Numerical Methods for Partial Differential Equations

WILEY

RESEARCH ARTICLE

Compressive-Sensing-Assisted Mixed Integer Optimization for Dynamical System Discovery With Highly Noisy Data

Tony Shi[1] | Mason Ma[1] | Hoang Tran[2] | Guannan Zhang[2]

[1]Manufacturing Intelligence and Dynamic Systems Laboratory, University of Tennessee Knoxville, Knoxville, Tennessee, USA | [2]Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee, USA

Correspondence: Guannan Zhang (zhangg@ornl.gov)

Received: 16 February 2023 | **Revised:** 31 May 2024 | **Accepted:** 12 November 2024

Keywords: compressive sensing | dynamical systems | machine learning | mixed-integer optimization | model discovery

ABSTRACT

The identification of governing equations for dynamical systems is an everlasting challenge for the fundamental research in science and engineering. Machine learning has exhibited great success in learn and predicting dynamical systems from data. However, the fundamental challenges still exist: discovering the exact governing equations from highly noisy data. In the present work, we propose a compressive sensing-assisted mixed integer optimization (CS-MIO) method to make a step forward from a modern discrete optimization lens. In particular, we first formulate the problem into a mixed integer optimization model. The discrete optimization nature of the model leads to exact variable selection by means of cardinality constraint, and thereby powerful capability of exact discovery of governing equations from noisy data. Such capability is further enhanced by incorporating compressive sensing and regularization techniques for highly noisy data and high-dimensional problems. The case studies on classical dynamical systems have shown that CS-MIO can discover the exact governing equations from large-noise data, with up to two orders of magnitude larger noise compared with state-of-the-art methods. We also show its effectiveness for high-dimensional dynamical system identification through the chaotic Lorenz 96 system.

1 | Introduction

Governing equations of the ubiquitous dynamical systems are of critical significance to shape our comprehension of the physical world. the traditional regime of obtaining the equations respects the mathematical or physical derivations following the first principles, including conservation laws, mathematical symmetry, and invariants. This paradigm, however, might be intractable for dealing with many complex phenomena. With the availability of large datasets due to the advances of sensors and technologies, a new paradigm of discovering governing equations from data has evolved. Machine learning plays the pivotal role under this paradigm with a wide scope of methods, including symbolic regression [1, 2], Gaussian processes [3], deep neural network [4–12], Bayesian inference [13].

Even though neural networks have been proved to be an effective tool in learning and predicting trajectories of dynamical systems, it is often challenging to extract new physical laws out of neural network models. Thus, this work focuses on another

thrust of data-driven discovery of governing equations by exploiting sparse regression approaches [14–19]. Studies along this path typically construct a large library of candidate terms and eventually transform into a sparse regression problem, grounded on the realistic assumption that only parsimonious terms are active in the governing equations. The breakthrough work by [14] introduced a novel architecture called Sparse Identification of Nonlinear Dynamics (SINDy), which uses a sequential threshold least squares (or ridge regression [20]) algorithm to advocate sparsity. The SINDy framework is impressive for its succinct but useful rationale; that is, the sparsity is essentially incurred by the penalty on coefficients. In this regard, studies have been conducted from many perspectives, including the Lasso-based approach with a dictionary of partial derivatives [21], $\ell_{2,1}$-norm for data with highly corrupted segments [22], weak SINDy and discretization accounting for white noise [23], integral formulation of the differential equation [24], weak formulation with the orthogonal matching pursuit [25] and the compressed sensing technique [26], to name a few. However, these methods perform term selection essentially by imposing a penalty on coefficients, subject to which they are usually sensitive to noise and unable to control the exact level of sparsity for governing equations.

From the perspective of discrete optimization, this sparse regression problem can be formulated as a Mixed Integer Optimization (MIO) model, which is to identify a combination of k terms from a pool of p candidates and simultaneously regress the coefficients. This ℓ_0-norm constrained MIO problem is non-convex and $\mathcal{NP}$-hard [27], corresponding to the best subset selection problem in the statistics community [28, 29]. The $\mathcal{NP}$-hardness of the problem has contributed to the belief that discrete optimization problems were intractable [30]. For this reason, plenty of impressive sparsity-promoting techniques have focused on computationally feasible algorithms for solving the approximations, including Lasso [31], Elastic-net [32], non-convex regularization [33, 34] and stepwise regression [35]. These approximations induce obscure sparsity via regularizations that often include a large set of active terms (many are correlated terms, and the coefficients are shrunken to zero to avoid overfitting) in order to deliver good prediction. That is, regularization is used for both variable selection and shrinkage. In contrast, the MIO-based exact method allows one to control the exact level of sparsity via setting the value of k. When the MIO based exact method decides to select a term, it purely takes it in without any shrinkage on the coefficients, thereby draining the effect of its correlated terms [29]. Indeed, there is nothing more important than correct term selection in the identification of governing equations. Although existing methods lean heavily on the sparsity-promoting parameters to achieve indirect terms selection, domain-educated researchers and practitioners actually might have an intuition for the ground truth k. This motivates us in present work to enable independent and direct terms selection and coefficient estimation for solving the sparse regression problem in governing equations identification.

In this paper, we propose a discrete optimization-based method for the exact recovery of governing equations under large noise. Our method takes advantage of the nature of discrete optimization in the means of cardinality constraints for terms selection and is able to separately control the exact sparsity of the governing equations and estimate the associated coefficients. The powerful capability of term selection is the cornerstone for exact recovery under large noise in the data and is further enhanced by combining compressive sensing and regularization techniques for large noise and high-dimensional problems. We demonstrate the capability of our method with a wide variety of examples from [14], including the chaotic Lorenz 3 system, the fluid dynamics of vortex shedding behind a cylinder, and two dynamical systems with bifurcations. In addition, we test on the famous high-dimensional Lorenz 96 system. Our results show that the proposed method can recover exact governing equations with up to two orders of magnitude larger noise compared with the state-of-the-art method. This shows that the modern discrete optimization is significantly effective for identifying governing equations from noisy and high-dimensional data.

2 | Problem Setting

In this section, we describe the problem setting of data-driven discovery of dynamical system. In addition, we introduce the highly noisy data setting for the problem.

2.1 | Data-Driven Discovery of Dynamical Systems

We introduce the data-driven dynamical system discovery problem from the perspective of sparse recovery [14]. We define $\mathcal{J} = \{1, 2, \cdots, J\}$ for any $J \in \mathbb{Z}_+$ throughout this paper. We consider the following dynamical system consisting of J state variables, that is,

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{1}$$

where the vector $\mathbf{x}(t) = [x_1(t), \cdots, x_J(t)] \in \mathbb{R}^{1\times J}$ denotes the state of a system at time t, and $\mathbf{f}(\mathbf{x}(t)) = [f_1(\mathbf{x}(t)), \ldots, f_J(\mathbf{x}(t))] \in \mathbb{R}^{1\times J}$ represents the forcing term with $f_j(\mathbf{x}(t))$ being the forcing term of the j-th state variable x_j for $j \in \mathcal{J}$. A dictionary $\boldsymbol{\theta}(\mathbf{x})$ consisting of a total of P terms, denoted by

$$\boldsymbol{\theta}(\mathbf{x}) := [\theta_1(\mathbf{x}), \theta_2(\mathbf{x}), \ldots, \theta_P(\mathbf{x})] \tag{2}$$

which consists of nonlinear combinations of state $\mathbf{x}$ that can be candidate terms in $\mathbf{f}$. For example, $\boldsymbol{\theta}(\mathbf{x})$ may consist of polynomial and trigonometric terms of $\mathbf{x}$. Each term of $\boldsymbol{\theta}(\mathbf{x})$ represents a candidate term for the right-hand-side of Equation (1).

This work is based upon two assumptions that were also used in [14]. The first is that we assume the right-hand-side $\mathbf{f}$ in Equation (1) lives in the function space expanded by the dictionary $\boldsymbol{\theta}(\mathbf{x})$ in Equation (2). In other words, there exists a coefficient matrix $\boldsymbol{\Xi} := [\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_J] \in \mathbb{R}^{P\times J}$ such that

$$f_j(\mathbf{x}) = \boldsymbol{\theta}(\mathbf{x}) \cdot \boldsymbol{\xi}_j, \quad \text{for } j \in \mathcal{J} \tag{3}$$

