

# Discovering governing equations from data: Sparse identification of nonlinear dynamical systems

Steven L. Brunton<sup>1\*</sup>, Joshua L. Proctor<sup>2</sup>, J. Nathan Kutz<sup>3</sup>

<sup>1</sup> Department of Mechanical Engineering, University of Washington, Seattle, WA 98195, United States

<sup>2</sup> Institute for Disease Modeling, Bellevue, WA 98004, United States

<sup>3</sup> Department of Applied Mathematics, University of Washington, Seattle, WA 98195, United States

## Abstract

The ability to discover physical laws and governing equations from data is one of humankind's greatest intellectual achievements. A quantitative understanding of dynamic constraints and balances in nature has facilitated rapid development of knowledge and enabled advanced technological achievements, including aircraft, combustion engines, satellites, and electrical power. In this work, we combine sparsity-promoting techniques and machine learning with nonlinear dynamical systems to discover governing physical equations from measurement data. The only assumption about the structure of the model is that there are only a few important terms that govern the dynamics, so that the equations are sparse in the space of possible functions; this assumption holds for many physical systems. In particular, we use sparse regression to determine the fewest terms in the dynamic governing equations required to accurately represent the data. The resulting models are parsimonious, balancing model complexity with descriptive ability while avoiding overfitting. We demonstrate the algorithm on a wide range of problems, from simple canonical systems, including linear and nonlinear oscillators and the chaotic Lorenz system, to the fluid vortex shedding behind an obstacle. The fluid example illustrates the ability of this method to discover the underlying dynamics of a system that took experts in the community nearly 30 years to resolve. We also show that this method generalizes to parameterized, time-varying, or externally forced systems.

*Keywords*— Dynamical systems, Sparse regression, System identification, Compressed sensing.

## 1 Introduction

Extracting physical laws from data is a central challenge in many diverse areas of science and engineering. There are many critical data-driven problems, such as understanding cognition from neural recordings, inferring patterns in climate, determining stability of financial markets, predicting and suppressing the spread of disease, and controlling turbulence for greener transportation and energy. With abundant data and elusive laws, it is likely that data-driven discovery of dynamics will continue to play an increasingly important role in these efforts.

Advances in machine learning [19] and data science [29, 21] have promised a renaissance in the analysis and understanding of complex data, extracting patterns in vast multimodal data that is beyond the ability of humans to grasp. However, despite the rapid development of tools to understand static data based on statistical relationships, there has been slow progress in distilling physical models of dynamic processes from big data. This has limited the ability of data

---

\* Corresponding author. Tel.: +1 (609)-921-6415.  
E-mail address: sbrunton@uw.edu (S.L. Brunton).

science models to extrapolate the dynamics beyond the attractor where they were sampled and constructed.

An analogy may be drawn with the discoveries of Kepler and Newton. Kepler, equipped with the most extensive and accurate planetary data of the era, developed a *data-driven* model for the motion of the planets, resulting in his famous elliptic orbits. However, this was an *attractor* based view of the world, and it did not explain the fundamental dynamic relationships that give rise to planetary orbits, or provide a model for how these bodies react when perturbed. Newton, in contrast, discovered a dynamic relationship between momentum and energy that described the underlying processes responsible for these elliptic orbits. This dynamic model may be generalized to predict behavior in regimes where no data was collected. Newton’s model has proven remarkably robust for engineering design, making it possible to land a spacecraft on the moon, which would not have been possible using Kepler’s model alone.

A seminal breakthrough by Schmidt and Lipson [4, 40] has resulted in a new approach to determine the underlying structure of a nonlinear dynamical system from data. This method uses symbolic regression (i.e., genetic programming [22]) to find nonlinear differential equations, and it balances complexity of the model, measured in the number of terms, with model accuracy. The resulting model identification realizes a long-sought goal of the physics and engineering communities to discover dynamical systems from data. However, symbolic regression is expensive, does not scale well to large systems of interest, and may be prone to overfitting unless care is taken to explicitly balance model complexity with predictive power. In [40], the Pareto front is used to find parsimonious models in a large family of candidate models.

In this work, we re-envision the dynamical system discovery problem from an entirely new perspective of sparse regression [42, 14, 18] and compressed sensing [12, 8, 9, 7, 2, 43]. In particular, we leverage the fact that most physical systems have only a few relevant terms that define the dynamics, making the governing equations *sparse* in a high-dimensional nonlinear function space. Before the advent of compressive sampling, and related sparsity-promoting methods, determining the few non-zero terms in a nonlinear dynamical system would have involved a combinatorial brute-force search, meaning that the methods would not scale to larger problems with Moore’s law. However, powerful new theory guarantees that the sparse solution may be determined with high-probability using convex methods that do scale favorably with problem size. The resulting nonlinear model identification inherently balances model complexity (i.e., sparsity of right hand side dynamics) with accuracy, and the underlying convex optimization algorithms ensure that the method will be applicable to large-scale problems.

The method described here shares some similarity to the recent dynamic mode decomposition (DMD), which is a linear dynamic regression [36, 39]. DMD is an example of an equation-free method [20], since it only relies on measurement data, but not on knowledge of the governing equations. Recent advances in the extended DMD have developed rigorous connections between DMD built on nonlinear observable functions and the Koopman operator theory for nonlinear dynamical systems [36, 30]. However, there is currently no theory for which nonlinear observable functions to use, so that assumptions must be made on the form of the dynamical system. In contrast, the method developed here results in a *sparse, nonlinear* regression that automatically determines the relevant terms in the dynamical system. The trend to exploit sparsity in dynamical systems is recent but growing [38, 33, 25, 6, 35, 1]. In this work, promoting sparsity in the dynamics results in parsimonious natural laws.

## 2 Background

There is a long and fruitful history of modeling dynamics from data, resulting in powerful techniques for system identification [23]. Many of these methods arose out of the need to understand complex flexible structures, such as the Hubble space telescope or the international space station. The resulting models have been widely applied in nearly every branch of engineering and applied mathematics, most notably for model-based feedback control. However, methods for system identification typically require assumptions on the form of the model, and most often result in linear dynamics, limiting their effectiveness to small amplitude transient perturbations around a fixed point of the dynamics [15].

This work diverges from the seminal work on system identification, and instead builds on symbolic regression and sparse representation. In particular, symbolic regression is used to find nonlinear functions describing the relationships between variables and measured dynamics (i.e., time derivatives). Traditionally, model complexity is balanced with describing capability using parsimony arguments such as the Pareto front. Here, we use sparse representation to determine the relevant model terms in an efficient and scalable framework.

### 2.1 Symbolic regression and machine learning

Symbolic regression involves the determination of a function that relates input–output data, and it may be viewed as a form of machine learning. Typically, the function is determined using genetic programming, which is an evolutionary algorithm that builds and tests candidate functions out of simple building blocks [22]. These functions are then modified according to a set of evolutionary rules and generations of functions are tested until a pre-determined accuracy is achieved.

Recently, symbolic regression has been applied to data from *dynamical* systems, and ordinary differential equations were discovered from measurement data [40]. Because it is possible to overfit with symbolic regression and genetic programming, a parsimony constraint must be imposed, and in [40], they accept candidate equations that are at the Pareto front of complexity.

### 2.2 Sparse representation and compressive sensing

In many regression problems, only a few terms in the regression are important, and a *sparse feature selection* mechanism is required. For example, consider data measurements  $\mathbf{y} \in \mathbb{R}^m$  that may be a linear combination of columns from a feature library  $\Theta \in \mathbb{R}^{m \times p}$ ; the linear combination of columns is given by entries of the vector  $\boldsymbol{\xi} \in \mathbb{R}^p$  so that:

$$\mathbf{y} = \Theta \boldsymbol{\xi}. \quad (1)$$

Performing a standard regression to solve for  $\boldsymbol{\xi}$  will result in a solution with nonzero contributions in each element. However, if *sparsity* of  $\boldsymbol{\xi}$  is desired, so that most of the entries are zero, then it is possible to add an  $L^1$  regularization term to the regression, resulting in the LASSO [14, 18, 42]:

$$\boldsymbol{\xi} = \underset{\boldsymbol{\xi}'}{\operatorname{argmin}} \|\Theta \boldsymbol{\xi}' - \mathbf{y}\|_2 + \lambda \|\boldsymbol{\xi}'\|_1. \quad (2)$$

The parameter  $\lambda$  weights the sparsity constraint. This formulation is closely related to the compressive sensing framework, which allows for the sparse vector  $\boldsymbol{\xi}$  to be determined from relatively few *incoherent* random measurements [12, 8, 9, 7, 2, 43]. The sparse solution  $\boldsymbol{\xi}$  to Eq. 1 may also be used for sparse classification schemes, such as the sparse representation for classification (SRC) [44]. Importantly, the compressive sensing and sparse representation architectures are convex and scale well to large problems, as opposed to brute-force combinatorial alternatives.

### 3 Sparse identification of nonlinear dynamics (SINDy)

In this work, we are concerned with identifying the governing equations that underly a physical system based on data that may be realistically collected in simulations or experiments. Generically, we seek to represent the system as a nonlinear dynamical system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)). \quad (3)$$

The vector  $\mathbf{x}(t) = [x_1(t) \ x_2(t) \ \cdots \ x_n(t)]^T \in \mathbb{R}^n$  represents the state of the system at time  $t$ , and the nonlinear function  $\mathbf{f}(\mathbf{x}(t))$  represents the dynamic constraints that define the equations of motion of the system. In the following sections, we will generalize Eq. (3) to allow the dynamics  $\mathbf{f}$  to vary in time, and also with respect to a set of bifurcation parameters  $\boldsymbol{\mu} \in \mathbb{R}^q$ .

The key observation in this paper is that for many systems of interest, the function  $\mathbf{f}$  often consists of only a few terms, making it sparse in the space of possible functions. For example, the Lorenz system in Eq. (22c) has very few terms in the space of polynomial functions. Recent advances in compressive sensing and sparse regression make this viewpoint of sparsity favorable, since it is now possible to determine *which* right hand side terms are non-zero without performing a computationally intractable brute-force search.

