via lifting transformations

We begin by defining lifting transformations.

Definition 1. *Define the lifting map,*

$$\mathcal{T} : \mathcal{S} \to \mathcal{W} \subset \mathbb{R}^{d_w}, \quad d_w \geq d_s, \tag{12}$$

and let $\vec{w}(x,t) = \mathcal{T}(\vec{s}(x,t))$. $\mathcal{T}$ *is a* quadratic lifting *of eq.* (1) *if the following conditions are met:*

1. *the map $\mathcal{T}$ is differentiable with respect to $\vec{s}$ with bounded derivative, i.e., if $\mathcal{J}(\vec{s})$ is the Jacobian of $\mathcal{T}$ with respect to $\vec{s}$, then*

$$\sup_{\vec{s}\in\mathcal{S}} \|\mathcal{J}(\vec{s})\| \leq c, \tag{13}$$

for some $c > 0$, and

2. *the lifted state $\vec{w}$ satisfies*

$$\frac{\partial \vec{w}}{\partial t} = \frac{\partial \mathcal{T}(\vec{s})}{\partial t} = \mathcal{J}(\vec{s})\vec{f}(\vec{s}) = \vec{a}(\vec{w}) + \vec{h}(\vec{w}), \tag{14}$$

where

$$\vec{a}(\vec{w}) = \begin{pmatrix} a_1(\vec{w}) \\ \vdots \\ a_{d_w}(\vec{w}) \end{pmatrix}, \qquad \vec{h}(\vec{w}) = \begin{pmatrix} h_1(\vec{w}) \\ \vdots \\ h_{d_w}(\vec{w}) \end{pmatrix}, \tag{15}$$

for some linear functions a_j and quadratic functions h_j, $j = 1, 2, \ldots, d_w$.

The d_w-dimensional vector field $\vec{w}(x,t)$ is called the lifted state *and eq. (14) is the lifted PDE.*

Note that this definition may be extended to PDEs that contain constant and input-dependent terms — see Section 5.1 for an example. Lifting maps that transform non-polynomial dynamics into higher-order polynomial dynamics are also possible, but we focus on the quadratic case in this work. We now define a reverse lifting map which embeds the lifted state in the original state space.

Definition 2. *Given a lifting map $\mathcal{T}$, a map*

$$\mathcal{T}^\dagger : \mathcal{W} \to \mathcal{S} \tag{16}$$

is called a reverse lifting map if it is differentiable with respect to $\vec{w}$ with bounded derivative on $\mathcal{W}$ and satisfies $\mathcal{T}^\dagger(\mathcal{T}(\vec{s})) = \vec{s}$ for all $\vec{s} \in \mathcal{S}$.

We now present a simple illustrative example of lifting.

Example 1. *Consider the nonlinear PDE,*

$$\frac{\partial s}{\partial t} = -e^s. \tag{17}$$

To lift eq. (17)*, we define the auxiliary variable* $-e^s$*. That is, the lifting map and its reverse map are given by*

$$\mathcal{T} : s \mapsto \begin{pmatrix} s \\ -e^s \end{pmatrix} \equiv \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \vec{w}, \qquad \mathcal{T}^\dagger : \vec{w} \mapsto w_1, \tag{18}$$

so that the lifted system is quadratic:

$$\frac{\partial}{\partial t}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -e^s \end{pmatrix}\frac{\partial s}{\partial t} = \begin{pmatrix} 1 \\ -e^s \end{pmatrix}(-e^s) = \begin{pmatrix} w_2 \\ (w_2)^2 \end{pmatrix}. \tag{19}$$

The lifting map must be specifically derived for the problem at hand. One strategy for doing so is to introduce auxiliary variables for the non-quadratic terms in the governing PDE and augment the system with evolution equations for these auxiliary variables [22]. The work in [22] shows that a large class of nonlinear terms which appear in engineering systems may be lifted to quadratic form in this way with $\mathcal{O}(d_w) = \mathcal{O}(d_s)$, including monomial, sinusoidal, and exponential terms. In [23], this strategy is used to lift PDEs that govern systems in neuron modeling and combustion to quadratic form. Note that $\mathcal{T}$ is generally non-unique (consider $w_2 = e^s$ in Equation (19)), and for a given $\mathcal{T}$, the reverse lifting map is also non-unique.

3.2. Lifted training data

This section presents a method for obtaining data in the lifted variables from the available non-lifted model (eq. (4)).

State data. We first collect original snapshot data by simulating the original full model eq. (4). Then, for each column of the data matrix (eq. (5)), we apply the lifting map node-wise to the discrete state to obtain lifted snapshot data. That is, the lifted data matrix $\mathbf{W} \in \mathbb{R}^{nd_w \times K}$ is given by

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 & \cdots & \mathbf{w}_K \end{bmatrix} = \begin{bmatrix} \mathbf{T}(\mathbf{s}_1) & \cdots & \mathbf{T}(\mathbf{s}_K) \end{bmatrix}, \tag{20}$$

where $\mathbf{T}$ denotes the discrete lifting map defined by applying $\mathcal{T}$ node-wise to each spatial node.

We denote the SVD of the transformed data by $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, with $\mathbf{U} \in \mathbb{R}^{nd_w \times nd_w}$, $\mathbf{\Sigma} \in \mathbb{R}^{nd_w \times nd_w}$, $\mathbf{V} \in \mathbb{R}^{nd_w \times K}$. The r-dimensional POD basis matrix is given by the leading r columns of $\mathbf{U}$, denoted $\mathbf{U}_r \in \mathbb{R}^{nd_w \times r}$. Projection onto $\mathbf{U}_r$ yields state data in the reduced space:

$$\hat{\mathbf{W}} = \mathbf{U}_r^\top \mathbf{W} = \begin{bmatrix} \hat{\mathbf{w}}_1 & \cdots & \hat{\mathbf{w}}_K \end{bmatrix} \in \mathbb{R}^{r \times K}. \tag{21}$$

Time derivative data. To learn the matrix operators of a lifted reduced model of the form in eq. (10), reduced state time derivative data are also required. To obtain data for dynamics that are Markovian in the reduced state, we adapt the procedure in [27] to the Lift & Learn setting. For each $\hat{\mathbf{w}}_k$,

$$\mathbf{w}_{\text{proj},k} = \mathbf{U}_r \hat{\mathbf{w}}_k = \mathbf{U}_r \mathbf{U}_r^\top \mathbf{w}_k \in \mathbb{R}^{nd_w} \tag{22}$$

denotes the projection of the lifted state onto the subspace spanned by the POD basis. Denote by $\mathbf{T}^\dagger$ the discrete reverse lifting defined by applying $\mathcal{T}^\dagger$ node-wise for each spatial node. We assume that $\mathbf{U}_r$ is a sufficiently rich basis that $\mathbf{T}^\dagger$ is well-defined for $\mathbf{w}_{\text{proj},k}$, and reverse the lifting for all k to obtain a discrete non-lifted state that corresponds to the projected lifted state:

$$\mathbf{s}_{\text{proj},k} = \mathbf{T}^\dagger(\mathbf{w}_{\text{proj},k}). \tag{23}$$