We remark that the definition of the dictionary $\boldsymbol{\theta}(\mathbf{x})$ would require domain knowledge about the specific scientific problem in order to ensure that all the terms in $\mathbf{f}(\mathbf{x})$ are included in $\boldsymbol{\theta}(\mathbf{x})$. Since this work is to study sparse recovery of $\mathbf{f}(\mathbf{x})$, how to properly choose $\boldsymbol{\theta}(\mathbf{x})$ to ensure Equation (3) is out of the scope of this paper. The second assumption is that the forcing term $\mathbf{f}$ consists

of only a few terms, that is, it is very sparse in the function space expanded by the dictionary $\boldsymbol{\theta}(\mathbf{x})$, regardless of the dimensionality J. Specifically, to indicate the presence of each term of $\boldsymbol{\theta}(\mathbf{x})$ in the right-hand side $\mathbf{f}$, we introduce the following indicator matrix

$$\boldsymbol{\Gamma} := [\boldsymbol{\gamma}_1, \cdots, \boldsymbol{\gamma}_J] = \begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1J} \\ \vdots & \ddots & \vdots \\ \gamma_{P1} & \cdots & \gamma_{PJ} \end{bmatrix} \tag{4}$$

$$\gamma_{pj} := \begin{cases} 1, & \text{if } f_j(\mathbf{x}) \text{ includes } \theta_p(\mathbf{x}), \\ 0, & \text{otherwise,} \end{cases}$$

where $\boldsymbol{\gamma}_j = (\gamma_{1j}, \ldots, \gamma_{Pj})^T \in \mathbb{B}^{P\times 1}$ and $\mathbb{B}$ is the Boolean domain $\mathbb{B} = \{0, 1\}$. Moreover, we denote the number of active terms in $\mathbf{f} = [f_1, \ldots, f_J]$ by a vector

$$\boldsymbol{k} = [k_1, k_2, \ldots, k_J] \tag{5}$$

where k_j is the number of non-zeros in $\boldsymbol{\gamma}_j$. When the dynamical system satisfies the above two assumptions, Equation (1) can be written as

$$\dot{\mathbf{x}}(t) = \boldsymbol{\theta}(\mathbf{x}(t))(\boldsymbol{\Gamma}\circ\boldsymbol{\Xi}) \tag{6}$$

where $\boldsymbol{\Gamma}\circ\boldsymbol{\Xi}$ is the element-wise product (Hadamard product) of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Xi}$.

2.2 | The Noisy Data

The state $\mathbf{x}$ and its time derivative $\dot{\mathbf{x}}$ can be measured and collected at a series of time instants $t_1, t_2, \ldots, t_N$. With the measurements of $\mathbf{x}(t)$ and $\dot{\mathbf{x}}(t)$, we will be given two data matrices, denoted by $\mathbf{X} \in \mathbb{R}^{N\times J}$ and $\dot{\mathbf{X}} \in \mathbb{R}^{N\times J}$, of the following forms,

$$\mathbf{X} = \begin{bmatrix} x_1(t_1) & \cdots & x_J(t_1) \\ x_1(t_2) & \cdots & x_J(t_2) \\ \vdots & \ddots & \vdots \\ x_1(t_N) & \cdots & x_J(t_N) \end{bmatrix}, \quad \text{and} \quad \dot{\mathbf{X}} = \begin{bmatrix} \dot{x}_1(t_1) & \cdots & \dot{x}_J(t_1) \\ \dot{x}_1(t_2) & \cdots & \dot{x}_J(t_2) \\ \vdots & \ddots & \vdots \\ \dot{x}_1(t_N) & \cdots & \dot{x}_J(t_N) \end{bmatrix} \tag{7}$$

where the measurements of $\dot{\mathbf{x}}(t)$ can be numerically approximated using the data $\mathbf{X}$ if $\dot{\mathbf{x}}(t)$ is not directly measurable. In practice, the measurements $\mathbf{X}$ and $\dot{\mathbf{X}}$ are usually corrupted with random noises, so that the matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Xi}$ in Equation (6) need to be recovered with noisy data, denoted by

$$\mathbf{X}^{\text{noisy}} := \mathbf{X} + \boldsymbol{\mathcal{U}} \quad \text{and} \quad \dot{\mathbf{X}}^{\text{noisy}} := \dot{\mathbf{X}} + \boldsymbol{\mathcal{V}} \tag{8}$$

where $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{N\times J}$ and $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{N\times J}$ are additive noise.

Evaluating the library $\boldsymbol{\theta}(\mathbf{x})$ at each data point in $\mathbf{X}^{\text{noisy}}$, we can construct an augmented data matrix, denoted by $\boldsymbol{\Theta}(\mathbf{X}^{\text{noisy}})$, consisting of candidate nonlinear functions of the columns of $\mathbf{X}^{\text{noisy}}$. For ease of notation, we use $\boldsymbol{\Theta}^{\text{noisy}}$ instead of $\boldsymbol{\Theta}(\mathbf{X}^{\text{noisy}})$ in the following. Since there are P terms in $\boldsymbol{\theta}(\mathbf{x})$, the matrix $\boldsymbol{\Theta}^{\text{noisy}} \in \mathbb{R}^{N\times P}$ is represented by

$$\boldsymbol{\Theta}^{\text{noisy}} := [\theta_1(\mathbf{X}^{\text{noisy}}), \cdots, \theta_P(\mathbf{X}^{\text{noisy}})] \tag{9}$$

Similar to the standard SINDy method in [14], we assume the entries of the noise matrices $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{V}}$ in Equation (8) are independent and identically distributed (i.i.d.) Gaussian random variables with zero mean and standard deviation σ. In this work, we are particularly interested in the scenario with relatively large standard deviation of the noises, that is, low signal-to-noise ratio. Details about the definition of the noises are given in Section 4.

The goal of sparse recovery of the dynamical system in Equation (6) is to correctly identify $\boldsymbol{\Gamma}$ and calculate the non-zero elements of $\boldsymbol{\Xi}$ from measurement data of $\mathbf{x}$ and/or $\dot{\mathbf{x}}$. As discussed in Section 1, existing studies on sparse recovery of dynamical systems, for example [14–19, 26]" perform term selection and promote sparsity by imposing penalties on the coefficients. In other words, these methods try to recover the product $\boldsymbol{\Gamma}\circ\boldsymbol{\Xi}$ as a whole. Despite the success of these methods, they are usually very sensitive to the noise in the measurement data. When the signal-to-noise ratio is low, the method like SINDy may fail to identify the correct terms of $\mathbf{f}$ in the dictionary $\boldsymbol{\theta}(\mathbf{x})$. The motivation of this work is to recover $\boldsymbol{\Gamma}$ and $\boldsymbol{\Xi}$ separately, where $\boldsymbol{\Gamma}$ is recovered by solving a compressive-sensing-assisted mixed integer optimization problem in order to identify the correct terms in the case of having data with low signal-to-noise ratio.

3 | The Compressive Sensing-Assisted Mixed Integer Optimization Method

This section describes the details of the proposed method. Specifically, a linear regression model subject to sparsity constraints for Equation (6) can be set up as follows,

$$\min_{\boldsymbol{\Gamma},\boldsymbol{\Xi}} \left\| \dot{\mathbf{X}}^{\text{noisy}} - \boldsymbol{\Theta}^{\text{noisy}}(\boldsymbol{\Gamma}\circ\boldsymbol{\Xi}) \right\|_F^2, \quad s.t.\ \boldsymbol{\Gamma}^T\boldsymbol{e} \le \boldsymbol{k}^{\max} \tag{10}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\boldsymbol{e} \in \mathbb{R}^{P\times 1}$ is a vector with all the entries equal to one, such that the product $\boldsymbol{\Gamma}^T\boldsymbol{e}$ are exactly the cardinality constraints to indicate the active terms in each equation, and $\boldsymbol{k}^{\max} = [k_1^{\max}, \ldots, k_J^{\max}]^T$ consists of the maximum allowable sparsity for the J components. The main idea of the CS-MIO method is to separately identify the physical terms (i.e., $\boldsymbol{\Gamma}$) and the corresponding coefficients (i.e., $\boldsymbol{\Xi}$) in a two-stage manner. The indicator matrix $\boldsymbol{\Gamma}$ is determined by mixed integer optimization. Once $\boldsymbol{\Gamma}$ is chosen, we can estimate the corresponding components of $\boldsymbol{\Xi}$ using the standard least-squares method. Nevertheless, when the size of the original dictionary, that is, the number of columns of $\boldsymbol{\Gamma}$, is large, it is computationally intractable for the state-of-the-art MIO algorithms. To resolve this issue, we propose to use compressive sensing, that is, ℓ_1 minimization, to reduce the size of the dictionary to the extent that can be handled by MIO algorithms.

In the rest of this section, we take the j-th component of $\mathbf{x}$ in Equation (1) as an example in the following derivation, which means we intend to use the j-th column of the data matrix $\dot{\mathbf{X}}^{\text{noisy}}$ to infer the j-th columns of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Xi}$. For notational simplicity, we omit the subscript j and use $\dot{\mathbf{x}}^{\text{noisy}}$, $\boldsymbol{\gamma}$, $\boldsymbol{\xi}$ to represent the j-column of $\dot{\mathbf{X}}^{\text{noisy}}$, $\boldsymbol{\Gamma}$ and $\boldsymbol{\Xi}$, respectively.