To determine the form of the function  $\mathbf{f}$  from data, we collect a time-history of the state  $\mathbf{x}(t)$  and its derivative  $\dot{\mathbf{x}}(t)$  sampled at a number of instances in time  $t_1, t_2, \dots, t_m$ . These data are then arranged into two large matrices:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^T(t_1) \\ \mathbf{x}^T(t_2) \\ \vdots \\ \mathbf{x}^T(t_m) \end{bmatrix} = \begin{array}{c} \overbrace{\begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_n(t_m) \end{bmatrix}}^{\text{state}} \\ \underbrace{\hspace{10em}}_{\text{time}} \end{array} \quad (4a)$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}^T(t_1) \\ \dot{\mathbf{x}}^T(t_2) \\ \vdots \\ \dot{\mathbf{x}}^T(t_m) \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \cdots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \cdots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \cdots & \dot{x}_n(t_m) \end{bmatrix}. \quad (4b)$$

Next, we construct an augmented library  $\Theta(\mathbf{X})$  consisting of candidate nonlinear functions of the columns of  $\mathbf{X}$ . For example,  $\Theta(\mathbf{X})$  may consist of constant, polynomial and trigonometric terms:

$$\Theta(\mathbf{X}) = \left[ \begin{array}{c|c|c|c|c|c|c|c|c|c} \mathbf{1} & \mathbf{X} & \mathbf{X}^{P_2} & \mathbf{X}^{P_3} & \cdots & \sin(\mathbf{X}) & \cos(\mathbf{X}) & \sin(2\mathbf{X}) & \cos(2\mathbf{X}) & \cdots \end{array} \right]. \quad (5)$$

Here, higher polynomials are denoted as  $\mathbf{X}^{P_2}, \mathbf{X}^{P_3}$ , etc. For example,  $\mathbf{X}^{P_2}$  denotes the quadratic nonlinearities in the state variable  $\mathbf{x}$ , given by:

$$\mathbf{X}^{P_2} = \begin{bmatrix} x_1^2(t_1) & x_1(t_1)x_2(t_1) & \cdots & x_2^2(t_1) & x_2(t_1)x_3(t_1) & \cdots & x_n^2(t_1) \\ x_1^2(t_2) & x_1(t_2)x_2(t_2) & \cdots & x_2^2(t_2) & x_2(t_2)x_3(t_2) & \cdots & x_n^2(t_2) \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_1^2(t_m) & x_1(t_m)x_2(t_m) & \cdots & x_2^2(t_m) & x_2(t_m)x_3(t_m) & \cdots & x_n^2(t_m) \end{bmatrix}. \quad (6)$$

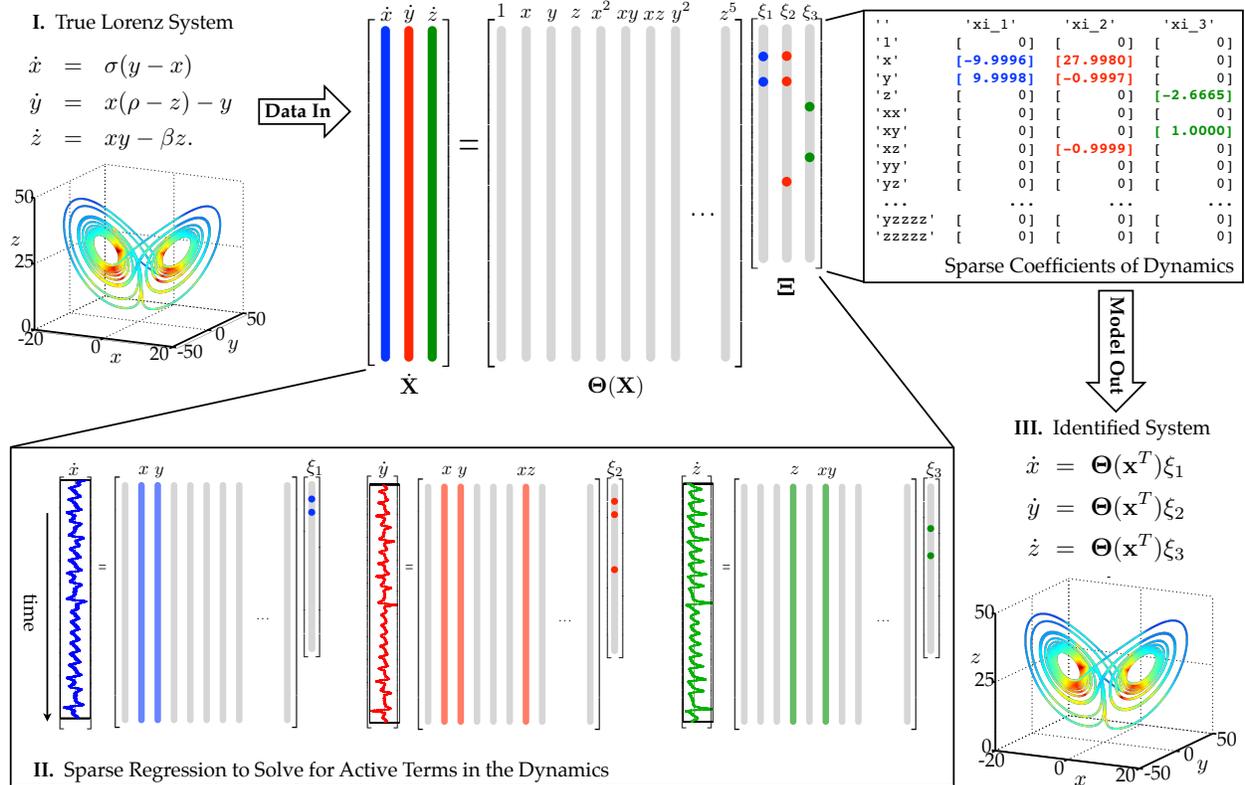


Figure 1: Schematic of our algorithm for sparse identification of nonlinear dynamics, demonstrated on the Lorenz equations. Data is collected from measurements of the system, including a time history of the states  $\mathbf{X}$  and derivatives  $\dot{\mathbf{X}}$ . Next, a library of nonlinear functions of the states,  $\Theta(\mathbf{X})$ , is constructed. This nonlinear feature library is used to find the fewest terms needed to satisfy  $\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi$ . The few entries in the vectors of  $\Xi$ , solved for by sparse regression, denote the relevant terms in the right-hand side of the dynamics. Parameter values are  $\sigma = 10, \beta = 8/3, \rho = 28, (x_0, y_0, z_0)^T = (-8, 7, 27)^T$ . The trajectory on the Lorenz attractor is colored by the adaptive time-step required, with red requiring a smaller timestepp.

Each column of  $\Theta(\mathbf{X})$  represents a candidate function for the right hand side of Eq. (3). There is tremendous freedom of choice in constructing the entries in this matrix of nonlinearities. Since we believe that only a few of these nonlinearities are active in each row of  $\mathbf{f}$ , we may set up a sparse regression problem to determine the sparse vectors of coefficients  $\Xi = [\xi_1 \ \xi_2 \ \dots \ \xi_n]$  that determine which nonlinearities are active, as illustrated in Fig. 1.

$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi. \quad (7)$$

Each column  $\xi_k$  of  $\Xi$  represents a sparse vector of coefficients determining which terms are active in the right hand side for one of the row equations  $\dot{x}_k = \mathbf{f}_k(\mathbf{x})$  in Eq. (3). Once  $\Xi$  has been determined, a model of each row of the governing equations may be constructed as follows:

$$\dot{x}_k = \mathbf{f}_k(\mathbf{x}) = \Theta(\mathbf{x}^T)\xi_k. \quad (8)$$

Note that  $\Theta(\mathbf{x}^T)$  is a vector of symbolic functions of elements of  $\mathbf{x}$ , as opposed to  $\Theta(\mathbf{X})$ , which is a data matrix. This results in the overall model

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) = \Xi^T(\Theta(\mathbf{x}^T))^T. \quad (9)$$

We may solve for  $\Xi$  in Eq. (7) using sparse regression.

### 3.1 Algorithm for sparse representation of dynamics with noise

There are a number of algorithms to determine sparse solutions  $\Xi$  to the regression problem in Eq. (7). Each column of Eq. (7) requires a distinct optimization problem to find the sparse vector of coefficients  $\xi_k$  for the  $k^{\text{th}}$  row equation.

For the examples in this paper, the matrix  $\Theta(\mathbf{X})$  has dimensions  $m \times p$ , where  $p$  is the number of candidate nonlinear functions, and where  $m \gg p$  since there are more time samples of data than there are candidate nonlinear functions. In most realistic cases, the data  $\mathbf{X}$  and  $\dot{\mathbf{X}}$  will be contaminated with noise so that Eq. (7) does not hold exactly. In the case that  $\mathbf{X}$  is relatively clean but the derivatives  $\dot{\mathbf{X}}$  are noisy, the equation becomes

$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\Xi + \eta\mathbf{Z}, \quad (10)$$

where  $\mathbf{Z}$  is a matrix of independent identically distributed Gaussian entries with zero mean, and  $\eta$  is the noise magnitude. Thus we seek a sparse solution to an overdetermined system with noise.

The LASSO [14, 42] from statistics works well with this type of data, providing a sparse regression. However, it may be computationally expensive for very large data sets.

An alternative is to implement the sequential thresholded least-squares algorithm in Code (1). In this algorithm, we start with a least-squares solution for  $\Xi$  and then threshold all coefficients that are smaller than some cutoff value  $\lambda$ . Once the indices of the remaining non-zero coefficients are identified, we obtain another least-squares solution for  $\Xi$  onto the remaining indices. These new coefficients are again thresholded using  $\lambda$ , and the procedure is continued until the non-zero coefficients converge. This algorithm is computationally efficient, and it rapidly converges to a sparse solution in a small number of iterations. The algorithm also benefits from simplicity, with a single parameter  $\lambda$  required to determine the degree of sparsity in  $\Xi$ .

Depending on the noise, it may still be necessary to filter the data  $\mathbf{X}$  and derivative  $\dot{\mathbf{X}}$  before solving for  $\Xi$ . In particular, if only the data  $\mathbf{X}$  is available, and  $\dot{\mathbf{X}}$  must be obtained by differentiation, then the resulting derivative matrix may have large noise magnitude. To counteract this, we use the total variation regularized derivative [10] to de-noise the derivative. An alternative would be to filter the data  $\mathbf{X}$  and  $\dot{\mathbf{X}}$ , for example using the optimal hard threshold for singular values described in [13].

It is important to note that previous algorithms to identify dynamics from data have been quite sensitive to noise [40]. The algorithm in Code (1) is remarkably robust to noise, and even when velocities must be approximated from noisy data, the algorithm works surprisingly well.

Code 1: Sparse representation algorithm in Matlab.

```

%% compute Sparse regression: sequential least squares
Xi = Theta\dXdT; % initial guess: Least-squares

% lambda is our sparsification knob.
for k=1:10
    smallinds = (abs(Xi)<lambda); % find small coefficients
    Xi(smallinds)=0; % and threshold
    for ind = 1:n % n is state dimension
        biginds = ~smallinds(:,ind);
        % Regress dynamics onto remaining terms to find sparse Xi
        Xi(biginds,ind) = Theta(:,biginds)\dXdT(:,ind);
    end
end
end

```

### 3.2 Cross-validation to determine parsimonious sparse solution on Pareto front

To determine the sparsification parameter  $\lambda$  in the algorithm in Code (1), it is helpful to use the concept of cross-validation from machine learning. It is always possible to hold back some test data apart from the training data to test the validity of models away from training values. In addition, it is important to consider the balance of model complexity (given by the number of nonzero coefficients in  $\Xi$ ) with the model accuracy. There is an “elbow” in the curve of accuracy vs. complexity parameterized by  $\lambda$ , the so-called Pareto front. This value of  $\lambda$  represents a good tradeoff between complexity and accuracy, and it is similar to the approach taken in [40].