We then evaluate the available nonlinear full model to obtain new time derivative data, which we denote $\mathbf{s}'_{\text{proj},k}$:

$$\mathbf{s}'_{\text{proj},k} = \mathbf{f}\left(\mathbf{T}^\dagger\left(\mathbf{w}_{\text{proj},k}\right)\right). \tag{24}$$

Let $\mathbf{J}(\mathbf{s}) \in \mathbb{R}^{nd_w \times nd_s}$ denote the Jacobian of $\mathbf{T}$ with respect to $\mathbf{s}$. Applying the chain rule to eq. (24) yields a time derivative in the discrete lifted state:

$$\mathbf{w}'_{\text{proj},k} = \mathbf{J}(\mathbf{s}_{\text{proj},k})\mathbf{s}'_{\text{proj},k}. \tag{25}$$

Time derivative data in the reduced space is then obtained by projecting eq. (25) back onto the POD basis:

$$\hat{\mathbf{w}}'_k = \mathbf{U}_r^\top \mathbf{J}(\mathbf{s}_{\text{proj},k})\mathbf{s}'_{\text{proj},k} = \mathbf{U}_r^\top \left(\mathbf{J}(\mathbf{T}^\dagger(\mathbf{U}_r\mathbf{U}_r^\top \mathbf{w}_k))\mathbf{f}\left(\mathbf{T}^\dagger(\mathbf{U}_r\mathbf{U}_r^\top \mathbf{w}_k)\right)\right). \tag{26}$$

The reduced time derivative data matrix collects this data:

$$\hat{\mathbf{W}}' = \begin{bmatrix} \hat{\mathbf{w}}'_1 & \cdots & \hat{\mathbf{w}}'_K \end{bmatrix}. \tag{27}$$

3.3. Least-squares operator inference procedure

Given K reduced state snapshots (eq. (21)) and the corresponding reduced time derivative data (eq. (27)) and a postulated model form (eq. (10)), operator inference [16] formulates the following minimization problem for learning the matrix operators of eq. (10):

$$\min_{\hat{\mathbf{A}}\in\mathbb{R}^{r\times r}, \hat{\mathbf{H}}\in\mathbb{R}^{r\times r^2}} \frac{1}{K}\left\| \hat{\mathbf{W}}^\top \hat{\mathbf{A}}^\top + \left(\hat{\mathbf{W}} \tilde{\otimes} \hat{\mathbf{W}}\right)^\top \hat{\mathbf{H}}^\top - \hat{\mathbf{W}}'^\top \right\|_F^2, \tag{28}$$

where $\tilde{\otimes}$ denotes the column-wise Kronecker product, also known as the Khatri-Rao product. Each column of $\hat{\mathbf{W}}^\top$, $\hat{\mathbf{W}}'^\top$, and $(\hat{\mathbf{W}}\tilde{\otimes}\hat{\mathbf{W}})^\top$ corresponds to a single component of the reduced state $\hat{\mathbf{w}}$, yielding r independent least-squares problems which each define one row of $\hat{\mathbf{A}}$ and $\hat{\mathbf{H}}$. Each least-squares problem has $r + \frac{r(r+1)}{2}$ degrees of freedom when the structural redundancy of the Kronecker product is accounted for.

In practice, to solve eq. (28), we form a data matrix,

$$\hat{\mathbf{D}} = \begin{pmatrix} \hat{\mathbf{W}}^\top & \hat{\mathbf{W}}_{\text{sq}}^\top \end{pmatrix} \in \mathbb{R}^{K \times \left(r + \frac{r(r+1)}{2}\right)}, \tag{29}$$

where $\hat{\mathbf{W}}_{\text{sq}} \in \mathbb{R}^{\frac{r(r+1)}{2} \times r}$ contains quadratic data with the redundant cross terms of the Kronecker product removed. We then solve the least-squares equation,

$$\hat{\mathbf{D}} \begin{pmatrix} \hat{\mathbf{A}}^\top \\ \hat{\mathbf{H}}_{\text{sq}}^\top \end{pmatrix} = \hat{\mathbf{W}}', \tag{30}$$

where $\hat{\mathbf{H}}_{\text{sq}} \in \mathbb{R}^{r \times \frac{r(r+1)}{2}}$ contains coefficients of quadratic terms without redundancy, and we reconstruct the symmetric tensor $\hat{\mathbf{H}} \in \mathbb{R}^{r \times r^2}$ by splitting the coefficients for quadratic cross terms in $\hat{\mathbf{H}}_{\text{sq}}$ across the redundant terms. As long as $\hat{\mathbf{D}}$ has full column rank, eq. (30) has a unique solution [28].

The Lift & Learn reduced model is then given by

$$\frac{\mathrm{d}\hat{\mathbf{w}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{w}} + \hat{\mathbf{H}}(\hat{\mathbf{w}} \otimes \hat{\mathbf{w}}). \tag{31}$$

Note that the linear-quadratic case is considered for simplicity and concreteness, but the operator inference procedure can be flexibly adapted to learn reduced models with constant, input-dependent, and higher-order polynomial terms. Section 5.1 contains an example in which input operators are learned.

3.4. Lift & Learn: Method summary

The Lift & Learn method is summarized in Algorithm 1. The first step is to identify an appropriate lifting transformation as described in Section 3.1. The available full model in the original non-lifted variables is then used to obtain lifted reduced state data as described in Section 3.2. The operator inference framework in Section 3.3 is then employed to learn a quadratic

Algorithm 1 Lift & Learn

1: Use knowledge of governing PDE to identify lifting map $\mathcal{T}$ as in Section 3.1.
2: Evaluate non-quadratic full model (eq. (4)) to obtain state data in the original, non-lifted variables, $\mathbf{S}$.
3: Use $\mathbf{T}$ (defined by $\mathcal{T}$) to transform nonlinear state data $\mathbf{S}$ to lifted data $\mathbf{W}$.
4: Compute POD basis $\mathbf{U}_r$ for the lifted state data.
5: Project to obtain reduced lifted state data (eq. (21)) and reduced lifted time derivative data (eq. (27)) as in Section 3.2.
6: Solve least-squares minimization (eq. (28)) using lifted data to infer operators $\hat{\mathbf{A}}$ and $\hat{\mathbf{H}}$.

reduced model. Note that although this paper has primarily considered the case where the governing physics are described by nonlinear PDEs, the approach applies equally to systems where the governing equations are high-dimensional nonlinear ODEs.

4. Finite difference analysis

This section presents analysis of the Lift & Learn method in the setting where the full model arises from a consistent finite difference discretization of the original nonlinear PDE. Section 4.1 presents the setting of our analysis and Section 4.2 proves an upper bound on the residual of the Lift & Learn minimization.