3.1 | Compressive Sensing for Reducing the Size of the Dictionary θ^{noisy}

The goal of this subsection is to reduce the size of the original dictionary $\boldsymbol{\Theta}^{\text{noisy}}$ in Equation (9), so that the modern integer optimization solvers, for example, `CPLEX` or `GUROBI`, can be used to determine the indicator vector $\boldsymbol{\gamma}$. To this end, we first solve the following ℓ_1 minimization problem:

$$\boldsymbol{\xi}^{\text{CS}} = \arg\min_{\xi} ||\dot{\mathbf{x}}^{\text{noisy}} - \boldsymbol{\Theta}^{\text{noisy}}\boldsymbol{\xi}||_2^2 + \lambda_1||\boldsymbol{\xi}||_1 \tag{11}$$

where $\|\cdot\|_1$ is the ℓ_1 norm and $\boldsymbol{\xi}^{\text{CS}}$ is the recovered coefficient by the ℓ_1 minimization. In this paper, we used LARS algorithm in [36] for ℓ_1 minimization. Then, we define a subset, denoted by $\mathcal{S}$, of $\mathcal{P} = [1, 2, \ldots, P]$ based on the magnitude of the components of $\boldsymbol{\xi}^{\text{CS}}$, that is,

$$\mathcal{S} := \left\{ i \in \mathcal{P} \;\middle|\; |\xi_i^{\text{CS}}| \geq \varepsilon,\ \xi_i^{\text{CS}} \in \boldsymbol{\xi}^{\text{CS}} \right\} \tag{12}$$

where the threshold $\varepsilon > 0$ is chosen such that the reduced dictionary can be handled by the state-of-the-art MIO algorithm. We denote the reduced dictionary, the reduced indicator vector, and the reduced coefficient vector by

$$\begin{aligned} \boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}} &:= \{\theta_i(\mathbf{X}^{\text{noisy}}) \in \boldsymbol{\Theta}^{\text{noisy}} \mid i \in \mathcal{S}\} \\ \boldsymbol{\gamma}_{\mathcal{S}} &:= \{\gamma_i \in \boldsymbol{\gamma} \mid i \in \mathcal{S}\} \\ \boldsymbol{\xi}_{\mathcal{S}} &:= \{\xi_i \in \boldsymbol{\xi}, \mid i \in \mathcal{S}\} \end{aligned} \tag{13}$$

respectively. We emphasize that the recovered coefficients $\boldsymbol{\xi}^{\text{CS}}$ in solving the ℓ_1 minimization problem are not used to determine the final estimation of the coefficients. Instead, it is only used to help screen and narrow down the range of candidate terms for high-dimensional problems with large P.

3.2 | Mixed-Integer Optimization for Determining the Indicator $\boldsymbol{\gamma}_{\mathcal{S}}$

We start from converting the problem in Equation (10) into an MIO problem. Then, the MIO problems constrained by a given sparsity can be written as

$$\min_{\boldsymbol{\xi}_{\mathcal{S}}, \boldsymbol{\gamma}_{\mathcal{S}}} \left\| \dot{\mathbf{x}}^{\text{noisy}} - \boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}}(\boldsymbol{\gamma}_{\mathcal{S}} \circ \boldsymbol{\xi}_{\mathcal{S}}) \right\|_2^2 + \lambda_2||\boldsymbol{\xi}_{\mathcal{S}}||_2^2 \tag{P0}$$

$$\text{s.t. } \|\boldsymbol{\xi}_{\mathcal{S}}\|_\infty \leq B \tag{14}$$

$$\boldsymbol{\gamma}_{\mathcal{S}}^T \mathbf{e} = k \tag{15}$$

where k denotes the sparsity of $\gamma_{\mathcal{S}}$ and B is the upper bound of the coefficient $\boldsymbol{\xi}_{\mathcal{S}}$, and the ℓ_2 regularization term $\lambda_2||\boldsymbol{\xi}_{\mathcal{S}}||_2^2$ is commonly added to help alleviate the influence of the measurement noises on the MIO optimization. P_0 can be equivalently reformulated as

$$\min_{\boldsymbol{\xi}_{\mathcal{S}}, \boldsymbol{\gamma}_{\mathcal{S}}} \left\| \dot{\mathbf{x}}^{\text{noisy}} - \boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}}\boldsymbol{\xi}_{\mathcal{S}} \right\|_2^2 + \lambda_2||\boldsymbol{\xi}_{\mathcal{S}}||_2^2 \tag{P1}$$

$$\text{s.t. } -B\boldsymbol{\gamma}_{\mathcal{S}} \leq \boldsymbol{\xi}_{\mathcal{S}} \leq B\boldsymbol{\gamma}_{\mathcal{S}} \tag{16}$$

$$\boldsymbol{\gamma}_{\mathcal{S}}^T \mathbf{e} = k \tag{17}$$

where Equation 16 refers to the element-wise inequalities, namely, for $i \in \mathcal{S}$, when element $\gamma_i = 0$, $\xi_i = 0$; otherwise $\xi_i \in [-B, B]$. Formulation P1 can be directly solved by MIO solvers like `CPLEX`.

Remark 1. (Using normalized data for MIO). The scales of different components of the dynamical system could be significantly different, which can affect the performance of the MIO solver in determining the optimal $\boldsymbol{\gamma}_{\mathcal{S}}$. To resolve this issue, we standardize the data $\boldsymbol{\Theta}^{\text{noisy}}$ and $\dot{\boldsymbol{x}}^{\text{noisy}}$, and use the standardized data in MIO.

Remark 2. We emphasize that splitting the coefficient of $\boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}}$ into $\boldsymbol{\gamma}_{\mathcal{S}}$ and $\boldsymbol{\xi}_{\mathcal{S}}$ is to indicate that the goal of solving the MIO is only to determine $\boldsymbol{\gamma}_{\mathcal{S}}$, that is, identify the correct terms in the reduced dictionary $\boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}}$. Even though an MIO algorithm will also provide an estimate of $\boldsymbol{\xi}_{\mathcal{S}}$, we will not use the estimate as our final solution.

The goal of this subsection is to determine $\boldsymbol{\gamma}_{\mathcal{S}}$ by solving the MIO problem in Equation (P1). However, there are two hyperparameters, that is, the sparsity k and the ℓ_2-norm weight λ_2, that could significantly affect the outcome of the MIO solver. To address this issue, we perform a grid search with a cross-validation metric to tune the two hyperparameters and obtain $\boldsymbol{\gamma}_{\mathcal{S}}$.

We first define a tensor grid of (k, λ_2). The grid for k is easily defined as $\{1, 2, \ldots, k^{\max}\}$ based on the maximum allowable sparsity $k^{\max}$. The upper bound for λ_2, denoted by $\lambda_2^{\max}$, is defined by the norm

$$\lambda_2^{\max} := \|(\boldsymbol{\Theta}^{\text{noisy}})^\top \dot{\mathbf{x}}^{\text{noisy}}\|_\infty$$

This is followed by setting a small ratio r of $\lambda_2^{\max}$ to set the minimum allowed value $\lambda_2^{\min}$, that is, $\lambda_2^{\min} = r\lambda_2^{\max}$. Empirically, if $N > P$, we set $r = 0.0001$; otherwise $r = 0.01$. Afterwards, we uniformly sample m values from interval $[\log(\lambda_2^{\min}), \log(\lambda_2^{\max})]$, where m is practically set to 50 or 100. Then by taking the exponential of the sampled values, we obtain a set of ℓ_2-norm weight, denoted by $\boldsymbol{\Lambda} = \{\lambda_2^1, \ldots, \lambda_2^m\}$.

We next perform the cross-validation to choose the best hyperparameters from the tensor grid of (k, λ_2). Specifically, we evenly partition the data set $\{\dot{\mathbf{x}}^{\text{noisy}}, \boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}}\}$ with a total of N measurements into T disjoint subsets, denoted by $\{\dot{\mathbf{x}}_1^{\text{noisy}}, \boldsymbol{\Theta}_{\mathcal{S},1}^{\text{noisy}}\}, \ldots, \{\dot{\mathbf{x}}_T^{\text{noisy}}, \boldsymbol{\Theta}_{\mathcal{S},T}^{\text{noisy}}\}$, respectively. For one subset $\{\dot{\mathbf{x}}_t^{\text{noisy}}, \boldsymbol{\Theta}_{\mathcal{S},t}^{\text{noisy}}\}$ and one pair of (k, λ_2), the error of the MIO solution is defined by

$$e_t(k, \lambda_2) := \left\| \dot{\mathbf{x}}_t^{\text{noisy}} - \boldsymbol{\Theta}_{\mathcal{S},t}^{\text{noisy}} (\boldsymbol{\gamma}_{\mathcal{S},t} \circ \boldsymbol{\xi}_{\mathcal{S},t}) \right\|_2^2 \tag{18}$$

where $\boldsymbol{\gamma}_{\mathcal{S},t}$ and $\boldsymbol{\xi}_{\mathcal{S},t}$ are obtained by solving the MIO problem in Equation (P1) using the complementary data set $\{\dot{\boldsymbol{x}}^{\text{noisy}} \setminus \dot{\boldsymbol{x}}_t^{\text{noisy}}, \boldsymbol{\Theta}_{\mathcal{S}}^{\text{noisy}} \setminus \boldsymbol{\Theta}_{\mathcal{S},t}^{\text{noisy}}\}$. The errors for other subsets and choices of k, λ_2 can be obtained similarly. Then the total error for the pair (k, λ_2) is defined by

$$\mathcal{E}(k, \lambda_2) := \sum_{t=1}^{T} e_t(k, \lambda_2) \tag{19}$$

and the best hyperparameters are obtained by

$$(k^*, \lambda_2^*) = \arg\min_{(k, \lambda_2)} \mathcal{E}(k, \lambda_2) \tag{20}$$

The final step in this subsection is to solve the MIO problem with the best hyperparameters (k^*, λ_2^*) using dataset $\{\dot{\mathbf{x}}^{\text{noisy}}, \boldsymbol{\Theta}_S^{\text{noisy}}\}$ to obtain the optimized indicator vector, denoted by $\boldsymbol{\gamma}_S^*$.