### 3.3 Extensions and Connections

There are a number of extensions to the basic theory above that generalize this approach to a broader set of problems. First, the method is generalized to a discrete-time formulation, establishing a connection with the dynamic mode decomposition (DMD). Next, high-dimensional systems obtained from discretized partial differential equations are considered, extending the method to incorporate dimensionality reduction techniques to handle big data. Finally, the sparse regression framework is modified to include bifurcation parameters, time-dependence, and external forcing.

#### 3.3.1 Discrete-time representation

The aforementioned strategy may also be implemented on discrete-time dynamical systems:

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k). \quad (11)$$

There are a number of reasons to implement Eq. (11). First, many systems, such as the logistic map in Eq. (26) are inherently discrete-time systems. In addition, it may be possible to recover specific integration schemes used to advance Eq. (3). The discrete-time formulation also foregoes the calculation of a derivative from noisy data. The data collection will now involve two matrices  $\mathbf{X}_1^{m-1}$  and  $\mathbf{X}_2^m$ :

$$\mathbf{X}_1^{m-1} = \begin{bmatrix} \text{---} & \mathbf{x}_1^T & \text{---} \\ \text{---} & \mathbf{x}_2^T & \text{---} \\ & \vdots & \\ \text{---} & \mathbf{x}_{m-1}^T & \text{---} \end{bmatrix}, \quad \mathbf{X}_2^m = \begin{bmatrix} \text{---} & \mathbf{x}_2^T & \text{---} \\ \text{---} & \mathbf{x}_3^T & \text{---} \\ & \vdots & \\ \text{---} & \mathbf{x}_m^T & \text{---} \end{bmatrix}. \quad (12)$$

The continuous-time sparse regression problem in Eq. (7) now becomes:

$$\mathbf{X}_2^m = \Theta(\mathbf{X}_1^{m-1})\Xi \quad (13)$$

and the function  $\mathbf{f}$  is the same as in Eq. (9).

In the discrete setting in Eq. (11), and for linear dynamics, there is a striking resemblance to dynamic mode decomposition. In particular, if  $\Theta(\mathbf{x}) = \mathbf{x}$ , so that the dynamical system is linear, then Eq. (13) becomes

$$\mathbf{X}_2^m = \mathbf{X}_1^{m-1}\Xi \implies (\mathbf{X}_2^m)^T = \Xi^T (\mathbf{X}_1^{m-1})^T. \quad (14)$$

This is equivalent to the DMD, which seeks a dynamic regression onto linear dynamics  $\Xi^T$ . In particular,  $\Xi^T$  is  $n \times n$  dimensional, which may be prohibitively large for a high-dimensional state  $\mathbf{x}$ . Thus, DMD identifies the dominant terms in the eigendecomposition of  $\Xi^T$ .

### 3.3.2 High-dimensional systems, partial differential equations, and dimensionality reduction

Often, the physical system of interest may be naturally represented by a partial differential equation (PDE) in a few spatial variables. If data is collected from a numerical discretization or from experimental measurements on a spatial grid, then the state dimension  $n$  may be prohibitively large. For example, in fluid dynamics, even simple two-dimensional and three-dimensional flows may require tens of thousands up to billions of variables to represent the discretized system.

The method described above is prohibitive for a large state dimension  $n$ , both because of the factorial growth of  $\Theta$  in  $n$  and because each of the  $n$  row equations in Eq. (8) requires a separate optimization. Fortunately, many high-dimensional systems of interest evolve on a low-dimensional manifold or attractor that may be well-approximated using a dimensionally reduced low-rank basis  $\Psi$  [16, 27]. For example, if data  $\mathbf{X}$  is collected for a high-dimensional system as in Eq. (4a), it is possible to obtain a low-rank approximation using the singular value decomposition (SVD):

$$\mathbf{X}^T = \Psi \Sigma \mathbf{V}^*. \quad (15)$$

In this case, the state  $\mathbf{x}$  may be well approximated in a truncated modal basis  $\Psi_r$ , given by the first  $r$  columns of  $\Psi$  from the SVD:

$$\mathbf{x} \approx \Psi_r \mathbf{a}, \quad (16)$$

where  $\mathbf{a}$  is an  $r$ -dimensional vector of mode coefficients. We assume that this is a good approximation for a relatively low rank  $r$ . Thus, instead of using the original high-dimensional state  $\mathbf{x}$ , it is possible to obtain a sparse representation of the Galerkin projected dynamics  $\mathbf{f}_P$  in terms of the coefficients  $\mathbf{a}$ :

$$\dot{\mathbf{a}} = \mathbf{f}_P(\mathbf{a}). \quad (17)$$

There are many choices for a low-rank basis, including proper orthogonal decomposition (POD) [3, 16], based on the SVD.

### 3.3.3 External forcing, bifurcation parameters, and normal forms

In practice, many real-world systems depend on parameters, and dramatic changes, or bifurcations, may occur when the parameter is varied [15, 26]. The algorithm above is readily extended to encompass these important parameterized systems, allowing for the discovery of normal forms associated with a bifurcation parameter  $\mu$ . First, we append  $\mu$  to the dynamics:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}; \mu) \quad (18a)$$

$$\dot{\mu} = 0. \quad (18b)$$

It is then possible to identify the right hand side  $\mathbf{f}(\mathbf{x}; \mu)$  as a sparse combination of functions of components in  $\mathbf{x}$  as well as the bifurcation parameter  $\mu$ . This idea is illustrated on two examples, the one-dimensional logistic map and the two-dimensional Hopf normal form.

Time-dependence, known external forcing or control  $\mathbf{u}(t)$  may also be added to the dynamics:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}(t), t) \quad (19a)$$

$$\dot{t} = 1. \quad (19b)$$

This generalization makes it possible to analyze systems that are externally forced or controlled. For example, the climate is both parameterized [26] and has external forcing, including carbon dioxide and solar radiation. The financial market presents another important example with forcing and active feedback control, in the form of regulations, taxes, and interest rates.

## 4 Results

We demonstrate the methods described in Sec. 3 on a number of canonical systems, ranging from simple linear and nonlinear damped oscillators, to noisy measurements of the fully chaotic Lorenz system, and to measurements of the unsteady fluid wake behind a circular cylinder, extending this method to nonlinear partial differential equations (PDEs) and high-dimensional data. Finally, we show that bifurcation parameters may be included in the sparse models, recovering the correct normal forms from noisy measurements of the logistic map and the Hopf normal form.

### 4.1 Example 1: Simple illustrative systems

#### 4.1.1 Example 1a: Two-dimensional damped oscillator (linear vs. nonlinear)

In this example, we consider the two-dimensional damped harmonic oscillator with either linear or cubic dynamics, as in Eq. (20). The dynamic data and the sparse identified model are shown in Fig. 2. The correct form of the nonlinearity is obtained in each case; the augmented nonlinear library  $\Theta(\mathbf{x})$  includes polynomials in  $\mathbf{x}$  up to fifth order. The sparse identified model and algorithm parameters are shown in the Appendix in Tables 1 and 2.

$$\frac{d}{dt} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -0.1 & 2 \\ -2 & -0.1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \frac{d}{dt} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -0.1 & 2 \\ -2 & -0.1 \end{bmatrix} \begin{bmatrix} x^3 \\ y^3 \end{bmatrix} \quad (20)$$

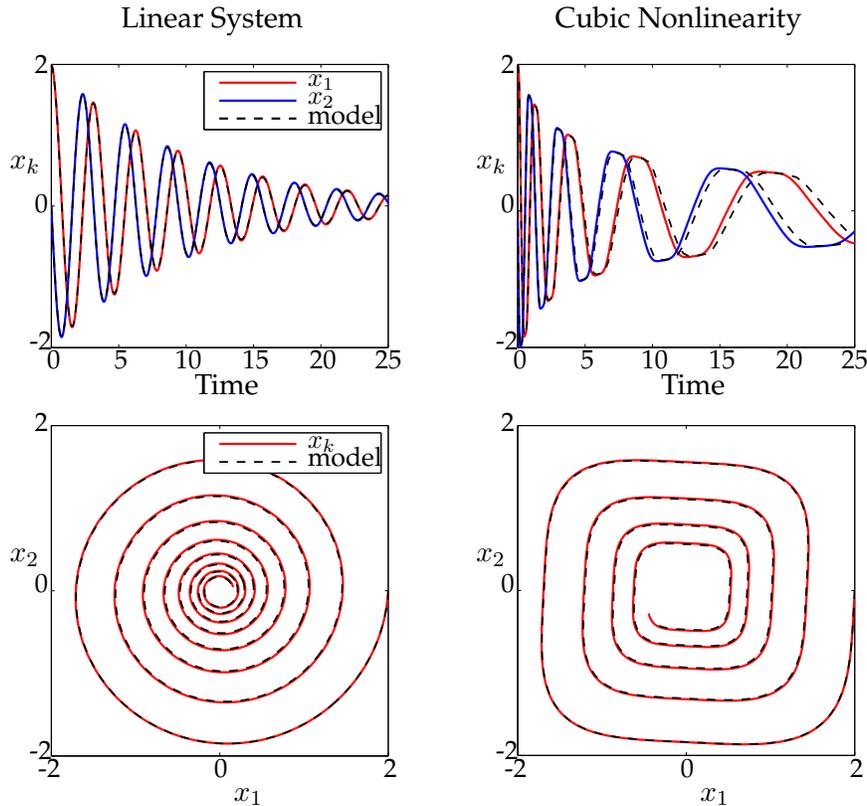


Figure 2: Comparison of linear system (left) and system with cubic nonlinearity (right). The sparse identified system correctly identifies the form of the dynamics and accurately reproduces the phase portraits.

### 4.1.2 Example 1b: Three-dimensional linear system

A linear system with three variables and the sparse approximation are shown in Fig. 3. In this case, the dynamics are given by

$$\frac{d}{dt} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -0.1 & -2 & 0 \\ 2 & -0.1 & 0 \\ 0 & 0 & -0.3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \quad (21)$$

The sparse identification algorithm correctly identifies the system in the space of polynomials up to second or third order, and the sparse model is given in Table 3. Interestingly, including polynomial terms of higher order (e.g. orders 4 or 5) introduces a degeneracy in the sparse identification algorithm, because linear combinations of powers of  $e^{\lambda t}$  may approximate other exponential rates. This unexpected degeneracy motivates a hierarchical approach to identification, where subsequently higher order terms are included until the algorithm either converges or diverges.

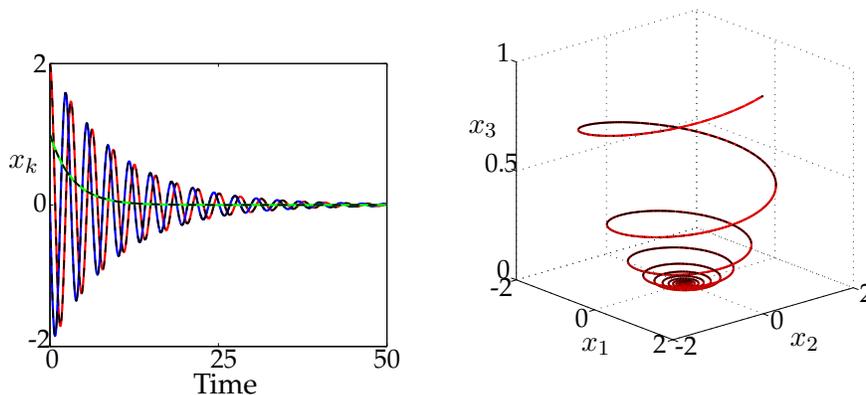


Figure 3: Three-dimensional linear system (solid colored lines) is well-captured by sparse identified system (dashed black line).