4.1. Consistent finite difference models

We now provide a local error analysis for the setting where eq. (4) arises from a consistent finite-difference discretization of the state. Let $\bar{\mathbf{s}}$ denote

the vector of original state values at the grid points $\{x_l\}_{l=1}^n$:

$$\bar{\mathbf{s}}(t) = \begin{pmatrix} s_1(x_1,t) \\ \vdots \\ s_1(x_n,t) \\ \vdots \\ s_{d_s}(x_1,t) \\ \vdots \\ s_{d_s}(x_n,t) \end{pmatrix}. \tag{32}$$

Note that $\bar{\mathbf{s}}(t)$ differs from $\mathbf{s}(t)$ in that $\bar{\mathbf{s}}(t)$ is the exact continuous (strong) solution of the original PDE (eq. (1)) evaluated at the grid points, and $\mathbf{s}(t)$ is the semi-discrete solution of the spatially discrete set of ODEs (eq. (4)). If eq. (4) is a consistent order-p discretization of eq. (1), then for any given n there exists a constant $c_s > 0$ such that

$$\left| \mathbf{f}_{(j-1)n+l}(\bar{\mathbf{s}}(t)) - f_j(\vec{s}(x,t))\big|_{x=x_l} \right| \leq c_s n^{-p}, \quad \forall t, \tag{33}$$

for $j = 1, 2, \ldots, d_s$ and $l = 1, 2, \ldots, n$, where f_j is defined as in eq. (3) and $\mathbf{f}_{(j-1)n+l}$ denotes the $((j-1)n+l)$-th entry of the vector-valued nonlinear function $\mathbf{f}$, which corresponds to the time derivative of $s_j(x_l, t)$.

Now, let $\bar{\mathbf{w}}$ denote the discretization of the exact continuous lifted state on the same spatial grid:

$$\bar{\mathbf{w}}(t) = \mathbf{T}(\bar{\mathbf{s}}(t)) = \begin{pmatrix} \mathcal{T}_1(\vec{s}(x_1,t)) \\ \vdots \\ \mathcal{T}_1(\vec{s}(x_n,t)) \\ \vdots \\ \mathcal{T}_{d_w}(\vec{s}(x_1,t)) \\ \vdots \\ \mathcal{T}_{d_w}(\vec{s}(x_n,t)) \end{pmatrix}. \tag{34}$$

Then,

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mathbf{A}\mathbf{w} + \mathbf{H}(\mathbf{w} \otimes \mathbf{w}), \quad \mathbf{A} \in \mathbb{R}^{nd_w \times nd_w}, \, \mathbf{H} \in \mathbb{R}^{nd_w \times n^2 d_w^2} \tag{35}$$

is a consistent order-p discretization of eq. (14) if for any given n there exists a constant $c_w > 0$ such that, for all t,

$$\left| \mathbf{A}_{(j-1)n+l,:}\bar{\mathbf{w}} + \mathbf{H}_{(j-1)n+l,:}(\bar{\mathbf{w}} \otimes \bar{\mathbf{w}}) - \left(\vec{a}_j(\vec{w}) + \vec{h}_j(\vec{w}) \Big|_{x=x_l} \right| \leq c_w n^{-p}, \quad (36)$$

for $j = 1, 2, \ldots, d_w$ and $l = 1, 2, \ldots, n$, where $\mathbf{A}_{(j-1)n+l,:}$ and $\mathbf{H}_{(j-1)n+l,:}$ denote the $((j-1)n+l)$-th rows of $\mathbf{A}$ and $\mathbf{H}$, respectively, which correspond to the time derivative of the j-th lifted state at the l-th spatial node.

We note that in Lift & Learn, we assume that the discretized nonlinear model (eq. (4)) is available, and that it is stable. In contrast, for the lifted system, we assume only that the discrete operators $\mathbf{A}$ and $\mathbf{H}$ for a consistent discretization exist — they are used for analysis only, and availability is not required.

4.2. Theoretical results

We now prove that the minimum achieved by the Lift & Learn model in eq. (28) is bounded above by the objective value achieved by the intrusive lifted reduced model. This demonstrates two advantages of our method:

1. our non-intrusively obtained model is able to model the data at least as well as an intrusive reduced model, and
2. unlike many other learning methods, the quadratic form of the model we learn respects the model physics, enabling us to put an upper bound on the residual of the Lift & Learn model on the training data.

We begin with a consistency result.

Lemma 1. *If $\bar{\mathbf{w}}(t)$ is defined as in eq. (34), and if eq. (4) and eq. (35) are consistent discretizations of eq. (1) and eq. (14), respectively, then*

$$\left\| \mathbf{A}\bar{\mathbf{w}} + \mathbf{H}(\bar{\mathbf{w}} \otimes \bar{\mathbf{w}}) - \mathbf{J}(\mathbf{T}^\dagger(\bar{\mathbf{w}}))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}})) \right\|_2 \lesssim n^{\frac{1}{2}-p}, \quad \forall t, \quad (37)$$

where $\lesssim$ denotes a bound up to a constant independent of n.

Proof. Because $\mathbf{T}$ is defined by applying $\mathcal{T}$ node-wise, the $((j-1)n+l)$-th row of $\mathbf{J}$ contains partial derivatives of the j-th lifted state at the l-th spatial node with respect to $\mathbf{s}$. This corresponds to the j-th row of $\mathcal{J}$ evaluated at

l-th spatial node. That is, for all t,

$$\mathbf{J}_{(j-1)n+l,:}(\bar{\mathbf{s}})\mathbf{f}(\bar{\mathbf{s}}) = \sum_{i=1}^{d_w} \mathbf{J}_{(j-1)n+l\,,\,(i-1)n+l}\mathbf{f}_{(i-1)n+l}(\bar{\mathbf{s}}) \tag{38}$$

$$= \sum_{i=1}^{d_w} \mathcal{J}_{j,i}(\vec{s}(x_l))\mathbf{f}_{(i-1)n+l}(\bar{\mathbf{s}}), \tag{39}$$

for all $j = 1, 2, \ldots, d_w$, $l = 1, 2, \ldots, n$. Then, for all t,

$$\left|\mathbf{J}_{(j-1)n+l,:}(\bar{\mathbf{s}})\mathbf{f}(\bar{\mathbf{s}}) - \mathcal{J}_j(\vec{s}(x_l))\vec{f}(\vec{s})\big)\big|_{x=x_l}\right|$$

$$= \left|\sum_{i=1}^{d_w}\Big(\mathcal{J}_{j,i}(\vec{s}(x_l))\mathbf{f}_{(i-1)n+l}(\bar{\mathbf{s}}) - \mathcal{J}_{j,i}(\vec{s}(x_l))\vec{f}_i(\vec{s})|_{x=x_l}\Big)\right| \tag{40}$$

$$\leq \sum_{i=1}^{d_w}\big|\mathcal{J}_{j,i}(\vec{s}(x_l))\big| \cdot \big|\mathbf{f}_{(i-1)n+l}(\bar{\mathbf{s}}) - \vec{f}_i(\vec{s})|_{x=x_l}\big|. \tag{41}$$

Since $\mathcal{T}$ is has bounded derivative, $\mathcal{J}$ is bounded. Then, because $\mathbf{f}$ is consistent, we have, for all t,

$$\left|\mathbf{J}_{(j-1)n+l,:}(\bar{\mathbf{s}})\mathbf{f}(\bar{\mathbf{s}}) - \mathcal{J}_j(\vec{s}(x_l))\big(\vec{f}(\vec{s})\big)\big|_{x=x_l}\right| \lesssim n^{-p}. \tag{42}$$