After the optimal $\boldsymbol{\gamma}_S^*$ is determined using the MIO method, we use the standard least-squares approach to estimate the coefficient $\boldsymbol{\xi}_S$. In this case, we use the original data, not the standardized data used in the MIO method, to solve the following least-squares problem

$$\boldsymbol{\xi}_S^* = \arg\min_{\boldsymbol{\xi}_S} \|\dot{\mathbf{x}}^{\text{noisy}} - (\boldsymbol{\Theta}^{\text{noisy}}\boldsymbol{\gamma}_S^*)\,\boldsymbol{\xi}_S\|_2^2 \tag{21}$$

where the matrix $\boldsymbol{\Theta}^{\text{noisy}}\boldsymbol{\gamma}_S^*$ only contains the columns of $\boldsymbol{\Theta}^{\text{noisy}}$ identified by $\boldsymbol{\gamma}_S^*$.

3.3 | Summary of the CS-MIO Algorithm

We summarize the proposed CS-MIO method in Algorithm 1. The CS-MIO algorithm is general by combining the capability of expected sparsity control with physical term selection and coefficient estimation. The key of the CS-MIO algorithm is on solving the MIO formulation. In the present study, we fully take advantage of state-of-the-art algorithms in modern optimization solvers `CPLEX` for solving the MIO problem. With appropriate settings for the time limit and optimality gap, the solver returns the optimal solution. Even if we terminate the algorithm early, it still provides a solution with suboptimality guaranteed. We will discuss the details of parameter settings for the optimization solver in the following experimental studies.

ALGORITHM 1 | The CS-MIO algorithm.

Input : The noisy data $\mathbf{X}^{\text{noisy}}$, $\dot{\mathbf{X}}^{\text{noisy}}$

Construct matrix $\boldsymbol{\Theta}^{\text{noisy}}$ by evaluating $\theta(\mathbf{x})$ at the data points in $\mathbf{X}^{\text{noisy}}$;
Standardize the columns of $\boldsymbol{\Theta}^{\text{noisy}}$ and $\dot{\mathbf{X}}^{\text{noisy}}$ to have zero means and unit variance;
for $j \in \mathcal{J}$ **do**
  if $P > P_{\max}$ **then** `/* Compressive sensing-based dictionary reduction */`
    Construct the reduced dictionary $\mathcal{S}$ in Equation (13) by solving Equation (12)
  end
  for $k = 1, 2, \ldots, k^{\max}$ **do** `/* MIO for determining` $\boldsymbol{\gamma}_S$ `in Equation (P1) */`
    Construct a grid $\boldsymbol{\Lambda}$ of λ_2 in Equation (P1);
    Divide $\{\dot{\mathbf{x}}^{\text{noisy}}, \boldsymbol{\Theta}_S^{\text{noisy}}\}$ into T disjoint subsets;
    for $\lambda_2 \in \boldsymbol{\Lambda}$ **do**
      for $t = 1, \ldots, T$ **do**
        Solve the MIO problem in Equation (P1) using k and λ_2;
        Compute the error $e_t(k, \lambda_2)$ in Equation (18);
      end
      Compute total error $\mathcal{E}(k, \lambda_2)$ in Equation (19);
    end
  end
  Find the best hyperparameters $(k^*, \lambda_2^*) = \arg\min_{(k,\lambda_2)} \mathcal{E}(k, \lambda_2)$
  Identify the optimal indicator $\boldsymbol{\gamma}_S^*$ by solving the problem (P1) using (k^*, λ_2^*);
  Determine the optimal coefficient $\boldsymbol{\xi}_S^*$ by solving the problem in Equation~(21);
  Set $\boldsymbol{\gamma}_S^*$ and $\boldsymbol{\xi}_S^*$ as the j-th column of $\boldsymbol{\Gamma}^*$ and $\boldsymbol{\Xi}^*$, respectively;
end
Return $\dot{\mathbf{x}} = \theta(\mathbf{x})(\boldsymbol{\Gamma}^* \circ \boldsymbol{\Xi}^*)$

4 | Numerical Experiments

We demonstrate the effectiveness of the proposed CS-MIO method for recovery of governing equations from large noise data. We use several classical dynamical systems in [14] as the testing problems, including the chaotic Lorenz 3 system, vortex shedding after a cylinder, and bifurcation dynamical systems like the Hopf normal form and logistic map. In addition, we also study the high-dimensional Lorenz 96 system. We compare CS-MIO with the state-of-the-art method SINDy, specifically the Python version solver PySINDy [37, 38]. For all the example systems, the experiments are deployed on a mobile workstation with an Intel(R) Xeon(R) W-10885M CPU @ 2.40 GHz, 128 GB of memory, and a 64-bit Windows 10 Pro operating system for workstations.

Remark 3. (Reproducibility). The algorithm of CS-MIO is implemented in Python. The code is publicly available at https://github.com/utk-ideas-lab/CS-MIO. All the numerical results presented in this section can be exactly reproduced using the code on Github.

4.1 | Experimental Settings

We first give the experimental settings throughout the case studies. To better measure the noise level and the anti-noise capability of the method, we consider the signal-to-noise ratio (SNR). In this work, we consider the averaged SNR of the dynamical system consisting of a set of J governing equations,

$$\text{SNR} := \frac{1}{J}\sum_{j=1}^{J}\frac{\text{Var}(S_j)}{\text{Var}(\mathcal{Z}_j)} \tag{22}$$

where $S_j \in \{X_j, \dot{X}_j\}$ is the signal data, that is, the j-th column of matrices $\mathbf{X}$ or $\dot{\mathbf{X}}$, and $\mathcal{Z}_j \in \{\mathcal{U}_j, \mathcal{V}_j\}$ are the additive Gaussian noise, that is, the j-th column of matrices $\mathcal{U}$ or $\mathcal{V}$ in Equation (8). The SNR gives a good indicator to assess the ability of methods to withstand noise in the data. Smaller SNR indicates a system with larger noise. We examine the anti-noise capability of the methods over a wide range of SNRs for the studied examples. In the following cases, we impose the below two types of noise by considering the signal that can be measured.

- *Type 1 Noise*: Both the state variables $\mathbf{x}$ and time derivatives $\dot{\mathbf{x}}$ can be measured; Gaussian noise is added to $\dot{\mathbf{x}}$.
- *Type 2 Noise*: Only state variables $\mathbf{x}$ can be measured. Gaussian noise is added to $\mathbf{x}$. The time derivatives $\dot{\mathbf{x}}$ are computed by total variation derivative (TVD) [39].

We use state-of-the-art algorithm in the modern optimization solver `CPLEX` 20.1 with the Python interface `docplex` for solving the MIO problem. Unless specifically mentioned, we use up to fifth-order total-degree polynomials throughout the examples to define the initial dictionary. The choice of the upper bound B in Equation (14) impacts the strength of the MIO formulation, especially when looking for good lower bounds. $B \in \mathbb{R}$ is a sufficiently large constant such that $B \geq \|\xi^*\|_\infty$. This setting is, however, not applicable because the ξ^* is not known a prior. Some methods have been studied to set B values by finding the upper bound of ξ^* using data-driven manners such as cumulative coherence function and solving convex optimization problems [29]. In this paper, we use a looser upper bound $B = 1000$ for all the examples. Besides, we set the `timelimit` to be 600 s and the `mipgap` to be 0 for the invoked branch-and-cut algorithm in `CPLEX`. This refers to that if the branch-and-cut finds a solution within 600 s, it will be the optimal solution with zero gap; otherwise, the provided solution will be suboptimal, and its gap to the lower bound, and thus to the optima, will be clearly quantified.

The metrics for performance comparison. We evaluate the performance of the identification of $\boldsymbol{\Gamma}$ in Equation (10) by the number of exactly recovered equations of the target dynamical system, defined by

$$A(\boldsymbol{\Gamma}) := \sum_{j=1}^{J} \mathbf{1}_{\gamma_j = \gamma_j^\dagger}, \text{ with } \mathbf{1}_{\gamma_j = \gamma_j^\dagger} = \begin{cases} 1, & \text{if } \gamma_j := \gamma_j^\dagger \\ 0, & \text{if } \gamma_j \neq \gamma_j^\dagger \end{cases} \tag{23}$$

where $\gamma_j^\dagger$ is the ground truth and γ_j is recovered by a method. The exact recovery for the entire dynamical system occurs when $A(\boldsymbol{\Gamma}) = J$. When the exact $\boldsymbol{\Gamma}$ can be recovered, we evaluate the accuracy of the approximation of the coefficients in $\boldsymbol{\Xi}$ in Equation (10) by the differences between the approximate and the exact coefficients and trajectories.