## 4.2 Example 2: Lorenz system (Nonlinear ODE)

Here, we consider the nonlinear Lorenz system [24] to explore the identification of chaotic dynamics evolving on an attractor, shown in Fig. 1:

$$\dot{x} = \sigma(y - x) \quad (22a)$$

$$\dot{y} = x(\rho - z) - y \quad (22b)$$

$$\dot{z} = xy - \beta z. \quad (22c)$$

Although these equations give rise to rich and chaotic dynamics that evolve on an attractor, there are only a few terms in the right-hand side of the equations. Figure 1 shows a schematic of how data is collected for this example, and how sparse dynamics are identified in a space of possible right-hand side functions using convex  $\ell_1$ -minimization.

For this example, data is collected for the Lorenz system, and stacked into two large data matrices  $\mathbf{X}$  and  $\dot{\mathbf{X}}$ , where each row of  $\mathbf{X}$  is a snapshot of the state  $\mathbf{x}$  in time, and each row of  $\dot{\mathbf{X}}$

is a snapshot of the time derivative of the state  $\dot{\mathbf{x}}$  in time. Here, the right-hand side dynamics are identified in the space of polynomials  $\Theta(\mathbf{X})$  in  $(x, y, z)$  up to fifth order:

$$\Theta(\mathbf{X}) = \left[ \begin{array}{c|c|c|c|c|c|c|c|c|c|c} \mathbf{x}(t) & \mathbf{y}(t) & \mathbf{z}(t) & \mathbf{x}(t)^2 & \mathbf{x}(t)\mathbf{y}(t) & \mathbf{x}(t)\mathbf{z}(t) & \mathbf{y}(t)^2 & \mathbf{y}(t)\mathbf{z}(t) & \mathbf{z}(t)^2 & \dots & \mathbf{z}(t)^5 \end{array} \right]. \quad (23)$$

Each column of  $\Theta(\mathbf{X})$  represents a candidate function for the right hand side of Eq. (3), and a sparse regression determines which terms are active in the dynamics, as in Fig. 1, and Eq. (7).

Zero-mean Gaussian measurement noise with variance  $\eta$  is added to the derivative calculation to investigate the effect of noisy derivatives. The short-time ( $t = 0$  to  $t = 20$ ) and long-time ( $t = 0$  to  $t = 250$ ) system reconstruction is shown in Fig. 4 for two different noise values,  $\eta = 0.01$  and  $\eta = 10$ . The trajectories are also shown in dynamo view in Fig. 5, and the  $\ell_2$  error vs. time for increasing noise  $\eta$  is shown in Fig. 6. Although the  $\ell_2$  error increases for large noise values  $\eta$ , the form of the equations, and hence the attractor dynamics, are accurately captured. Because the system has a positive Lyapunov exponent, small differences in model coefficients or initial conditions grow exponentially, until saturation, even though the attractor may remain unchanged.

In the Lorenz example, the ability to capture dynamics on the attractor is more important than the ability to predict an individual trajectory, since chaos will quickly cause any small variations in initial conditions or model coefficients to diverge exponentially. As shown in Fig. 1, our sparse model identification algorithm accurately reproduces the attractor dynamics from chaotic trajectory measurements. The algorithm not only identifies the correct linear and quadratic terms in the dynamics, but it accurately determines the coefficients to within .03% of the true values. When the derivative measurements are contaminated with noise, the correct dynamics are identified, and the attractor is well-preserved for surprisingly large noise values. When the noise is too large, the structure identification fails before the coefficients become too inaccurate.

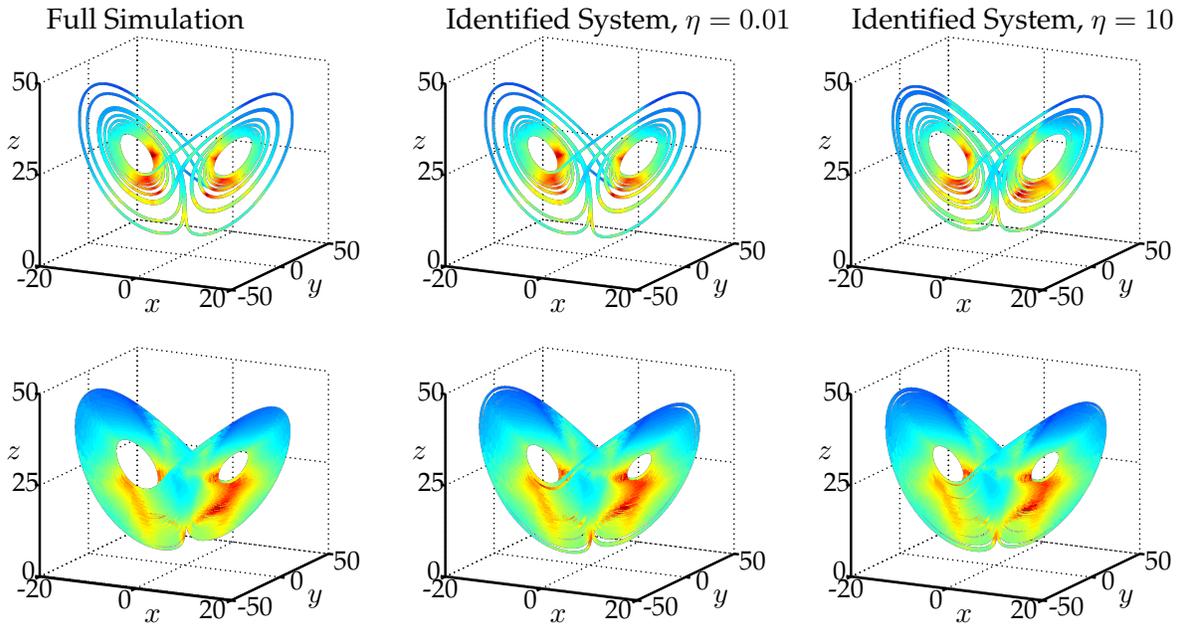


Figure 4: Trajectories of the Lorenz system for short-time integration from  $t = 0$  to  $t = 20$  (top) and long-time integration from  $t = 0$  to  $t = 250$  (bottom). The full dynamics (left) are compared with the sparse identified systems (middle, right) for various additive noise. The trajectories are colored by  $\Delta t$ , the adaptive Runge-Kutta time step. This color is a proxy for local sensitivity.

For this example, we use the standard parameters  $\sigma = 10, \beta = 8/3, \rho = 28$ , with an initial condition  $[x \ y \ z]^T = [-8 \ 7 \ 27]^T$ . Data is collected from  $t = 0$  to  $t = 100$  with a time-step of  $\Delta t = 0.001$ .

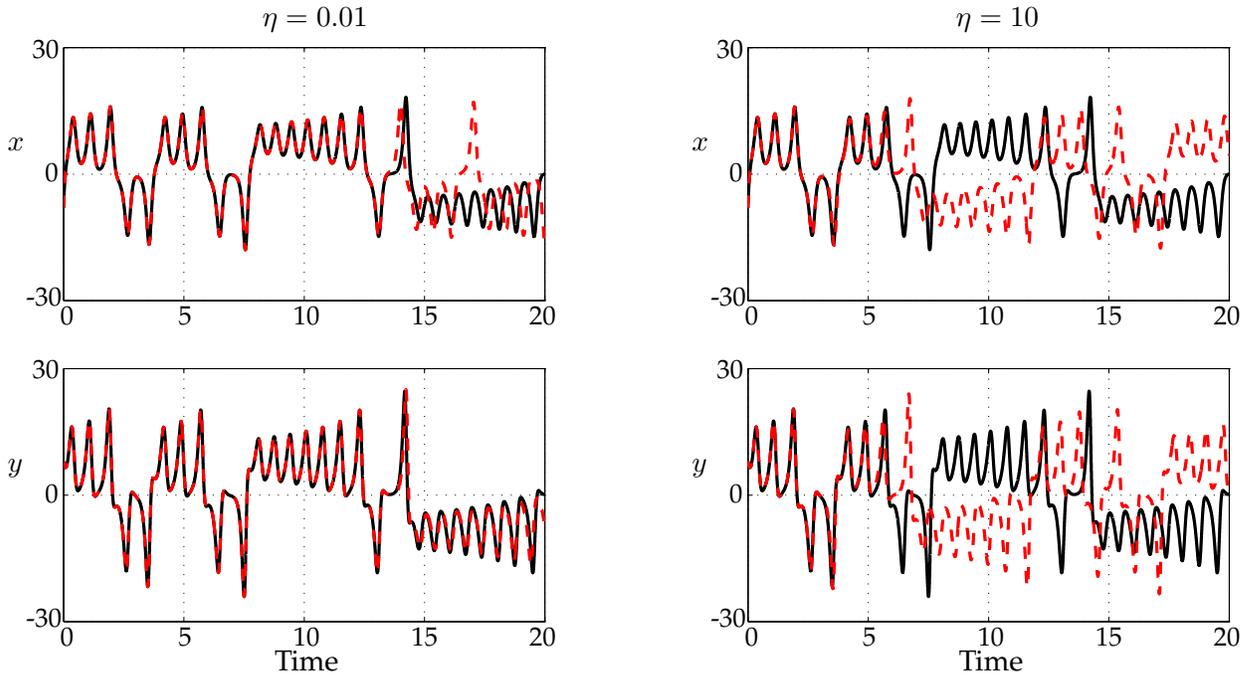


Figure 5: Dynamo view of trajectories of the Lorenz system. The exact system is shown in black (—) and the sparse identified system is shown in the dashed red arrow (---).

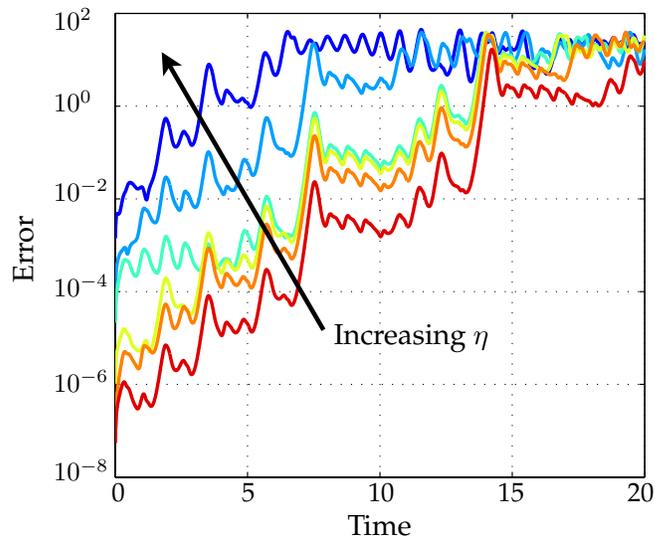


Figure 6: Error vs. time for sparse identified systems generated from data with increasing sensor noise  $\eta$ . This error corresponds to the difference between solid black and dashed red curves in Fig. 5. Sensor noise values are  $\eta \in \{0.0001, 0.001, 0.01, 0.1, 1.0, 10.0\}$ .