By definition, the lifted dynamics are exact for $\vec{w} = \mathcal{T}(\vec{s})$, so for all j, k, and t,

$$\mathcal{J}_j(\vec{s}(x_l))\big(\vec{f}(\vec{s})\big)\big|_{x=x_l} = \Big(\vec{a}_j(\vec{w}) + \vec{h}_j(\vec{w})\Big)\Big|_{x=x_l}. \tag{43}$$

Since $\bar{\mathbf{s}} = \mathbf{T}^\dagger(\bar{\mathbf{w}})$, we can use the triangle inequality to combine eqs. (36), (42) and (43) as follows:

$$\big|\mathbf{A}_{(j-1)n+l,:}\bar{\mathbf{w}} + \mathbf{H}_{(j-1)n+l,:}(\bar{\mathbf{w}} \otimes \bar{\mathbf{w}}) - \mathbf{J}_{(j-1)n+l,:}(\mathbf{T}^\dagger(\bar{\mathbf{w}}))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}}))\big| \lesssim n^{-p} \tag{44}$$

for all $j = 1, 2, \ldots, d_w$, $l = 1, 2, \ldots, n$, and t. Since there are nd_w discrete lifted states, we have for all t the following bound (up to a constant):

$$\big\|\mathbf{A}\bar{\mathbf{w}} + \mathbf{H}(\bar{\mathbf{w}} \otimes \bar{\mathbf{w}}) - \mathbf{J}(\mathbf{T}^\dagger(\bar{\mathbf{w}}))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}}))\big\|_2 \lesssim n^{-(p-\frac{1}{2})}. \tag{45}$$

□

Lemma 1 holds for any state $\bar{\mathbf{w}} = \mathbf{T}(\bar{\mathbf{s}})$ for which the quadratic lifted dynamics of the corresponding $\vec{w}$ are exactly equivalent to the non-lifted dynamics (eq. (43)). Let $\{\bar{\mathbf{w}}_k\}_{k=1}^K$ be a collection of snapshots of these exact lifted states and suppose that $\mathbf{U}_r$ is an r-dimensional POD basis for the $\bar{\mathbf{w}}_k$. We now prove an analogous bound for the projected snapshots $\mathbf{U}_r\mathbf{U}_r^\top\bar{\mathbf{w}}_k$:

Lemma 2. *Assume* $\mathbf{J}$ *is also Lipschitz and define* $\mathbf{P}_r = \mathbf{U}_r\mathbf{U}_r^\top$. *Then, there exist constants* C_0, C_1, *and* C_2 *such that for all* k

$$\begin{aligned}\left\|\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k \otimes \mathbf{P}_r\bar{\mathbf{w}}_k) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right\|_2 \\ \leq C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\|, \quad (46)\end{aligned}$$

where $\mathbf{I}_{nd_w}$ *is the identity matrix of dimension* nd_w.

Proof. We begin by breaking the left side of eq. (46) into the following terms using the triangle inequality

$$\begin{aligned}\left\|\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k \otimes \mathbf{P}_r\bar{\mathbf{w}}_k) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right\| \\ \leq \left\|\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k \otimes \mathbf{P}_r\bar{\mathbf{w}}_k) - \left(\mathbf{A}\bar{\mathbf{w}}_k + \mathbf{H}(\bar{\mathbf{w}}_k \otimes \bar{\mathbf{w}}_k)\right)\right\| \\ + \left\|\mathbf{A}\bar{\mathbf{w}}_k + \mathbf{H}(\bar{\mathbf{w}}_k \otimes \bar{\mathbf{w}}_k) - \mathbf{J}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k))\right\| \\ + \left\|\mathbf{J}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k)) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right\|. \quad (47)\end{aligned}$$

The middle term can be bounded by Lemma 1. For the first term, define $\mathbf{Q}(\mathbf{w}) = \mathbf{A}\mathbf{w} + \mathbf{H}(\mathbf{w} \otimes \mathbf{w})$. Note that since $\mathbf{Q}$ is polynomial, it is Lipschitz over the finite set of snapshots, so there exists a constant C_1 such that

$$\left\|\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k \otimes \mathbf{P}_r\bar{\mathbf{w}}_k) - \left(\mathbf{A}\bar{\mathbf{w}}_k + \mathbf{H}(\bar{\mathbf{w}}_k \otimes \bar{\mathbf{w}}_k)\right)\right\| \leq C_1\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\|. \quad (48)$$

For the third term on the right side of eq. (47), since $\mathbf{J}$ and $\mathbf{f}$ are Lipschitz, the function $\mathbf{J}(\mathbf{T}^\dagger(\cdot))\mathbf{f}(\mathbf{T}^\dagger(\cdot))$ is also Lipschitz over the finite set of snapshots. Then, for some constant C_2,

$$\left\|\mathbf{J}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\bar{\mathbf{w}}_k)) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right\| \leq C_2\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\|. \quad (49)$$

Combining eqs. (48) and (49) with Lemma 1 yields the result. □

This result lets us upper-bound the residual of the Lift & Learn minimization (eq. (28)).

Theorem 1. *Let σ_i be the singular values of the snapshot data matrix $\mathbf{W}$. Then, the residual of the Lift & Learn operator inference is bounded as follows:*

$$\min_{\hat{\mathbf{A}}\in\mathbb{R}^{r\times r},\hat{\mathbf{H}}\in\mathbb{R}^{r\times r^2}} \frac{1}{K}\left\|\hat{\mathbf{W}}^\top\hat{\mathbf{A}}^\top + \left(\hat{\mathbf{W}}\otimes\hat{\mathbf{W}}\right)^\top\hat{\mathbf{H}}^\top - \hat{\mathbf{W}}'^\top\right\|_F^2 \\ \left(C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\varepsilon\right)^2, \quad (50)$$

where $\varepsilon^2 = \sum_{i=r+1}^{nd_w}\sigma_i^2$.

Proof. Apply Lemma 2 to each projected state snapshot $\mathbf{P}_r\bar{\mathbf{w}}_k$:

$$\left\|\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k\otimes\mathbf{P}_r\bar{\mathbf{w}}_k) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right\|_2 \\ < C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\|. \quad (51)$$

Then, for each snapshot,

$$\left\|\mathbf{U}_r^\top\left(\mathbf{A}\mathbf{P}_r\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{P}_r\bar{\mathbf{w}}_k\otimes\mathbf{P}_r\bar{\mathbf{w}}_k) - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right)\right\|_2 \\ \leq C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\| \quad (52)$$

since $\|\mathbf{U}_r\|_2 = 1$ because its columns are orthonormal vectors. Since the snapshots in the Lift & Learn data are used to compute the POD basis,

$$\|(\mathbf{P}_r - \mathbf{I}_{nd_w})\bar{\mathbf{w}}_k\| \leq \varepsilon, \quad (53)$$

for all snapshots $\bar{\mathbf{w}}_k$. Thus, taking the mean of the square over all snapshots,

$$\frac{1}{K}\sum_{k=1}^{K}\left\|\mathbf{U}_r^\top\left(\mathbf{A}\mathbf{U}_r\mathbf{U}_r^\top\bar{\mathbf{w}}_k + \mathbf{H}(\mathbf{U}_r\mathbf{U}_r^\top\bar{\mathbf{w}}_k\otimes\mathbf{U}_r\mathbf{U}_r^\top\bar{\mathbf{w}}_k)\right.\right. \\ \left.\left. - \mathbf{J}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\mathbf{f}(\mathbf{T}^\dagger(\mathbf{P}_r\bar{\mathbf{w}}_k))\right)\right\|_2^2 \leq \left(C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\varepsilon\right)^2. \quad (54)$$