4.2 | The Chaotic Lorenz 3 System

Consider the 3-dimensional chaotic Lorenz system governed by the following equations:

$$\dot{x} = \alpha(y - x) \tag{24}$$

$$\dot{y} = x(\rho - z) - y \tag{25}$$

$$\dot{z} = xy - \beta z \tag{26}$$

With $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$, the Lorenz 3 system performs chaotically. We generate the data using the initial point $(x, y, z) = (-8, 8, 27)$ with a time step $\Delta t = 0.001$ in $t \in [0, 60]$. A set of noise standard deviation σ is used to better quantify the spectrum of anti-noise capability of the methods. In particular, under Type 1 noise, Gaussian noise is added to $\dot{\mathbf{x}}$ with σ ranging from 1 to 3,000. SNR is computed by the added noise and $\dot{\mathbf{x}}$. When under Type 2 noise, the Gaussian noise is added to $\mathbf{x}$ with σ ranging from 0.01 to 20, and the SNR is computed by the added noise and $\mathbf{x}$. In this case, $\dot{\mathbf{x}}$ is smoothed using TVD in [39]. The comparison results of CS-MIO and PySINDy for both cases are presented in Tables 1a and 1b, respectively.

Table 1a,b show that CS-MIO significantly outperforms PySINDy in terms of the number of exactly recovered equations. Under Type 1 noise as shown in Table 1a, CS-MIO is able to exactly recover the governing equations with SNR as low as 0.047. Comparing to the SNR value of 4191.621 by PySINDy, this results in a tremendous difference of almost 100,000 times. Similar conclusions can be made under Type 2 noise as shown in Table 1b. It is noted in this case the noise added to $\mathbf{x}$ is no longer Gaussian after using numerical differentiation and is difficult to handle. Thus, the performance of both CS-MIO and PySINDy is downgraded to smaller SNRs.

Table 2a,b show the discovered equations by CS-MIO and PySINDy under Type 1 noise (at noise standard deviation to be 300) and Type 2 noise (at noise standard deviation to be 2), respectively. Obviously from these tables, PySINDy includes redundant false terms. On the contrary, CS-MIO identifies all and only the ground truth terms while remaining small deviations of the identified parameters from the ground truth. This can be seen from the trajectories of the discovered equations by both PySINDy and CS-MIO in Figure 1. Figure 1a shows the trajectory of the PySINDy identified system under Type 1 noise at $\sigma = 300$. It is seen the trajectory starts to deviate from the ground truth right at the beginning, shown by the red dot. In contrast, the trajectory of the CS-MIO identified system can coincide for a longer time well with the ground truth, as shown in Figure 1b. Additional numerical results on identification of the Lorenz-3 system using

TABLE 1 | Parameters and results in the case study 1.

(a) Results under Type 1 noise.

		The metric $A(\boldsymbol{\Gamma})$	
Noise std	**SNR**	**PySINDy**	**CS-MIO**
1	4191.621	**3**	**3**
10	41.921	2	**3**
50	1.677	1	**3**
100	0.419	0	**3**
200	0.105	0	**3**
300	0.047	0	**3**
500	0.017	0	2
1000	0.004	0	1
3000	0.001	0	0

(b) Results under Type 2 noise.

0.01	729427.159	**3**	**3**
0.05	29178.677	2	**3**
0.1	7295.616	2	**3**
0.5	292.848	1	**3**
1	73.981	0	**3**
2	19.255	0	**3**
5	3.926	0	2
10	1.733	0	1
20	1.184	0	0

CS-MIO method are shown in Figures A1, A2 and Tables A1, A2 in Appendix A.

4.3 | The Chaotic Lorenz 96 System

We demonstrate the effectiveness of the proposed CS-MIO method for high-dimensional problems using the Lorenz 96 dynamic system, which is defined as follows. For $j = 1, \ldots, J$,

$$\dot{x}_j = (x_{j+1} - x_{j-2})x_{j-1} - x_j + F \tag{27}$$

where x_j is the state variable and F is a forcing constant. Here it is assumed that $x_{-1} = x_{J-1}$, $x_0 = x_J$, $x_{J+1} = x_1$ and $J \geq 4$. In this study, we set $J = 96$. $F = 8$ is a common value known to cause chaotic behavior. We use the initial condition $\mathbf{x}(0) = \mathbf{1}$ with a small perturbation of 0.01 added to $x_1(0)$ to generate the dataset with a time step of $\Delta t = 0.01$ in $t \in [0, 600]$. We use second-order polynomials in CS-MIO for the Lorenz 96 system with 96 variables, which results in 4,752 polynomial terms. This leads to huge difficulties in dealing with the high dimension.

For this high-dimensional Lorenz 96 system, we use a compressive sensing approach as described in Equation (12) for pre-selecting a subset of at most $S = 100$ significant terms from 4,752 candidate terms. We use the `LassoLars` algorithm from the Python package `scikit-learn`. We set $\lambda_1 = 10^{-6}$, a very small regularization weight but capable of narrowing down thousands of terms to hundreds. Other settings remain as default. It is seen in most cases that the resulting subset of terms has a larger size than S. In this case, we order the nonzero terms in decreasing order of the absolute values of their coefficients and select the top S terms to form the preselected subset $\mathcal{S}$.

Tables 3a and 3b show that CS-MIO outperforms PySINDy under large noise in terms of the number of exactly recovered equations, that is, the metric $A(\mathbf{\Gamma})$, under both noise types. Note from $\sigma = 70$ under Type 1 noise and $\sigma = 1$ under Type 2 noise, CS-MIO fails to completely discover the 96 equations because the ℓ_1 regularization fails to include all the ground truth terms within the first 100 significant terms.

Figure 2 shows in the form of a Hovermöller plot the trajectory difference between the identified system and the ground truth. In particular, the CS-MIO-identified systems at $\sigma = 50$ in Table 3a and $\sigma = 0.8$ in Table 3b, are used to run simulations in the time interval $t \in [0, 10]$ with a time step $\Delta t = 0.01$ s. In the Hovermöller plot, the horizontal axis is the time, and the vertical axis refers to the index of the state variables. The differences between the state values of ground truth $x_j(t)$ and those of the identified system $\hat{x}_j(t)$, $\Delta x_j(t) = x_j(t) - \hat{x}_j(t)$ for $j \in \{1, 2, \cdots, 96\}$, are shown with different colors. The more white-colored areas indicate the trajectory of the identified system agrees better with the ground truth. For example, it is seen roughly at $t \in [0, 1]$ from Figure 2a, the identified system trajectory coincides well with the ground truth while the deviation starts to increase after that. We

TABLE 2 | Comparison of discovered equations by PySINDy and CS-MIO under (a) Type 1 noise at $\sigma = 300$; and (b) Type 2 noise at $\sigma = 2$. The CS-MIO method correctly identified all the terms in the Lorenz 3 system.

(a) Results under Type 1 noise with $\sigma = 300$.	
Ground Truth	$\dot{x} = -10x + 10y$
	$\dot{y} = 28x - y - xz$
	$\dot{z} = -\frac{8}{3}z + xy$
PySINDy	$\dot{x} = -6.62 - 13.62x + 11.80y + 0.38z - 0.14x^2 + 0.23xy + 0.11xz - 0.11y^2 - 0.05yz$
	$\dot{y} = 3.46 + 29.32x - 1.32y - 0.05z - 1.03xz$
	$\dot{z} = -7.14 - 0.13x + 0.15y - 2.28z - 0.08x^2 + 1.05xy$
CS-MIO	$\dot{x} = -9.72x + 9.70y$
	$\dot{y} = 29.28x - 1.31y - 1.03xz$
	$\dot{z} = -2.64z + 1.00xy$
(b) Results under Type 2 noise with $\sigma = 2$.	
Ground Truth	$\dot{x} = -10x + 10y$
	$\dot{y} = 28x - y - xz$
	$\dot{z} = -\frac{8}{3}z + xy$
PySINDy	$\dot{x} = -0.21 - 9.87x + 9.89y$
	$\dot{y} = 0.10 + 27.23x - 0.73y$
	$\dot{z} = -1.05 - 2.62z + 1.00xy$
CS-MIO	$\dot{x} = -9.87x + 9.89y$
	$\dot{y} = 27.23x - 0.73y - 0.98xz$
	$\dot{z} = -2.66z + 1.00xy$

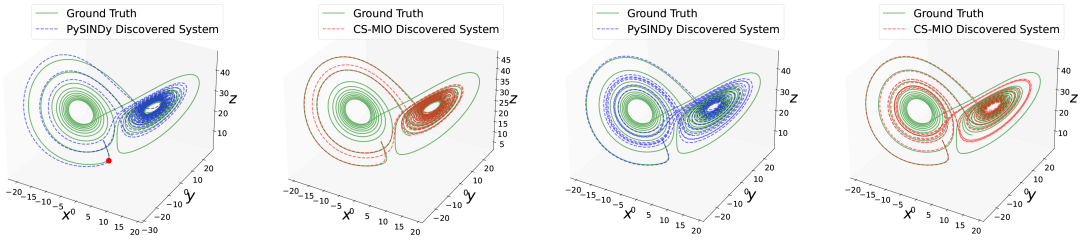


FIGURE 1 | (a) and (b) show the trajectories of PySINDy and CS-MIO discovered equations, respectively, in Table 2a. The trajectory of the PySINDy-identified system deviates from the ground truth right at the beginning (the red dot) of the trajectory. (c) and (d) show the trajectories of PySINDy and CS-MIO identified equations, respectively, in Table 2b.

do not show the figure for SINDy since its identified equations result in an unstable attractor. Additional numerical results on identification of the Lorenz-96 system using CS-MIO method are shown in Figures B1, B2 and Tables B1, B2 in Appendix B.