### 4.3 Example 3: Fluid wake behind a cylinder (Nonlinear PDE)

The Lorenz system is a low-dimensional model of more realistic high-dimensional partial differential equation (PDE) models for fluid convection in the atmosphere. Many systems of interest are governed by PDEs [38], such as weather and climate, epidemiology, and the power grid, to name a few. Each of these examples are characterized by big data, consisting of large spatially resolved measurements consisting of millions or billions of states and spanning orders of magnitude of scale in both space and time. However, many high-dimensional, real-world systems evolve on a low-dimensional attractor, making the effective dimension much smaller [16].

Here we generalize the sparse identification of nonlinear dynamics method to an example in fluid dynamics that typifies many of the challenges outlined above. Data is collected for the fluid flow past a cylinder at Reynolds number 100 using direct numerical simulations of the two-dimensional Navier-Stokes equations [41, 11]. Then, the nonlinear dynamic relationship between the dominant coherent structures is identified from these flow field measurements with no knowledge of the governing equations.

The low-Reynolds number flow past a cylinder is a particularly interesting example because of its rich history in fluid mechanics and dynamical systems. It has long been theorized that turbulence may be the result of a sequence of Hopf bifurcations that occur as the Reynolds number of the flow increases [37]. The Reynolds number is a rough measure of the ratio of inertial and viscous forces, and an increasing Reynolds number may correspond, for example, to increasing flow velocity, giving rise to more rich and intricate structures in the fluid.

After 15 years, the first Hopf bifurcation was discovered in a fluid system, in the transition from a steady laminar wake to laminar periodic vortex shedding at Reynolds number 47 [17, 45, 32]. This discovery led to a long-standing debate about how a Hopf bifurcation, with cubic nonlinearity, can be exhibited in a Navier-Stokes fluid with quadratic nonlinearities. After 15 more years, this was finally resolved using a separation of time-scales argument and a mean-field model [31], shown in Eq. (24). It was shown that coupling between oscillatory modes and the base flow gives rise to a slow manifold (see Fig. 7, left), which results in algebraic terms that approximate cubic nonlinearities on slow timescales.

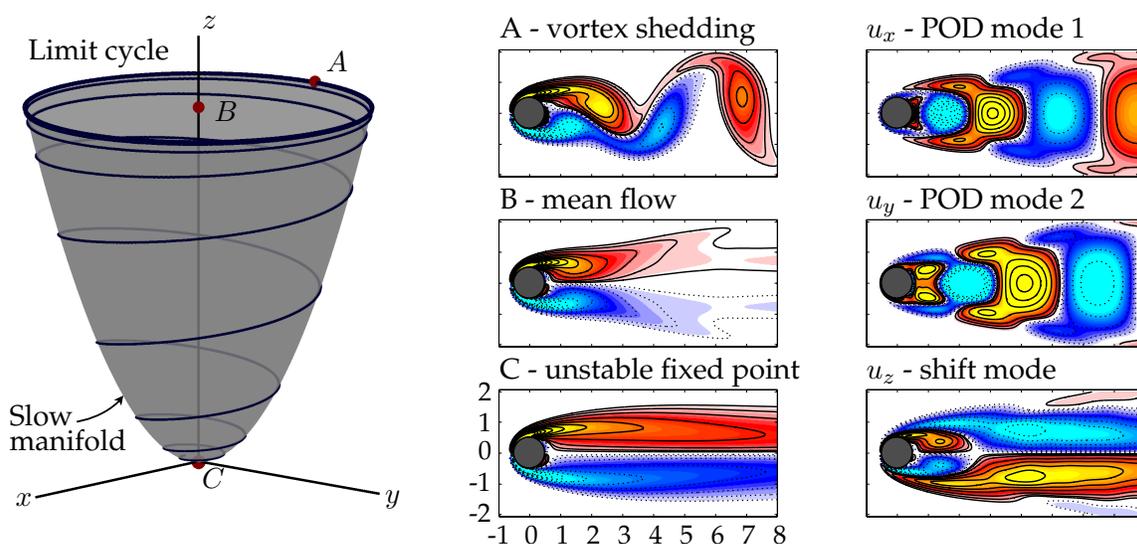


Figure 7: Illustration of the low-rank dynamics underlying the periodic vortex shedding behind a circular cylinder at low Reynolds number,  $Re = 100$ .

This example provides a compelling test-case for the proposed algorithm, since the underlying form of the dynamics took nearly three decades to uncover. Indeed, the sparse dynamics algorithm correctly identifies the on-attractor and off-attractor dynamics using quadratic nonlinearities and preserves the correct slow-manifold dynamics. It is interesting to note that when the off-attractor trajectories are not included in the system identification, the algorithm incorrectly identifies the dynamics using cubic nonlinearities, and fails to correctly identify the dynamics associated with the shift mode, which connects the mean flow to the unstable steady state.

### 4.3.1 Direct numerical simulation

The direct numerical simulation involves a fast multi-domain immersed boundary projection method [41, 11]. Four grids are used, each with a resolution of  $450 \times 200$ , with the finest grid having dimensions of  $9 \times 4$  cylinder diameters and the largest grid having dimensions of  $72 \times 32$  diameters. The finest grid has 90,000 points, and each subsequent coarser grid has 67,500 distinct points. Thus, if the state includes the vorticity at each grid point, then the state dimension is 292,500. The vorticity field on the finest grid is shown in Fig. 7. The code is non-dimensionalized so that the cylinder diameter and free-stream velocity are both equal to one:  $D = 1$  and  $U_\infty = 1$ , respectively. The simulation time-step is  $\Delta t = 0.02$  non dimensional time units.

### 4.3.2 Mean field model

To develop a mean-field model for the cylinder wake, first we must reduce the dimension of the system. The proper orthogonal decomposition (POD) [16], provides a low-rank basis that is optimal in the  $L^2$  sense, resulting in a hierarchy of orthonormal modes that are ordered by mode energy. The first two most energetic POD modes capture a significant portion of the energy; the steady-state vortex shedding is a limit cycle in these coordinates. An additional mode, called the shift mode, is included to capture the transient dynamics connecting the unstable steady state with the mean of the limit cycle [31] (i.e., the direction connecting point ‘C’ to point ‘B’ in Fig. 7).

In the three-dimensional coordinate system described above, the mean-field model for the cylinder dynamics are given by:

$$\dot{x} = \mu x - \omega y + Axz \quad (24a)$$

$$\dot{y} = \omega x + \mu y + Ayz \quad (24b)$$

$$\dot{z} = -\lambda(z - x^2 - y^2). \quad (24c)$$

If  $\lambda$  is large, so that the  $z$ -dynamics are fast, then the mean flow rapidly corrects to be on the (slow) manifold  $z = x^2 + y^2$  given by the amplitude of vortex shedding. When substituting this algebraic relationship into Eqs. 24a and 24b, we recover the Hopf normal form on the slow manifold.

Remarkably, similar dynamics are discovered by the sparse dynamics algorithm, purely from data collected from simulations. The identified model coefficients, shown in Table 5, only include quadratic nonlinearities, consistent with the Navier-Stokes equations. Moreover, the transient behavior, shown in Figs. 9 and 10, is captured qualitatively for solutions that do not start on the slow manifold. When the off-attractor dynamics in Fig. 9 are not included in the training data, the model incorrectly identifies a simple Hopf normal form in  $x$  and  $y$  with cubic nonlinearities.

The data from Fig. 10 was not included in the training data, and although qualitatively similar, the identified model does not exactly reproduce the transients. Since this initial condition had twice the fluctuation energy in the  $x$  and  $y$  directions, the slow manifold approximation may not be valid here. Relaxing the sparsity condition, it is possible to obtain models that agree almost perfectly with the data in Figs. 8-10, although the model includes higher order nonlinearities.

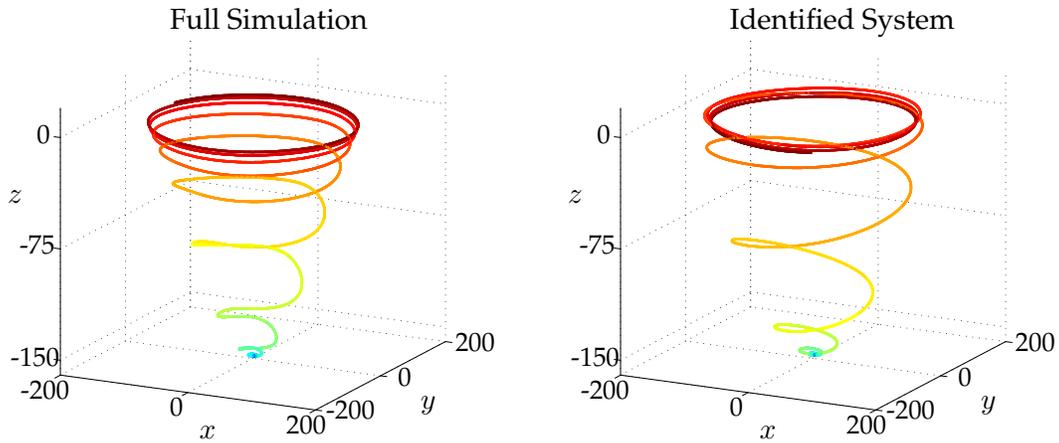


Figure 8: Evolution of the cylinder wake trajectory in reduced coordinates. The full simulation (left) comes from direct numerical simulation of the Navier-Stokes equations, and the identified system (right) captures the dynamics on the slow manifold. Color indicates simulation time.

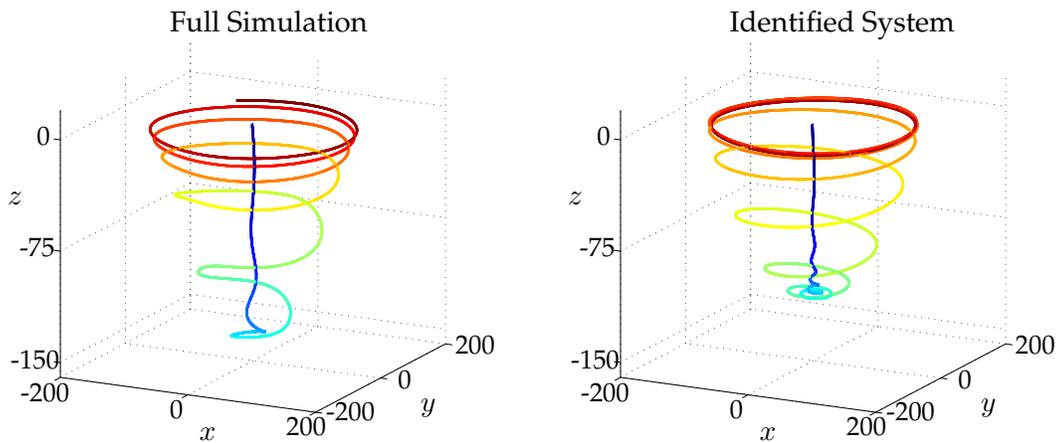


Figure 9: Evolution of the cylinder wake trajectory starting from a flow state initialized at the mean of the steady-state limit cycle. Both the full simulation and sparse model capture the off-attractor dynamics, characterized by rapid attraction of the trajectory onto the slow manifold.