Note that this sum is exactly the objective function in eq. (28). Thus, choosing $\hat{\mathbf{A}} = \mathbf{U}_r^\top\mathbf{A}\mathbf{U}_r$ and $\hat{\mathbf{H}} = \mathbf{U}_r^\top\mathbf{H}(\mathbf{U}_r\otimes\mathbf{U}_r)$ in eq. (28) would yield an objective value less than $\left(C_0 n^{\frac{1}{2}-p} + (C_1+C_2)\varepsilon\right)^2$, so the minimizer must be bounded by this value as well. □

The least-squares residual is a measure of the mismatch between the postulated quadratic model form and the true dynamics of the system. Theorem 1 shows that this mismatch has an upper bound dependent on two factors: the truncation error of the full model and the projection error of the POD basis.

5. Results

In this section, the Lift & Learn method is applied to two different dynamical systems. Section 5.1 considers the FitzHugh-Nagumo prototypical neuron activation model and Section 5.2 considers the Euler equations for inviscid fluid flow.

5.1. Application to the FitzHugh-Nagumo system

The FitzHugh-Nagumo system was first proposed in 1961 [29] and then realized as a circuit for electronic pulse transmission in [30]. The system has been used to study exciteable oscillatory dynamics such as those found in cardiac [31] and neuron modeling [32], and has become a benchmark problem in nonlinear model reduction [33]. Section 5.1.1 introduces the FitzHugh-Nagumo equations and lifts the system to quadratic form. Section 5.1.2 describes the data used for learning and the performance of the model on training and test sets.

5.1.1. FitzHugh-Nagumo problem statement and lifting transformation

The FitzHugh-Nagumo equations are a simplified neuron activation model with two states: s_1 represents voltage and s_2 represents voltage recovery. We consider the equations using the same parameters as in [33]:

$$\gamma \frac{\partial s_1}{\partial t} = \gamma^2 \frac{\partial^2 s_1}{\partial x^2} - s_1^3 + 1.1 s_1^2 - 0.1 s_1 + s_2 + 0.05, \tag{55a}$$

$$\frac{\partial s_2}{\partial t} = 0.5 s_1 - 2 s_2 + 0.05, \tag{55b}$$

where $\gamma = 0.015$. The full model solves the system eq. (55) on the spatial domain $x \in [0, 1]$ for the timeframe $t \in [0, 4]$. At $t = 0$, the states s_1 and s_2 are both zero everywhere, and the boundary conditions are given by

$$\left.\frac{\partial s_1}{\partial x}\right|_{x=0} = g(t), \qquad \left.\frac{\partial s_1}{\partial x}\right|_{x=1} = 0, \tag{56}$$

where $g(t)$ is a time-dependent input which represents a neuron stimulus. Because eq. (55a) contains a cubic nonlinear term, the model is lifted to quadratic form. Although the operator inference framework developed in [16] can, in principle, learn cubic and higher-order tensor terms, the number of unknowns in the learning problem can increase exponentially as the polynomial order of the model is increased, if the higher-order operators are dense.

The following lifting map $\mathcal{T}$ is used in [24, 23] to lift the system to quadratic form:

$$\mathcal{T} : \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \longmapsto \begin{pmatrix} s_1 \\ s_2 \\ (s_1)^2 \end{pmatrix} \equiv \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}. \tag{57}$$

The lifted system is then given by

$$\gamma \frac{\partial w_1}{\partial t} = \gamma^2 \frac{\partial^2 w_1}{\partial x^2} - w_1 w_3 + 1.1(w_1)^2 - 0.1 w_1 + w_2 + 0.05, \tag{58a}$$

$$\frac{\partial w_2}{\partial t} = 0.5 w_1 - 2 w_2 + 0.05, \tag{58b}$$

$$\begin{aligned} \frac{\partial w_3}{\partial t} &= 2 w_1 \frac{\partial w_1}{\partial t} \\ &= \frac{2}{\gamma} \left(\gamma^2 w_1 \frac{\partial^2 w_1}{\partial x^2} - w_3^2 + 1.1 w_1 w_3 - 0.1 w_3 + w_1 w_2 + 0.05 w_1 \right). \end{aligned} \tag{58c}$$

To be consistent with the prescribed initial and boundary conditions for the original state, w_3 must be zero everywhere at $t = 0$ and its boundary conditions are given by

$$\left. \frac{\partial w_3}{\partial x} \right|_{x=0} = 2 w_1 \left. \frac{\partial w_1}{\partial x} \right|_{x=0} = 2 w_1 g(t), \qquad \left. \frac{\partial w_3}{\partial x} \right|_{x=1} = 2 w_1 \left. \frac{\partial w_1}{\partial x} \right|_{x=L} = 0. \tag{59}$$

The lifted system in eq. (58) contains only quadratic nonlinear dependencies on the state. Note that no approximation has been introduced in lifting eq. (55) to eq. (58).

We now postulate the form of the lifted reduced model based on the lifted PDE (eq. (58)). In addition to linear and quadratic terms, eq. (58) also contains constant, input, and bilinear terms, so the postulated model form is given by

$$\frac{\mathrm{d}\hat{\mathbf{w}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{w}} + \hat{\mathbf{H}}(\hat{\mathbf{w}} \otimes \hat{\mathbf{w}}) + \hat{\mathbf{N}}\hat{\mathbf{w}} g(t) + \hat{\mathbf{B}} g(t) + \hat{\mathbf{C}}, \tag{60}$$

where $\hat{\mathbf{A}}, \hat{\mathbf{N}} \in \mathbb{R}^{r \times r}$, $\hat{\mathbf{H}} \in \mathbb{R}^{r \times r^2}$, and $\hat{\mathbf{B}}, \hat{\mathbf{C}} \in \mathbb{R}^r$.