TABLE 3 | Comparison of the number of exactly recovered equations, that is, the metric $A(\mathbf{\Gamma})$ in Equation (23), for the Lorenz 96 system. Compared with PySINDy, our CS-MIO method can correctly recover all the 96 equations, that is, identifying the correcting $\mathbf{\Gamma}$ in Equation (6), under smaller SNR values.

(a) Results under Type 1 noise.			
		The metric $A(\mathbf{\Gamma})$	
Noise std	**SNR**	**PySINDy**	**CS-MIO**
1	352.147	**96**	**96**
10	3.521	93	**96**
20	0.880	20	**96**
30	0.391	0	**96**
40	0.220	0	**96**
50	0.141	0	**96**
70	0.072	0	88
150	0.016	0	9
230	0.007	0	0
(b) Results under Type 2 noise.			
0.01	132501.470	**96**	**96**
0.05	5300.059	95	**96**
0.1	1325.015	91	**96**
0.2	331.254	45	**96**
0.4	82.813	14	**96**
0.6	36.806	0	**96**
0.8	20.703	0	**96**
1	13.250	0	92
10	0.133	0	0

4.4 | Bifurcations and Parameterized Systems

Parameterized systems exhibit rich dynamic behaviors with various parameter values, which is known as bifurcations. We consider two examples of parameterized systems used in [14]. The first is the 2D Hopf normal form with bifurcation parameter μ,

$$\dot{x} = \mu x - \omega y + Ax(x^2 + y^2) \tag{28}$$

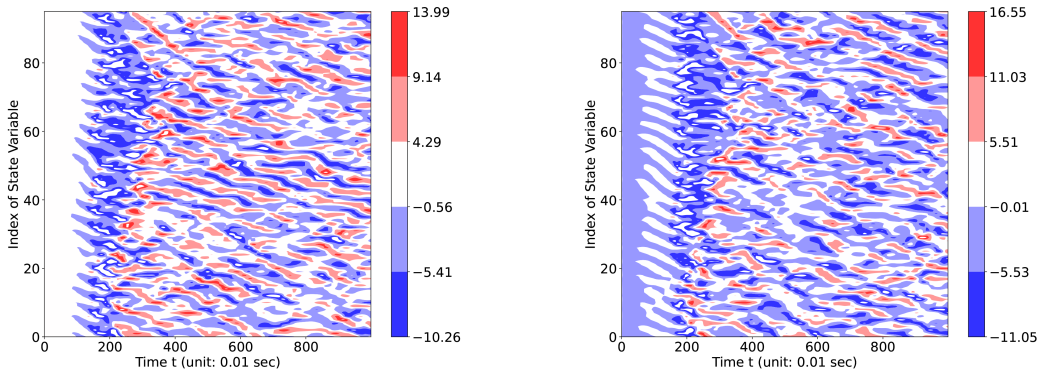
$$\dot{y} = \omega x + \mu y + Ay(x^2 + y^2) \tag{29}$$

To handle the bifurcation behaviors, the μ in the Hopf normal form is treated as additional state variables by adding the dummy differential equation $\dot{\mu} = 0$ to the system [14]. By adopting this setting, we use 14 values of μ to generate 14 datasets, and each dataset is collected using $\Delta t = 0.0025$ in $t \in [0, 75]$. We combine these datasets as a single training dataset to identify the governing equation as functions of state $\mathbf{x}$ and bifurcation parameter μ, that is, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mu)$. Tables 4a and 4b present the number of exact recovered equations of Hopf normal forms under various noise SNRs, Type 1 and Type 2 noises, respectively. Note herein we neglect the counting of the dummy differential equation in both examples. Under Type 1 noise, it is seen that the lowest SRN can be as low as 0.015 for CS-MIO to exactly recover all the equations. Figure 3 shows the trajectory of the CS-MIO-discovered systems under both Type 1 and 2 noise in comparison with the ground truth. Additional numerical results on identification of the Hopf normal form using CS-MIO method are shown in Tables C1 and C2 in Appendix C.

The second example is the 1D logistic map with stochastic forcing and bifurcation parameter r,

$$x_{n+1} = rx_n(1 - x_n) + \eta_n \tag{30}$$

where η_n is the stochastic forcing and r is the bifurcation parameter. Similar manner is imposed on the bifurcation parameter r in the logistic map with the dummy equation $r_{n+1} = r_n$. Besides, 10 values of r are used to collect the data. Within each dataset, we evolve the dynamical system for 1,000 discrete steps. Note that the logistic map is a discrete time dynamical system, so that there is



(a) **(b)**

FIGURE 2 | Hovermöller plot for the difference between the identified system and ground truth of the Lorenz 96 system in $t \in [0, 10]$. The vertical axis is the index j of the state variable. The values of the colors refer to the differences between the ground truth states $x_j(t)$ and the evolved states $\hat{x}_j(t)$ using the identified equations by CS-MIO. The more white-colored areas indicate the trajectory of the identified system agrees better with the ground truth. (**a**) Hovermöller plot under Type 1 noise with $\sigma = 50$. (**b**) Hovermöller plot under Type 2 noise with $\sigma = 0.8$.

only one manner for adding noise (herein η_n) to the state variables $\mathbf{x}$. Table 5 presents the comparison results under various SNRs of noise. CS-MIO exhibits strong capability of recovering governing equations from large noise. In Figure 4, we compare the trajectories of the CS-MIO-identified system with the ground truth. Note we neglect the stochastic forcing η_n when evolving the trajectories in both figures, namely $\eta_n = 0$. The right panel of both Figures 4a and 4b limits the μ in the range of $[3.5, 4]$ for clearer presentation. It can be seen that the trajectory of the CS-MIO-identified system agrees well with the ground truth simulation. Additional numerical results on identification of the Logistic map system using CS-MIO method are shown in Table D1 in Appendix D.

TABLE 4 | Comparison of the number of exactly recovered equations, that is, the metric $A(\mathbf{\Gamma})$ in Equation (23), for Hopf Normal form. Compared with PySINDy, our CS-MIO method can correctly recover all the two equations, that is, identifying the correct $\mathbf{\Gamma}$ in Equation (6), under smaller SNR values.

(a) Results under Type 1 noise.			
		The metric $A(\mathbf{\Gamma})$	
Noise std	**SNR**	**PySINDy**	**CS-MIO**
0.1	13.758	**2**	**2**
0.3	1.529	**2**	**2**
0.5	0.550	1	**2**
0.7	0.281	1	**2**
1	0.138	0	**2**
2	0.034	0	**2**
3	0.015	0	**2**
4	0.009	0	0
(b) Results under Type 2 noise.			
0.001	120306.233	**2**	**2**
0.003	13367.359	**2**	**2**
0.005	4812.249	**2**	**2**
0.007	2455.229	**2**	**2**
0.010	1203.062	0	**2**
0.013	711.871	0	**2**
0.015	534.694	0	**2**
0.017	416.285	0	0

4.5 | PDE for Vortex Shedding Behind a Cylinder

The last example system is the fluid dynamics for vortex shedding behind a cylinder, which are high-dimensional partial differential equations (PDE). As discussed in [14], the high-dimensional PDEs of cylinder dynamics can evolve on a low-dimensional attractor governed by ordinary differential equations after dimension reduction using proper orthogonal decomposition (POD). The mean-field model using three POD modes as coordinate system is given as follows.