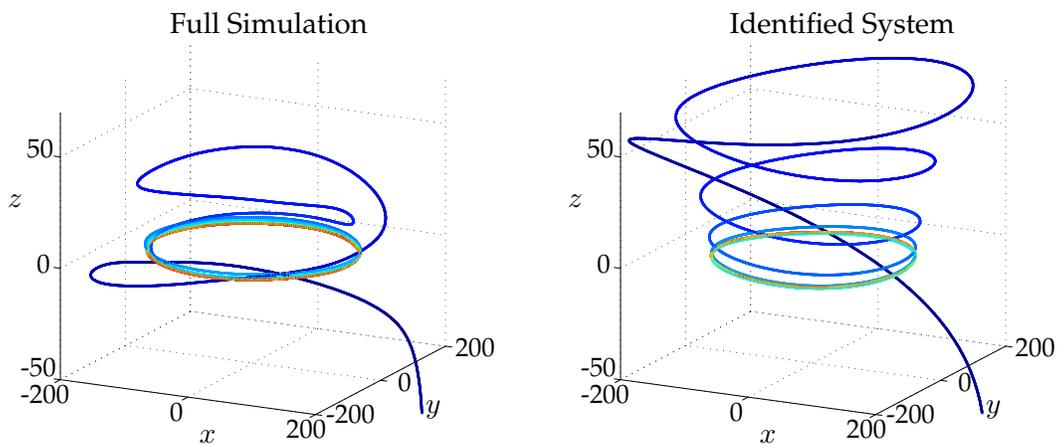


Figure 10: This simulation corresponds to an initial condition obtained by doubling the magnitude of the limit cycle behavior. This data was not included in the training of the sparse model.

## 4.4 Example 4: Bifurcations and Normal Forms

It is also possible to identify normal forms associated with a bifurcation parameter  $\mu$  by suspending it in the dynamics as a variable:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}; \mu) \quad (25a)$$

$$\dot{\mu} = 0. \quad (25b)$$

It is then possible to identify the right hand side  $\mathbf{f}(\mathbf{x}; \mu)$  as a sparse combination of functions of components in  $\mathbf{x}$  as well as the bifurcation parameter  $\mu$ . This idea is illustrated on two examples, the one-dimensional logistic map and the two-dimensional Hopf normal form.

### 4.4.1 Logistic map

The logistic map is a classical model that exhibits a cascade of bifurcations, leading to chaotic trajectories. The dynamics with stochastic forcing  $\eta_k$  and parameter  $\mu$  are given by

$$x_{k+1} = \mu x_k(1 - x_k) + \eta_k. \quad (26)$$

Sampling the stochastic system at ten parameter values of  $\mu$ , the algorithm correctly identifies the underlying parameterized dynamics, shown in Fig. 11 and Table 6.

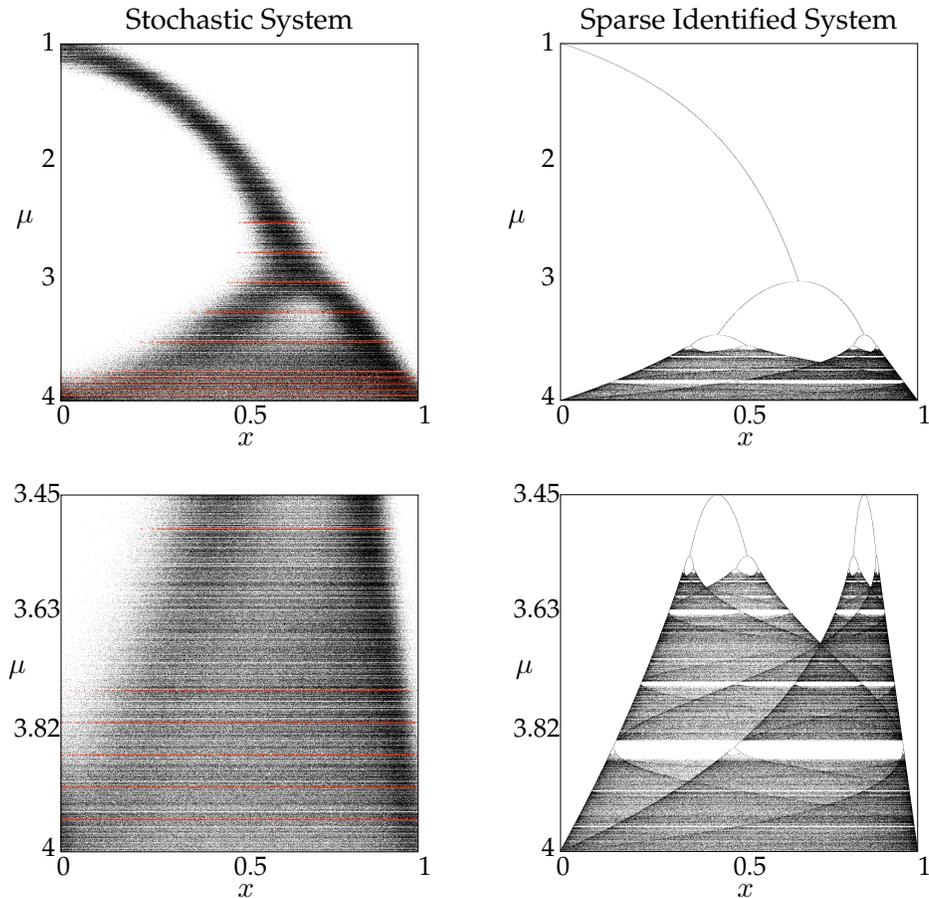


Figure 11: Attracting sets of the logistic map vs. the parameter  $\mu$ . (left) Data from stochastically forced system and (right) the sparse identified system. Data is sampled at rows indicated in red for  $\mu \in \{2.5, 2.75, 3, 3.25, 3.5, 3.75, 3.8, 3.85, 3.9, 3.95\}$ . The forcing  $\eta_k$  is Gaussian with magnitude 0.025.

#### 4.4.2 Hopf normal form

The final example illustrating the ability of the sparse dynamics method to identify parameterized normal forms is the Hopf normal form [28]. Noisy data is collected from the Hopf system

$$\dot{x} = \mu x + \omega y - Ax(x^2 + y^2) \quad (27a)$$

$$\dot{y} = -\omega x + \mu y - Ay(x^2 + y^2) \quad (27b)$$

for various values of the parameter  $\mu$ . Data is collected on the blue and red trajectories in Fig. 12, and noise is added to simulate sensor noise. The total variation derivative [10] is used to de-noise the derivative for use in the algorithm.

The sparse model identification algorithm correctly identifies the Hopf normal form, with model parameters given in Table 7. The noise-free model reconstruction is shown in Fig. 13. Note that with noise in the training data, although the model terms are correctly identified, the actual values of the cubic terms are off by almost 8%. Collecting more training data or reducing the noise magnitude both improve the model agreement.

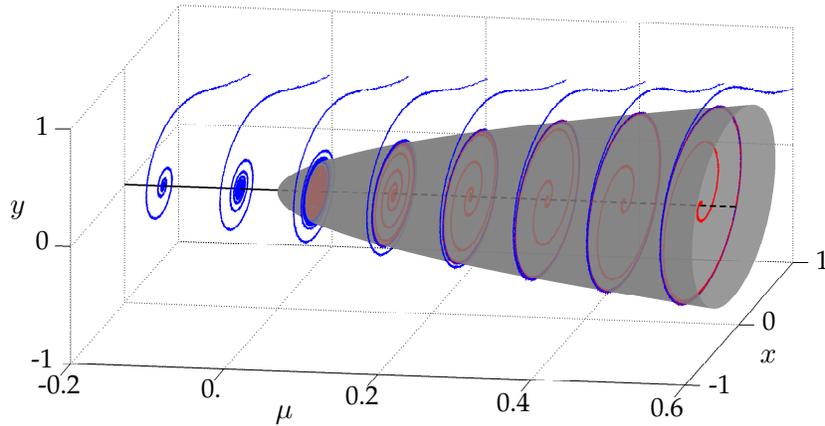


Figure 12: Training data to identify Hopf normal form. Blue trajectories denote solutions that start outside of the fixed point for  $\mu < 0$  or the limit cycle for  $\mu > 0$ , and red trajectories denote solutions that start inside of the limit cycle.

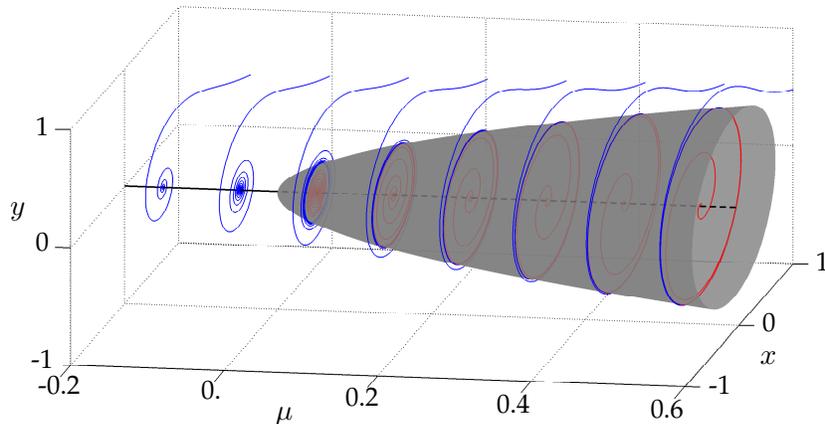


Figure 13: Sparse model captures the Hopf normal form. Initial conditions are the same as in Fig. 12

## 5 Discussion

In summary, we have demonstrated a powerful new technique to identify nonlinear dynamical systems from data without assumptions on the form of the governing equations. This builds on prior work in symbolic regression but with innovations related to sparse regression, which allow our algorithms to scale to high-dimensional systems. We demonstrate this new method on a number of example systems exhibiting chaos, high-dimensional data with low-rank coherence, and parameterized dynamics. As shown in the Lorenz example, the ability to predict a specific trajectory may be less important than the ability to capture the attractor dynamics. The example from fluid dynamics highlights the remarkable ability of this method to extract dynamics in a fluid system that took three decades for experts in the community to explain. There are numerous fields where this method may be applied, where there is ample data and the absence of governing equations, including neuroscience, climate science, epidemiology, and financial markets. Fields that already use genetic programming, such as machine learning control for turbulent fluid systems [5, 34], may also benefit. Finally, normal forms may be discovered by including parameters in the optimization, as shown on two examples. The identification of *sparse* governing equations and parameterizations marks a significant step toward the long-held goal of intelligent, unassisted identification of dynamical systems.