5.1.2. Numerical experiments

We wish to model the response of the FitzHugh-Nagumo system to inputs of the form

$$g(t) = \alpha t^3 e^{-\beta t} \tag{61}$$

where the parameter α varies log-uniformly between 500 and 50000, and the parameter β varies uniformly on the range $[10, 15]$. To train our Lift & Learn model of the form eq. (60), snapshot data from nine simulations of the original equations (eq. (55)) corresponding to the parameters $\alpha = [500, 5000, 50000]$ and $\beta = [10, 12.5, 15]$ are generated. The simulation outputs the system state s_1 and s_2 on a uniform spatial grid with $n = 512$ nodes. The data are recorded every 0.01 seconds, yielding 400 state snapshots for each simulation, with a total of 3600 snapshots used for learning. The lifting map is applied to the state data to obtain lifted data for w_1, w_2, and w_3. Separate POD bases are computed for each lifted state variable. That is, if $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(3)} \in \mathbb{R}^{512\times 3600}$ denote the snapshot data in w_1, w_2, and w_3, respectively, then

$$\mathbf{W}^{(1)} = \mathbf{U}^{(1)}\mathbf{\Sigma}^{(1)}\mathbf{V}^{(1)^\top}, \quad \mathbf{W}^{(2)} = \mathbf{U}^{(2)}\mathbf{\Sigma}^{(2)}\mathbf{V}^{(2)^\top}, \quad \mathbf{W}^{(3)} = \mathbf{U}^{(3)}\mathbf{\Sigma}^{(3)}\mathbf{V}^{(3)^\top}. \tag{62}$$

The number of modes to retain in our low-dimensional model is determined by examining the quantity

$$1 - \sum_{i=r+1}^{nd_w} \sigma_i^2 \Big/ \sum_{i=1}^{nd_w} \sigma_i^2, \tag{63}$$

where σ_i are the singular values of the data. The quantity in eq. (63) is the relative energy of the unretained modes of the training data, or the energy lost in truncation. The energy spectrum of the training data in each separate lifted state variable is shown in Figure 1. The required numbers of modes needed to capture 99.9%, 99.99%, 99.999%, and 99.9999% of the energy in the data are tabulated in Table 1, along with the corresponding Lift & Learn model dimensions.

		# modes required			
Retained energy		w_1	w_2	w_3	Total
99.9	%	1	1	1	3
99.99	%	2	1	3	6
99.999	%	3	3	4	10
99.9999	%	5	4	5	14

Table 1: Number of modes required to retain different amounts of energy in training data. The total number of modes corresponds to the Lift & Learn model dimension.

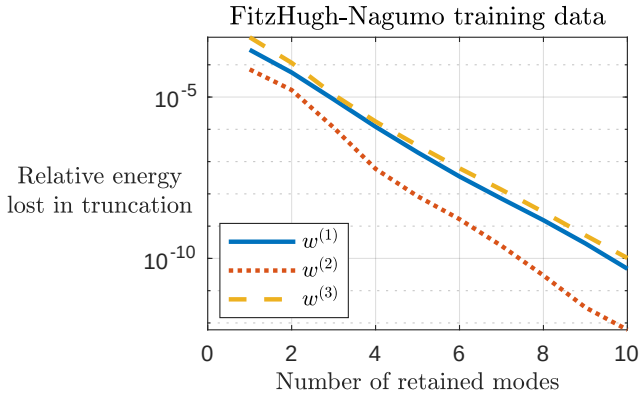

Figure 1: Energy spectrum of FitzHugh-Nagumo training data.

For each of the model sizes in Table 1, the data are generated by evaluating the original non-quadratic model and lifting the data, as described in Section 3.2. The corresponding input data $\mathbf{G}$ and bilinear state-input data $\hat{\mathbf{W}}\mathbf{G}$ are also collected. The state, time derivative, input, and bilinear state-input data are then used to learn a model of the form eq. (60). Additionally, we exploit knowledge of the governing equations to enforce block-sparsity in the least-squares solution; for example, there are only linear and constant terms in the evolution of w_2, so for the reduced state components corresponding to w_2, only the linear and constant terms are inferred.

The training set consists of the nine training trajectories described above. Two test sets are considered: one in which 100 trajectories are generated with α and β realizations randomly drawn from their distributions above, in the same regime as the training set, and a second test set in which 100 trajectories are generated from a different parameter regime, with α varying log-uniformly on $[50000, 5e6]$ and $\beta \sim \mathcal{U}([15, 20])$. Training and test errors are shown in Figure 2. For each training and test input, the error relative to the solution of the original full model, $\mathbf{S}_{\text{orig}}$, is calculated by solving the reduced model to generate predictions, then reconstructing the full lifted state from the Lift & Learn reduced trajectory, and finally reversing the lifting to obtain $\mathbf{S}_{\text{L\&L}}$, the Lift & Learn prediction of the trajectory in the original full state space. The relative error is given by

$$\frac{\|\mathbf{S}_{\text{L\&L}} - \mathbf{S}_{\text{orig}}\|_F}{\|\mathbf{S}_{\text{orig}}\|_F}. \tag{64}$$

The relative error of the intrusive lifted POD reduced model (obtained by

discretizing the lifted PDE and then reducing, as in [23]) is shown for reference.

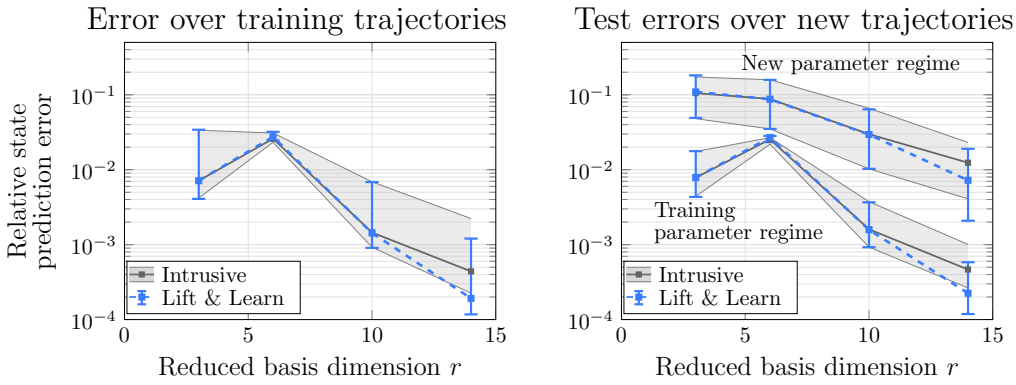

Figure 2: Lift & Learn model prediction error for FitzHugh-Nagumo system. Comparison to intrusive lifted POD reduced model performance for (left) the nine training trajectories and (right) two test sets of 100 new trajectories: one set drawn from the same parameter regime as the training trajectories and the other set drawn from a completely new parameter regime. Median and first/third quartile errors are shown.

For both test sets and the training set, the Lift & Learn model error is nearly the same as the lifted POD reduced model error for $r \leq 10$. The test errors for the first test set are of similar magnitude to the training errors, demonstrating the ability of the model to generalize to new inputs. For the second test set, the Lift & Learn model errors are higher than for the training set, but similar to the errors obtained by the intrusive lifted POD reduced model. In this case, the accuracy of the reduced model is limited by the ability of the POD basis computed from trajectories in one parameter regime to represent trajectories in a new parameter regime. The ability of the Lift & Learn model to recover the accuracy of the lifted POD reduced model is a key contribution of this work because it allows analyzable quadratic reduced models to be derived for nonlinear systems where the lifted full model is not available.