$$\dot{x} = \mu x - \omega y + Axz \tag{31}$$

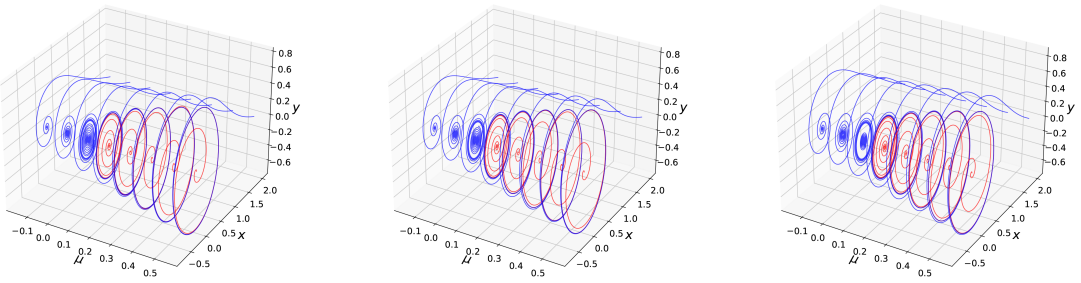
$$\dot{y} = \omega x + \mu y + Ayz \tag{32}$$

$$\dot{z} = -\lambda(z - x^2 - y^2) \tag{33}$$

Herein we use the same dataset used in [14], which is originally generated using direct numerical simulations of the 2D Navier-Stokes equations originally by [40, 41]. We do not employ either Type 1 or Type 2 noise instead of using this single dataset. We identify the governing equations using CS-MIO as presented in Table 6. Note here only second-order polynomials are used for CS-MIO and PySINDy. In this case, neither CS-MIO nor PySINDy is able to exactly identify the governing equations. However, it is

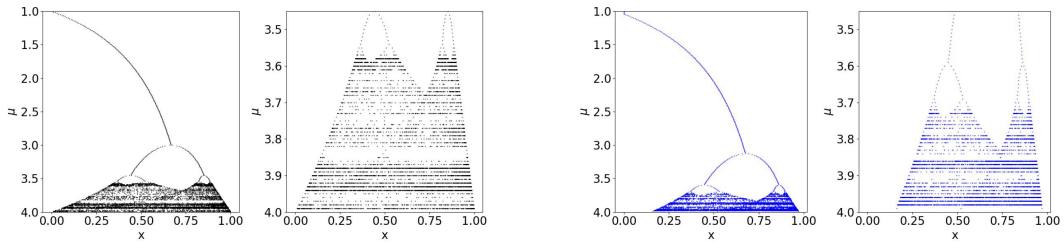
TABLE 5 | Comparison of the number of exactly recovered equations, that is, the metric $A(\mathbf{\Gamma})$ in Equation (23), for the logistic map. Compared with PySINDy, our CS-MIO method can correctly recover the single equation, that is, identify the correcting $\mathbf{\Gamma}$ in Equation (6), under smaller SNR values.

		The metric $A(\mathbf{\Gamma})$	
Noise std	**SNR**	**PySINDy**	**CS-MIO**
0.1	8.985	**1**	**1**
0.2	3.619	**1**	**1**
0.3	2.146	0	**1**
0.4	1.455	0	**1**
0.5	1.098	0	**1**
0.6	0.874	0	**1**
0.7	0.738	0	0



(a) Trajectory of the ground truth. **(b)** Type 1 noise at σ=3. **(c)** Type 2 noise at σ=0.015.

FIGURE 3 | Trajectories of CS-MIO discovered systems for Hopf normal form. (a) Trajectory of the ground truth full simulation. (b) Trajectory of the CS-MIO-identified system under Type 1 noise at $\sigma = 3$. (c) Trajectory of the CS-MIO-identified system under Type 2 noise at $\sigma = 0.015$.

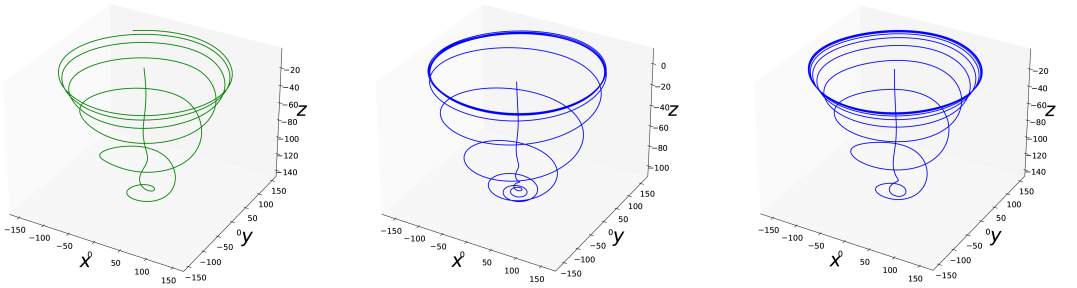


(a) **(b)**

FIGURE 4 | Trajectories of CS-MIO identified models for the logistic map system under noise magnitude 0.2 in (b) and the comparison to the ground truth in (a) for ten values of r. (**a**) Trajectory of the ground truth. (**b**) Trajectory of the CS-MIO identified system.

TABLE 6 | Identified coefficients by CS-MIO and PySINDy on flow wake behind a cylinder. Quadratic terms are identified. The bold coefficients refer to those in the ground truth mean field model. CS-MIO can identify all the ground truth terms using less nonzeros in $\mathbf{\Gamma}$ in comparison with PySINDy.

	Equation 1		Equation 2		Equation 3	
Term	**PySINDy**	**CS-MIO**	**PySINDy**	**CS-MIO**	**PySINDy**	**CS-MIO**
Bias	−0.1225	0	−0.0569	0	−21.9002	−20.8466
x	**−0.0092**	**−0.0092**	**1.0347**	**1.0346**	−0.0009	0
y	**−1.0224**	**−1.0225**	**0.0047**	**0.0046**	0	0
z	−0.0009	0	−0.0004	0	**−0.3117**	**−0.2968**
x^2	0	0	0	0	**0.0011**	**0.0011**
xy	0	0	0	0	0.0002	0
xz	**0.0002**	**0.0002**	0.0022	0.0022	0	0
y^2	0	0	0	0	**0.0009**	**0.0009**
yz	−0.0019	−0.0019	**−0.0018**	**−0.0018**	0	0
z^2	0	0	0	0	−0.0011	−0.0010



(a) **(b)** **(c)**

FIGURE 5 | Trajectories of the full simulation and the CS-MIO identified system for the cylinder dynamics. (**a**) Ground truth trajectory. (**b**) CS-MIO discovered a 2nd order system. (**c**) CS-MIO discovered a 3rd order system.

seen that CS-MIO uses 4, 4, and 5 terms, respectively, for each equation to include all those in the ground truth, while PySINDy uses 6, 6, and 7 terms and includes many false terms. In a word, CS-MIO uses much fewer nonlinear terms to include all those in the ground truth than PySINDy.

If λ is large, then the dynamics on the z coordinate is fast, resulting in the quick transient dynamics from the mean flow to the parabolic slow manifold, that is, $z = x^2 + y^2$ given by the amplitude of the vortex shedding. This dynamics are seen in Figure 5a as the sharp decreasing along the z coordinate and then correcting to the parabolic slow manifold. If substituting $z = x^2 + y^2$ into Equations 31 and 32, we obtain a Hopf normal form system on the slow manifold, which includes cubic nonlinearities. We thus set polynomial order to be three in both CS-MIO and PySINDy for recovery. The results are shown in Table 7. In this case, both CS-MIO and PySINDy fail to include all the ground truth terms, although they all involve many redundant terms. This is reasonable since higher-order nonlinearities can express the dynamics of lower-order nonlinearities. From Figures 5b and 5c, it can be seen that the CS-MIO identified

TABLE 7 | Identified coefficients of CS-MIO and PySINDy on flow wake behind a cylinder. Cubic terms are identified for the mean field model. The bold terms indicate the ground truth. CS-MIO can identify equal or more ground truth terms using less nonzeros in comparison with PySINDy.

	Equation 1		Equation 2		Equation 3	
Term	**PySINDy**	**CS-MIO**	**PySINDy**	**CS-MIO**	**PySINDy**	**CS-MIO**
Bias	0	0	0	0	0	−9.66082
x	**0**	**0**	**0**	**1.02896**	0	0
y	**−1.04203**	**−0.21545**	**0.00621**	**0.24547**	0.00025	0
z	0.00002	0	−0.00004	0	**0.47502**	**0.19082**
x^2	0	0	0	0	**0.00006**	**0.00047**
xy	0	0	0	0	−0.00019	0
xz	0.00138	0.00275	−0.00744	0.00222	0	0
y^2	0	0	0	0	**−0.00006**	**0.00038**
yz	−0.00367	0.00396	−0.00366	0	0	0
z^2	0	0	0	0	0.00532	0.00296
x^3	**0**	**0**	0.00005	0	0	0
x^2y	0	−0.00004	**0**	**−0.00001**	0	0
x^2z	0	0	0	0	−0.00003	−0.00002
xy^2	**0**	**0**	0.00005	0	0	0
xyz	0	0	0	0	−0.00002	−0.00002
xz^2	0.00001	0.00002	−0.00002	0	0	0
y^3	0	−0.00004	**0**	**−0.00001**	0	0
y^2z	0	0	0	0	−0.00002	−0.00002
yz^2	−0.00002	0	−0.00002	0	0	0
z^3	0	0	0	0	0.00001	0.00001

system agrees almost perfectly with the full simulation using the original dataset.