A number of open problems remain surrounding the dynamical systems aspects of this procedure. For example, many systems possess dynamical symmetries and conserved quantities that may alter the form of the identified dynamics. For example, the degenerate identification of a linear system in a space of high-order polynomial nonlinearities suggest a connection with near-identity transformations and dynamic similarity. We believe that this may be a fruitful line of research. Finally, it will be important to identify which approximating function space to use based on the data available. For example, it may be possible to improve the function space to make the dynamics more sparse through subsequent coordinate transformations [15].

Data science is not a panacea for all problems in science and engineering, but used in the right way, it provides a principled approach to maximally leverage the data that we have and inform what new data to collect. Big data is happening all across the sciences, where the data is inherently *dynamic*, and where traditional approaches are prone to overfitting. Data discovery algorithms that produce *parsimonious* models are both rare and desirable. Data-science will only become more critical to efforts in science in engineering, where data is abundant, but physical laws remain elusive. These efforts include understanding the neural basis of cognition, extracting and predicting coherent changes in the climate, stabilizing financial markets, managing the spread of disease, and controlling turbulence,

## Acknowledgements

We gratefully acknowledge valuable discussions with Bingni W. Brunton and Bernd R. Noack. SLB acknowledges support from the University of Washington department of Mechanical Engineering and as a Data Science Fellow in the eScience Institute (NSF, Moore-Sloan Foundation, Washington Research Foundation). JLP thanks Bill and Melinda Gates for their active support of the Institute for Disease Modeling and their sponsorship through the Global Good Fund. JNK acknowledges support from the U.S. Air Force Office of Scientific Research (FA9550-09-0174).

## References

- [1] Zhe Bai, Thakshila Wimalajeewa, Zachary Berger, Guannan Wang, Mark Glauser, and Pramod K Varshney. Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA Journal*, pages 1–14, 2014.
- [2] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–120, 2007.
- [3] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 23:539–575, 1993.
- [4] Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.
- [5] S. L. Brunton and B. R. Noack. Closed-loop turbulence control: Progress and challenges. *Applied Mechanics Reviews*, 67:050801–1–050801–48, 2015.
- [6] S. L. Brunton, J. H. Tu, I. Bright, and J. N. Kutz. Compressive sensing and low-rank libraries for classification of bifurcation regimes in nonlinear dynamical systems. *SIAM Journal on Applied Dynamical Systems*, 13(4):1716–1732, 2014.
- [7] E. J. Candès. Compressive sensing. *Proc. International Congress of Mathematics*, 2006.
- [8] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.
- [9] E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 59(8):1207–1223, 2006.
- [10] Rick Chartrand. Numerical differentiation of noisy, nonsmooth data. *ISRN Applied Mathematics*, 2011, 2011.
- [11] T. Colonius and K. Taira. A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. *Computer Methods in Applied Mechanics and Engineering*, 197:2131–2146, 2008.
- [12] D. L. Donoho. Compressed sensing. *IEEE Trans. Information Theory*, 52(4):1289–1306, 2006.
- [13] M. Gavish and D. L. Donoho. The optimal hard threshold for singular values is  $4/\sqrt{3}$ . *ArXiv e-prints*, 2014.
- [14] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.
- [15] P. Holmes and J. Guckenheimer. *Nonlinear oscillations, dynamical systems, and bifurcations of vector fields*, volume 42 of *Applied Mathematical Sciences*. Springer-Verlag, Berlin, 1983.
- [16] P. J. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Monographs in Mechanics. Cambridge University Press, Cambridge, England, 2nd edition, 2012.
- [17] C. P. Jackson. A finite-element study of the onset of vortex shedding in flow past variously shaped bodies. *Journal of Fluid Mechanics*, 182:23–45, 1987.
- [18] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Springer, 2013.
- [19] MI Jordan and TM Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.
- [20] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and C. Theodoropoulos. Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Science*, 1(4):715–762, 2003.
- [21] Muin J Khoury and John PA Ioannidis. Medicine. big data meets public health. *Science*, 346(6213):1054–1055, 2014.
- [22] J. R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.
- [23] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, 1999.
- [24] Edward N Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sciences*, 20(2):130–141, 1963.
- [25] Alan Mackey, Hayden Schaeffer, and Stanley Osher. On the compressive spectral method. *Multiscale Modeling & Simulation*, 12(4):1800–1827, 2014.
- [26] Andrew J Majda, Christian Franzke, and Daan Crommelin. Normal forms for reduced stochastic climate models. *Proceedings of the National Academy of Sciences*, 106(10):3649–3653, 2009.
- [27] Andrew J Majda and John Harlim. Information flow between subspaces of complex dynamical systems. *Proceedings of the National Academy of Sciences*, 104(23):9558–9563, 2007.
- [28] Jerrold E Marsden and Marjorie McCracken. *The Hopf bifurcation and its applications*, volume 19. Springer-Verlag, 1976.
- [29] Vivien Marx. Biology: The big challenges of big data. *Nature*, 498(7453):255–260, 2013.

- [30] Igor Mezic. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.
- [31] B. R. Noack, K. Afanasiev, M. Morzynski, G. Tadmor, and F. Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363, 2003.
- [32] DJ Olinger and KR Sreenivasan. Nonlinear dynamics of the wake of an oscillating cylinder. *Physical review letters*, 60(9):797, 1988.
- [33] Vidvuds Ozoliņš, Rongjie Lai, Russel Caflisch, and Stanley Osher. Compressed modes for variational problems in mathematics and physics. *Proceedings of the National Academy of Sciences*, 110(46):18368–18373, 2013.
- [34] V. Parezanovic, J.-C. Larentie, T. Duriez, C. Fourment, J. Delville, J.-P. Bonnet, L. Cordier, B. R. Noack, M. Segond, M. Abel, T. Shaqarin, and S. L. Brunton. Mixing layer manipulation experiment – from periodic forcing to machine learning closed-loop control. *Journal Flow Turbulence and Combustion*, 94(1):155–173, 2015.
- [35] J. L. Proctor, S. L. Brunton, B. W. Brunton, and J. N. Kutz. Exploiting sparsity and equation-free architectures in complex systems (invited review). *The European Physical Journal Special Topics*, 223(13):2665–2684, 2014.
- [36] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D.S. Henningson. Spectral analysis of nonlinear flows. *J. Fluid Mech.*, 645:115–127, 2009.
- [37] D. Ruelle and F. Takens. On the nature of turbulence. *Communications in Mathematical Physics*, 20:167–192, 1971.
- [38] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences USA*, 110(17):6634–6639, 2013.
- [39] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, August 2010.
- [40] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.
- [41] K. Taira and T. Colonius. The immersed boundary method: a projection approach. *Journal of Computational Physics*, 225(2):2118–2137, 2007.
- [42] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. of the Royal Statistical Society B*, pages 267–288, 1996.
- [43] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.
- [44] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.
- [45] Z. Zebib. Stability of viscous flow past a circular cylinder. *Journal of Engineering Mathematics*, 21:155–165, 1987.

# Appendix

Table 1: Damped harmonic oscillator with linear terms.

''	'xdot'	'ydot'
'1'	[ 0]	[ 0]
'x'	[-0.1015]	[-1.9990]
'y'	[ 2.0027]	[-0.0994]
'xx'	[ 0]	[ 0]
'xy'	[ 0]	[ 0]
'yy'	[ 0]	[ 0]
'xxx'	[ 0]	[ 0]
'xxy'	[ 0]	[ 0]
'xyy'	[ 0]	[ 0]
'yyy'	[ 0]	[ 0]
'xxxx'	[ 0]	[ 0]
'xxxxy'	[ 0]	[ 0]
'xxyy'	[ 0]	[ 0]
'xyyy'	[ 0]	[ 0]
'yyyy'	[ 0]	[ 0]
'xxxxx'	[ 0]	[ 0]
'xxxxxy'	[ 0]	[ 0]
'xxxxyy'	[ 0]	[ 0]
'xxyyyy'	[ 0]	[ 0]
'xyyyyy'	[ 0]	[ 0]
'yyyyyy'	[ 0]	[ 0]

Table 2: Damped harmonic oscillator with cubic nonlinearity.

' '	' xdot'	' ydot'
' 1'	[ 0]	[ 0]
' x'	[ 0]	[ 0]
' y'	[ 0]	[ 0]
' xx'	[ 0]	[ 0]
' xy'	[ 0]	[ 0]
' yY'	[ 0]	[ 0]
' xxx'	[-0.0996]	[-1.9994]
' xxy'	[ 0]	[ 0]
' xyy'	[ 0]	[ 0]
' yyy'	[ 1.9970]	[-0.0979]
' xxxx'	[ 0]	[ 0]
' xxxy'	[ 0]	[ 0]
' xxyy'	[ 0]	[ 0]
' xyyy'	[ 0]	[ 0]
' yyyy'	[ 0]	[ 0]
' xxxxx'	[ 0]	[ 0]
' xxxxy'	[ 0]	[ 0]
' xxxyy'	[ 0]	[ 0]
' xxyyy'	[ 0]	[ 0]
' xyyyy'	[ 0]	[ 0]
' yyyyy'	[ 0]	[ 0]

Table 3: Three-dimensional linear system.

' '	' xdot'	' ydot'	' zdot'
' 1'	[ 0]	[ 0]	[ 0]
' x'	[-0.0996]	[-1.9997]	[ 0]
' y'	[ 2.0005]	[-0.0994]	[ 0]
' z'	[ 0]	[ 0]	[-0.3003]
' xx'	[ 0]	[ 0]	[ 0]
' xy'	[ 0]	[ 0]	[ 0]
' xz'	[ 0]	[ 0]	[ 0]
' yY'	[ 0]	[ 0]	[ 0]
' yz'	[ 0]	[ 0]	[ 0]
' zz'	[ 0]	[ 0]	[ 0]

Table 4: Lorenz system identified using sparse representation with  $\eta = 1.0$ .