5.2. Application to the Euler equations

The fluid dynamics community has long recognized the utility of alternative variable representations. While the Euler and Navier-Stokes equations are most commonly derived in conservative variables, where each state is a conserved quantity (mass, momentum, energy), symmetric variables have

been exploited to guarantee stable models [34], and the quadratic structure of the specific volume variable representation has been exploited in [35] to allow a model stabilization procedure to be applied. In Section 5.2.1 the Euler equations are presented in the typical conservative formulation as well as in the quadratic specific volume formulation. Section 5.2.2 describes the test problem and presents Lift & Learn results when applied to this problem.

5.2.1. Euler equations and specific volume transformation

The one-dimensional Euler equations in the conservative variables are given by

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho e\end{pmatrix} = -\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2 + p\\ (\rho e + p)u\end{pmatrix}, \tag{65}$$

where the state $\vec{s} = \begin{pmatrix}\rho & \rho u & \rho e\end{pmatrix}^\top$ is comprised of the density ρ, specific momentum ρu, and total energy ρe. The equation of state $\rho e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u^2$ relates energy and pressure via the heat capacity ratio γ. Equation (65) contains non-polynomial nonlinearities in the conservative state. However, the Euler equations can be alternatively formulated in the specific volume state representation:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - \zeta\frac{\partial p}{\partial x}, \tag{66a}$$

$$\frac{\partial p}{\partial t} = -\gamma p\frac{\partial u}{\partial x} - \frac{\partial p}{\partial x}, \tag{66b}$$

$$\frac{\partial \zeta}{\partial t} = -u\frac{\partial \zeta}{\partial x} + \zeta\frac{\partial u}{\partial x}, \tag{66c}$$

where $\zeta = \frac{1}{\rho}$ is the specific volume, u the velocity, and p pressure. That is, the map $\mathcal{T}$ is given by

$$\mathcal{T} : \begin{pmatrix}\rho\\ \rho u\\ \rho e\end{pmatrix} \longmapsto \begin{pmatrix}u\\ p\\ 1/\rho\end{pmatrix} \equiv \vec{w}. \tag{67}$$

For constant γ, eq. (66) contains only quadratic nonlinear dependencies on the state and its spatial derivatives. This is a special case where a quadratic representation can be achieved via nonlinear state transformations without the addition of auxiliary variables, so $d_s = d_w$ and the map $\mathcal{T}$ is invertible. Note that eq. (66) can be extended to the three-dimensional Euler setting by adding the y- and z-velocity equations.

5.2.2. Numerical experiments

Equation (65) is solved on the periodic domain $x \in [0, 2)$ with mesh size $\Delta x = 0.01$ from $t = 0$ to $t = 0.01$ with timestep $\Delta t = 10^{-5}$. The initial pressure is 1 bar everywhere. The initial density is a periodic cubic spline interpolation at the points $x = 0, \frac{2}{3}, \frac{4}{3}$, where the interpolation values are drawn from a uniform distribution on the interval $[20, 24]$ kg/m^3. The initial velocity is also a periodic cubic spline interpolation of values at the same x-locations with interpolation values drawn from a uniform distribution on the interval [95, 105] m/s. The initial condition is therefore parametrized by six degrees of freedom: three for density and three for velocity.

A training data set of 64 trajectories is constructed using initial conditions corresponding to the 64 corners of the parameter domain. This data set is then transformed to the specific volume representation and non-dimensionalized. For this test problem, we use a single POD basis to represent the entire state. The energy spectrum of the non-dimensional training data is plotted in Figure 3. The numbers of modes required to retain different levels of energy are tabulated in Table 2.

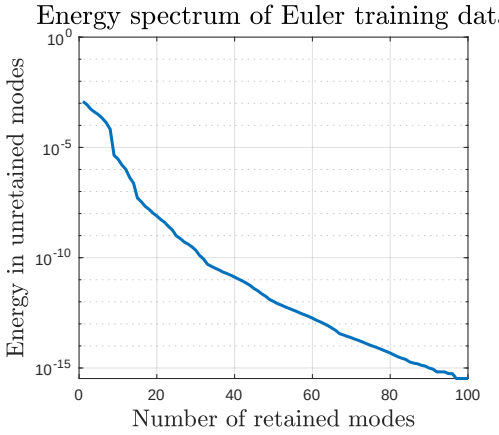

Figure 3: POD energy spectrum of Euler training data set (all state variables)

Retained energy	# modes required
$1 - 10^{-3}$	2
$1 - 10^{-5}$	9
$1 - 10^{-7}$	15
$1 - 10^{-9}$	25

Table 2: Number of modes required to retain energy of training data. The total number of modes corresponds to the Lift & Learn model dimension.

Note that the transformed system in eq. (66) contains no linear dependencies on the state, so only a quadratic operator $\hat{\mathbf{H}}$ is inferred. The inferred reduced model thus has the form

$$\frac{\mathrm{d}\hat{\mathbf{w}}}{\mathrm{d}t} = \hat{\mathbf{H}}(\hat{\mathbf{w}} \otimes \hat{\mathbf{w}}). \tag{68}$$

The Lift & Learn reduced model is then used to predict the state evolution for the training data set as well as a test data set. The test set consists of 100 trajectories based on initial conditions drawn randomly from their distributions. Median and first and third quartile relative errors over the training and test sets are shown in Figure 4. The performance of a lifted POD reduced model is shown for reference.

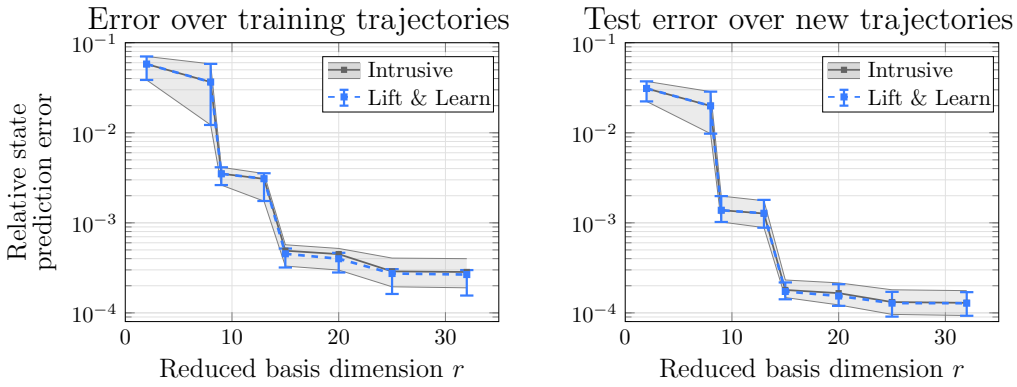

Figure 4: Lift & Learn model prediction error for Euler equations and comparison to intrusive lifted POD reduced model performance. Errors over the 64 training trajectories are plotted on the left and test error over 100 new test trajectories are plotted on the right. Median, first and third quartile errors are shown.

The Lift & Learn models are stable, accurate, and generalizable, achieving an error under 0.1% on both the training and test sets, which is similar to that of the lifted POD reduced model. Again, we emphasize that the lifted POD reduced model approach is usually not viable because the lifted full model is generally not available. The non-intrusive ability to recover the generalizability of the intrusive reduced model is a key contribution of our method.