''	'xdot'	'ydot'	'zdot'
'1'	[ 0]	[ 0]	[ 0]
'x'	[-9.9996]	[27.9980]	[ 0]
'y'	[ 9.9998]	[-0.9997]	[ 0]
'z'	[ 0]	[ 0]	[-2.6665]
'xx'	[ 0]	[ 0]	[ 0]
'xy'	[ 0]	[ 0]	[ 1.0000]
'xz'	[ 0]	[-0.9999]	[ 0]
'yy'	[ 0]	[ 0]	[ 0]
'yz'	[ 0]	[ 0]	[ 0]
'zz'	[ 0]	[ 0]	[ 0]
'xxx'	[ 0]	[ 0]	[ 0]
'xxy'	[ 0]	[ 0]	[ 0]
'xxz'	[ 0]	[ 0]	[ 0]
'xyy'	[ 0]	[ 0]	[ 0]
'xyz'	[ 0]	[ 0]	[ 0]
'xzz'	[ 0]	[ 0]	[ 0]
'yyy'	[ 0]	[ 0]	[ 0]
'yyz'	[ 0]	[ 0]	[ 0]
'yzz'	[ 0]	[ 0]	[ 0]
'zzz'	[ 0]	[ 0]	[ 0]
'xxxx'	[ 0]	[ 0]	[ 0]
'xxxxy'	[ 0]	[ 0]	[ 0]
'xxxxz'	[ 0]	[ 0]	[ 0]
'xxyy'	[ 0]	[ 0]	[ 0]
'xxyz'	[ 0]	[ 0]	[ 0]
'xxzz'	[ 0]	[ 0]	[ 0]
'xyyy'	[ 0]	[ 0]	[ 0]
'xyyz'	[ 0]	[ 0]	[ 0]
'xyzz'	[ 0]	[ 0]	[ 0]
'xzzz'	[ 0]	[ 0]	[ 0]
'yyyy'	[ 0]	[ 0]	[ 0]
'yyyz'	[ 0]	[ 0]	[ 0]
'yyzz'	[ 0]	[ 0]	[ 0]
'yzzz'	[ 0]	[ 0]	[ 0]
'zzzz'	[ 0]	[ 0]	[ 0]
'xxxxx'	[ 0]	[ 0]	[ 0]
'xxxxxy'	[ 0]	[ 0]	[ 0]
'xxxxxz'	[ 0]	[ 0]	[ 0]
'xxxxyy'	[ 0]	[ 0]	[ 0]
'xxxxyz'	[ 0]	[ 0]	[ 0]
'xxxzzz'	[ 0]	[ 0]	[ 0]
'xxyyy'	[ 0]	[ 0]	[ 0]
'xxyyz'	[ 0]	[ 0]	[ 0]
'xxyz'	[ 0]	[ 0]	[ 0]
'xxzzz'	[ 0]	[ 0]	[ 0]
'xyyyy'	[ 0]	[ 0]	[ 0]
'xyyyz'	[ 0]	[ 0]	[ 0]
'xyyzz'	[ 0]	[ 0]	[ 0]
'xyzzz'	[ 0]	[ 0]	[ 0]
'xzzzz'	[ 0]	[ 0]	[ 0]
'yyyyy'	[ 0]	[ 0]	[ 0]
'yyyyyz'	[ 0]	[ 0]	[ 0]
'yyyzzz'	[ 0]	[ 0]	[ 0]
'yyzzz'	[ 0]	[ 0]	[ 0]
'yzzzz'	[ 0]	[ 0]	[ 0]
'zzzzz'	[ 0]	[ 0]	[ 0]

Table 5: Dynamics of cylinder wake modes using sparse representation. Notice that quadratic terms are identified.

	'xdot'	'ydot'	'zdot'
'1'	[ -0.1225]	[ -0.0569]	[ -20.8461]
'x'	[ -0.0092]	[ 1.0347]	[-4.6476e-04]
'y'	[ -1.0224]	[ 0.0047]	[ 2.4057e-04]
'z'	[-9.2203e-04]	[-4.4932e-04]	[ -0.2968]
'xx'	[ 0]	[ 0]	[ 0.0011]
'xy'	[ 0]	[ 0]	[ 0]
'xz'	[ 2.1261e-04]	[ 0.0022]	[ 0]
'yy'	[ 0]	[ 0]	[ 8.6432e-04]
'yz'	[ -0.0019]	[ -0.0018]	[ 0]
'zz'	[ 0]	[ 0]	[ -0.0010]
'xxx'	[ 0]	[ 0]	[ 0]
'xxy'	[ 0]	[ 0]	[ 0]
'xxz'	[ 0]	[ 0]	[ 0]
'xyy'	[ 0]	[ 0]	[ 0]
'xyz'	[ 0]	[ 0]	[ 0]
'xzz'	[ 0]	[ 0]	[ 0]
'yyy'	[ 0]	[ 0]	[ 0]
'yyz'	[ 0]	[ 0]	[ 0]
'yzz'	[ 0]	[ 0]	[ 0]
'zzz'	[ 0]	[ 0]	[ 0]
'xxxx'	[ 0]	[ 0]	[ 0]
'xxxxy'	[ 0]	[ 0]	[ 0]
'xxxxz'	[ 0]	[ 0]	[ 0]
'xxyy'	[ 0]	[ 0]	[ 0]
'xxyz'	[ 0]	[ 0]	[ 0]
'xxzz'	[ 0]	[ 0]	[ 0]
'xyyy'	[ 0]	[ 0]	[ 0]
'xyyz'	[ 0]	[ 0]	[ 0]
'xyzz'	[ 0]	[ 0]	[ 0]
'xzzz'	[ 0]	[ 0]	[ 0]
'yyyy'	[ 0]	[ 0]	[ 0]
'yyyz'	[ 0]	[ 0]	[ 0]
'yyzz'	[ 0]	[ 0]	[ 0]
'yzzz'	[ 0]	[ 0]	[ 0]
'zzzz'	[ 0]	[ 0]	[ 0]
'xxxxx'	[ 0]	[ 0]	[ 0]
'xxxxxy'	[ 0]	[ 0]	[ 0]
'xxxxxz'	[ 0]	[ 0]	[ 0]
'xxxxyy'	[ 0]	[ 0]	[ 0]
'xxxxyz'	[ 0]	[ 0]	[ 0]
'xxxzzz'	[ 0]	[ 0]	[ 0]
'xxyyy'	[ 0]	[ 0]	[ 0]
'xxyyz'	[ 0]	[ 0]	[ 0]
'xxyzz'	[ 0]	[ 0]	[ 0]
'xxzzz'	[ 0]	[ 0]	[ 0]
'xyyyy'	[ 0]	[ 0]	[ 0]
'xyyyz'	[ 0]	[ 0]	[ 0]
'xyyzz'	[ 0]	[ 0]	[ 0]
'xyzzz'	[ 0]	[ 0]	[ 0]
'xzzzz'	[ 0]	[ 0]	[ 0]
'yyyyy'	[ 0]	[ 0]	[ 0]
'yyyyyz'	[ 0]	[ 0]	[ 0]
'yyyzzz'	[ 0]	[ 0]	[ 0]
'yyzzz'	[ 0]	[ 0]	[ 0]
'yzzzz'	[ 0]	[ 0]	[ 0]
'zzzzz'	[ 0]	[ 0]	[ 0]

Table 6: Logistic map identified using sparse representation.

	'x_{k+1}'	'r_{k+1}'
'1'	[ 0]	[ 0]
'x'	[ 0]	[ 0]
'r'	[ 0]	[1.0000]
'xx'	[ 0]	[ 0]
'xr'	[ 0.9993]	[ 0]
'rr'	[ 0]	[ 0]
'xxx'	[ 0]	[ 0]
'xxr'	[-0.9989]	[ 0]
'xrr'	[ 0]	[ 0]
'rrr'	[ 0]	[ 0]
'xxxx'	[ 0]	[ 0]
'xxxr'	[ 0]	[ 0]
'xxrr'	[ 0]	[ 0]
'xrrr'	[ 0]	[ 0]
'rrrr'	[ 0]	[ 0]
'xxxxx'	[ 0]	[ 0]
'xxxxr'	[ 0]	[ 0]
'xxxrr'	[ 0]	[ 0]
'xxrrr'	[ 0]	[ 0]
'xrrrr'	[ 0]	[ 0]
'rrrrr'	[ 0]	[ 0]

Table 7: Hopf normal form identified using sparse representation. Here  $u$  represents the bifurcation parameter  $\mu$ .

	'xdot'	'ydot'	'udot'
'1'	[ 0]	[ 0]	[ 0]
'x'	[ 0]	[ 0.9914]	[ 0]
'y'	[-0.9920]	[ 0]	[ 0]
'u'	[ 0]	[ 0]	[ 0]
'xx'	[ 0]	[ 0]	[ 0]
'xy'	[ 0]	[ 0]	[ 0]
'xu'	[ 0.9269]	[ 0]	[ 0]
'yy'	[ 0]	[ 0]	[ 0]
'yu'	[ 0]	[ 0.9294]	[ 0]
'uu'	[ 0]	[ 0]	[ 0]
'xxx'	[-0.9208]	[ 0]	[ 0]
'xxy'	[ 0]	[-0.9244]	[ 0]
'xxu'	[ 0]	[ 0]	[ 0]
'xyy'	[-0.9211]	[ 0]	[ 0]
'xyu'	[ 0]	[ 0]	[ 0]
'xuu'	[ 0]	[ 0]	[ 0]
'yyy'	[ 0]	[-0.9252]	[ 0]
'yyu'	[ 0]	[ 0]	[ 0]
'yuu'	[ 0]	[ 0]	[ 0]
'uuu'	[ 0]	[ 0]	[ 0]
'xxxx'	[ 0]	[ 0]	[ 0]
'xxxxy'	[ 0]	[ 0]	[ 0]
'xxxu'	[ 0]	[ 0]	[ 0]
'xxyy'	[ 0]	[ 0]	[ 0]
'xxyu'	[ 0]	[ 0]	[ 0]
'xxuu'	[ 0]	[ 0]	[ 0]
'xyyy'	[ 0]	[ 0]	[ 0]
'xyyu'	[ 0]	[ 0]	[ 0]
'xyuu'	[ 0]	[ 0]	[ 0]
'xuuu'	[ 0]	[ 0]	[ 0]
'yyyy'	[ 0]	[ 0]	[ 0]
'yyyu'	[ 0]	[ 0]	[ 0]
'yyuu'	[ 0]	[ 0]	[ 0]
'yuuu'	[ 0]	[ 0]	[ 0]
'uuuu'	[ 0]	[ 0]	[ 0]
'xxxxx'	[ 0]	[ 0]	[ 0]
'xxxxxy'	[ 0]	[ 0]	[ 0]
'xxxxu'	[ 0]	[ 0]	[ 0]
'xxxxyy'	[ 0]	[ 0]	[ 0]
'xxxxyu'	[ 0]	[ 0]	[ 0]
'xxxuu'	[ 0]	[ 0]	[ 0]
'xxyyy'	[ 0]	[ 0]	[ 0]
'xxyyu'	[ 0]	[ 0]	[ 0]
'xxyuu'	[ 0]	[ 0]	[ 0]
'xxuuu'	[ 0]	[ 0]	[ 0]
'xyyyy'	[ 0]	[ 0]	[ 0]
'xyyyu'	[ 0]	[ 0]	[ 0]
'xyyuu'	[ 0]	[ 0]	[ 0]
'xyuuu'	[ 0]	[ 0]	[ 0]
'xuuuu'	[ 0]	[ 0]	[ 0]
'yyyyy'	[ 0]	[ 0]	[ 0]
'yyyyyu'	[ 0]	[ 0]	[ 0]
'yyyuu'	[ 0]	[ 0]	[ 0]
'yyuuu'	[ 0]	[ 0]	[ 0]
'yuuuu'	[ 0]	[ 0]	[ 0]
'uuuuu'	[ 0]	[ 0]	[ 0]