6. Conclusions

For machine learning tools to achieve their full potential in scientific computing, they must be reliable, robust, and generalizable. Approaches which incorporate domain knowledge into the learning problem can help achieve these goals. We have presented Lift & Learn, a new domain-aware learning method which uses lifting transformations to expose quadratic structure

in a nonlinear dynamical system. This structure is then exploited to learn low-dimensional models which generalize well to conditions outside of their training data. Our method differs from previous works that use lifting to obtain reduced models in that our approach learns the quadratic reduced model from data generated by a model in the original nonlinear variables; we do not require the availability of a lifted full model. In contrast to learning approaches that fit models with arbitrary architectures to data, our physics-informed approach fits a quadratic model that respects the physics in the lifted variables, so that the learned model residual can be upper bounded by the truncation error of the full model and the projection error of the reduced basis. Numerical experiments on two different test problems demonstrate the ability of our learned models to recover the robustness and accuracy of the intrusive reduced models.

Acknowledgments

This work was supported in part by the US Air Force Center of Excellence on Multi-Fidelity Modeling of Rocket Combustor Dynamics award FA9550-17-1-0195, the Air Force Office of Scientific Research MURI on managing multiple information sources of multi-physics systems awards FA9550-15-1-0038 and FA9550-18-1-0023, the US Department of Energy Applied Mathematics MMICC Program award DESC0019334, and the SUTD-MIT International Design Centre. The first author also acknowledges support from the National Science Foundation Graduate Research Fellowship Program and the Fannie and John Hertz Foundation. The third author was partially supported by the US Department of Energy, Office of Advanced Scientific Computing Research, Applied Mathematics Program (Program Manager Dr. Steven Lee), DOE Award DESC0019334.

References

[1] J. L. Lumley, The structure of inhomogeneous turbulent flows, Atmospheric Turbulence and Radio Wave Propagation (1967).

[2] L. Sirovich, Turbulence and the dynamics of coherent structures. i-coherent structures. ii-symmetries and transformations. iii-dynamics and scaling, Quarterly of Applied Mathematics 45 (1987) 561–571.

[3] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual Review of Fluid Mechanics 25 (1993) 539–575.

[4] P. Holmes, J. L. Lumley, G. Berkooz, C. Rowley, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University Press, 2012.

[5] T. Bui-Thanh, M. Damodaran, K. Willcox, Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition, AIAA Journal 42 (2004) 1505–1516.

[6] M. Mifsud, A. Vendl, L.-U. Hansen, S. Görtz, Fusing wind-tunnel measurements and CFD data using constrained gappy proper orthogonal decomposition, Aerospace Science and Technology 86 (2019) 312–326.

[7] C. Audouze, F. D. Vuyst, P. B. Nair, Nonintrusive reduced-order modeling of parametrized time-dependent partial differential equations, Numer. Methods Partial Differential Eq. 29 (2013) 1587–1628.

[8] Q. Wang, J. S. Hesthaven, D. Ray, Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem, Journal of Computational Physics 384 (2019) 289–307.

[9] L. Mainini, K. E. Willcox, Data to decisions: Real-time structural assessment from sparse measurements affected by uncertainty, Computers & Structures 182 (2017) 296–312.

[10] R. Swischuk, L. Mainini, B. Peherstorfer, K. Willcox, Projection-based model reduction: Formulations for physics-based machine learning, Computers and Fluids 179 (2019) 704–717.

[11] H. Schaeffer, G. Tran, R. Ward, Extracting sparse dynamics from limited data, SIAM Journal on Applied Mathematics 78 (2018) 3279–3295.

[12] S. L. Brunton, J. L. Proctor, J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proceedings of the National Academy of Sciences 113 (2016) 3932–3937.

[13] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Advances 3 (2017).

[14] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, D. S. Henningson, Spectral analysis of nonlinear flows, Journal of Fluid Mechanics 641 (2009) 115–127.

[15] P. J. Schmid, Dynamic mode decomposition of numerical and experimental data, Journal of Fluid Mechanics 656 (2010) 5–28.

[16] B. Peherstorfer, K. Willcox, Data-driven operator inference for nonintrusive projection-based model reduction, Computer Methods in Applied Mechanics and Engineering 306 (2016) 196–215.

[17] B. O. Koopman, J. Neumann, Dynamical systems of continuous spectra, Proceedings of the National Academy of Sciences 18 (1932) 255.

[18] M. O. Williams, I. G. Kevrekidis, C. W. Rowley, A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition, Journal of Nonlinear Science 25 (2015) 1307–1346.

[19] I. G. Kevrekidis, C. W. Rowley, M. O. Williams, A kernel-based method for data-driven Koopman spectral analysis, Journal of Computational Dynamics 2 (2016) 247–265.

[20] Q. Li, F. Dietrich, E. M. Bollt, I. G. Kevrekidis, Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the Koopman operator, Chaos: An Interdisciplinary Journal of Nonlinear Science 27 (2017) 103111.

[21] G. P. McCormick, Computability of global solutions to factorable nonconvex programs: Part I–convex underestimating problems, Mathematical Programming 10 (1976) 147–175.

[22] C. Gu, QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 30 (2011) 1307–1320.

[23] B. Kramer, K. Willcox, Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition, AIAA Journal 57 (2019) 2297–2307.

[24] P. Benner, T. Breiten, Two-sided projection methods for nonlinear model order reduction, SIAM Journal on Scientific Computing 37 (2015) B239–B260.

[25] P. Benner, P. Goyal, S. Gugercin, H2-quasi-optimal model order reduction for quadratic-bilinear control systems, SIAM Journal on Matrix Analysis and Applications 39 (2018) 983–1032.

[26] I. V. Gosea, A. C. Antoulas, Data-driven model order reduction of quadratic-bilinear systems, Numerical Linear Algebra with Applications 25 (2018) e2200.

[27] B. Peherstorfer, Sampling low-dimensional Markovian dynamics for preasymptotically recovering reduced models from data with operator inference, arXiv:1908.11233 (2019).

[28] G. H. Golub, C. F. Van Loan, Matrix Computations, Johns Hopkins University Press, 4th edition, 2013.

[29] R. FitzHugh, Impulses and physiological states in theoretical models of nerve membrane, Biophysical Journal 1 (1961) 445–466.

[30] J. Nagumo, S. Arimoto, S. Yoshizawa, An active pulse transmission line simulating nerve axon, Proceedings of the IRE 50 (1962) 2061–2070.

[31] C. Rocsoreanu, A. Georgescu, N. Giurgiteanu, The FitzHugh-Nagumo model: Bifurcation and dynamics, volume 10, Springer Science & Business Media, 2012.

[32] A. Longtin, Stochastic resonance in neuron models, Journal of Statistical Physics 70 (1993) 309–327.

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

[34] T. J. R. Hughes, L. R. Franca, M. Mallet, A new finite element formulation for computational fluid dynamics: I. symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics, Comput. Methods Appl. Mech. Eng. 54 (1986) 223–234.

[35] M. Balajewicz, I. Tezaur, E. Dowell, Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations, Journal of Computational Physics 321 (2016) 224–241.