5 | Conclusion

We have developed a compressive-sensing-assisted mixed-integer optimization method for recovery of dynamical systems from highly noisy data. As there remain many unknown governing equations across various disciplines in science and engineering, our developed method is critical for uncovering the unknown equations from the noisy data that is practically observed in such systems. The proposed method is developed on the important foundation that is the identification of terms in the governing equations is essentially a discrete optimization problem. Because of this, our method is able to separately control the exact sparsity of the governing equations and estimate the associated coefficients. This differs significantly from existing research where sparsity is incurred by a penalty on the coefficients. We also combine the mixed-integer optimization with compressive sensing and other regularization techniques for enhancing the capability for dealing with highly noisy and high-dimensional problems. Case studies using the classical dynamical system examples demonstrate the powerful capability of the proposed method to uncover the governing equations under large noise, significantly outperforming the state-of-the-art method. This work opens several doors for future directions. First, advanced algorithms could be developed to enhance the efficiency of the method for large-scale instances of the studied problem. In addition, the domain knowledge for specifying the number of active terms can be used to discover new governing equations in specific fields. The construction of candidate terms using rich symbolic expression is further an exciting potential direction.

Acknowledgements

This material is based upon work supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, and by the Laboratory Directed Research and Development program at the Oak Ridge National Laboratory, which is operated by UT-Battelle LLC, for the U.S. Department of Energy under Contract DE-AC05-00OR22725. This manuscript is partially supported by the Science Alliance GATE (Graduate Advancement, Training and Education) Award of the University of Tennessee Knoxville (ERJK388).

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

1. J. Bongard and H. Lipson, "Automated Reverse Engineering of Nonlinear Dynamical Systems," *Proceedings of the National Academy of Sciences* 104, no. 24 (2007): 9943–9948.

2. M. Schmidt and H. Lipson, "Distilling Free-Form Natural Laws From Experimental Data," *Science* 324, no. 5923 (2009): 81–85.

3. M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Machine Learning of Linear Differential Equations Using Gaussian Processes," *Journal of Computational Physics* 348 (2017): 683–693.

4. Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, "Learning Data-Driven Discretizations for Partial Differential Equations," *Proceedings of the National Academy of Sciences* 116, no. 31 (2019): 15344–15349, https://doi.org/10.1073/pnas.1814058116.

5. S. H. Rudy, J. Nathan Kutz, and S. L. Brunton, "Deep Learning of Dynamics and Signal-Noise Decomposition With Time-Stepping Constraints," *Journal of Computational Physics* 396 (2019): 483–506, https://doi.org/10.1016/j.jcp.2019.06.056.

6. T. Qin, Z. Chen, J. D. Jakeman, and D. Xiu, "Data-Driven Learning of Nonautonomous Systems," *SIAM Journal on Scientific Computing* 43, no. 3 (2021): A1607–A1624.

7. Z. Chen and D. Xiu, "On Generalized Residual Network for Deep Learning of Unknown Dynamical Systems," *Journal of Computational Physics* 438 (2021): 110362, https://doi.org/10.1016/j.jcp.2021.110362.

8. M. Yang, P. Wang, D. Del-Castillo-Negrete, Y. Cao, and G. Zhang, "A Pseudo-Reversible Normalizing Flow for Stochastic Dynamical Systems With Various Initial Conditions," *arXiv-preprint, arXiv:2306.05580* (2023).

9. F. Bao, Z. Zhang, and G. Zhang, "A Score-Based Filter for Nonlinear Data Assimilation," *arXiv-preprint, arXiv:2306.09282* (2024).

10. F. Bao, Z. Zhang, and G. Zhang, "An Ensemble Score Filter for Tracking High-Dimensional Nonlinear Dynamical System," *ORNL Report* 2023; ORNL/TM-2023/3086: 1–17.

11. Y. Liu, M. Yang, Z. Zhang, F. Bao, Y. Cao, and G. Zhang, "Diffusion-Model-Assisted Supervised Learning of Generative Models for Density Estimation," *Journal of Machine Learning for Modeling and Computing* 5, no. 1 (2024): 25–38.

12. F. Bao, Z. Zhang, and G. Zhang, "A Unified Filter Method for Jointly Estimating State and Parameters of Stochastic Dynamical Systems via the Ensemble Score Filter," *arXiv-preprint, arXiv:2312.10503* (2023).

13. S. Zhang and G. Lin, "Robust Data-Driven Discovery of Governing Physical Laws With Error Bars," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 474, no. 2217 (2018): 20180305, https://doi.org/10.1098/rspa.2018.0305.

14. S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering Governing Equations From Data by Sparse Identification of Nonlinear Dynamical Systems," *Proceedings of the National Academy of Sciences* 113, no. 15 (2016): 3932–3937.

15. S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Data-Driven Discovery of Partial Differential Equations," *Science Advances* 3, no. 4 (2017): e1602614.

16. S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Sparse Identification of Nonlinear Dynamics With Control (SINDYc)," *IFAC-PapersOnLine* 49, no. 18 (2016): 710–715.

17. J. C. Loiseau, B. R. Noack, and S. L. Brunton, "Sparse Reduced-Order Modelling: Sensor-Based Dynamics to Full-State Estimation," *Journal of Fluid Mechanics* 844 (2018): 459–490.

18. K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, "Data-Driven Discovery of Coordinates and Governing Equations," *Proceedings of the National Academy of Sciences* 116, no. 45 (2019): 22445–22451.

19. N. M. Mangan, T. Askham, S. L. Brunton, J. N. Kutz, and J. L. Proctor, "Model Selection for Hybrid Dynamical Systems via Sparse Regression," *Proceedings of the Royal Society A* 475, no. 2223 (2019): 20180534.

20. D. B. M. Silva, K. Champion, M. Quade, J. C. Loiseau, J. N. Kutz, and S. L. Brunton, "Pysindy: A Python Package for the Sparse Identification of Nonlinear Dynamics From Data," *arXiv preprint arXiv:2004.08424* (2020).

21. H. Schaeffer, "Learning Partial Differential Equations via Data Discovery and Sparse Optimization," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473, no. 2197 (2017): 20160446, https://doi.org/10.1098/rspa.2016.0446.

22. G. Tran and R. Ward, "Exact Recovery of Chaotic Systems From Highly Corrupted Data," *Multiscale Modeling and Simulation* 15, no. 3 (2017): 1108–1129, https://doi.org/10.1137/16M1086637.

23. D. A. Messenger and D. M. Bortz, "Weak SINDy: Galerkin-Based Data-Driven Model Selection," *Multiscale Modeling and Simulation* 19, no. 3 (2021): 1474–1497, https://doi.org/10.1137/20M1343166.

24. H. Schaeffer and S. G. McCalla, "Sparse Model Selection via Integral Terms," *Physical Review E* 96 (2017): 023302, https://doi.org/10.1103/PhysRevE.96.023302.

25. Y. Pantazis and I. Tsamardinos, "A Unified Approach for Sparse Dynamical System Inference From Temporal Measurements," *Bioinformatics* 35 (2019): 3387–3396.

26. W. X. Wang, R. Yang, Y. C. Lai, V. Kovanis, and C. Grebogi, "Predicting Catastrophes in Nonlinear Dynamical Systems by Compressive Sensing," *Physical Review Letters* 106, no. 15 (2011): 154101.

27. B. K. Natarajan, "Sparse Approximate Solutions to Linear Systems," *SIAM Journal on Computing* 24, no. 2 (1995): 227–234.

28. A. Miller, *Subset Selection in Regression* (Boca Raton, Florida: CRC Press, 2002).

29. D. Bertsimas, A. King, and R. Mazumder, "Best Subset Selection via a Modern Optimization Lens," *Annals of Statistics* 44, no. 2 (2016): 813–852.

30. D. Bertsimas, J. Pauphilet, and B. Van Parys, "Rejoinder: Sparse Regression: Scalable Algorithms and Empirical Performance," *Statistical Science* 35, no. 4 (2020): 623–624.

31. R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society: Series B: Methodological* 58, no. 1 (1996): 267–288.

32. H. Zou and T. Hastie, "Regularization and Variable Selection via the Elastic Net," *Journal of the Royal Statistical Society, Series B: Statistical Methodology* 67, no. 2 (2005): 301–320.

33. J. Fan and R. Li, "Variable Selection via Nonconcave Penalized Likelihood and Its Oracle Properties," *Journal of the American Statistical Association* 96, no. 456 (2001): 1348–1360.

34. R. Mazumder, J. H. Friedman, and T. Hastie, "Sparsenet: Coordinate Descent With Nonconvex Penalties," *Journal of the American Statistical Association* 106, no. 495 (2011): 1125–1138.

35. N. R. Draper and H. Smith, *Applied Regression Analysis. 326* (Hoboken, New Jersey: John Wiley & Sons, 1998).

36. B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least Angle Regression," *Annals of Statistics* 32, no. 2 (2004): 407–499.

37. D. B. Silva, K. Champion, M. Quade, J. C. Loiseau, J. Kutz, and S. Brunton, "Pysindy: A Python Package for the Sparse Identification of Nonlinear Dynamical Systems From Data," *Journal of Open Source Software* 5, no. 49 (2020): 2104, https://doi.org/10.21105/joss.02104.

38. A. A. Kaptanoglu, D. B. M. Silva, U. Fasel, et al., "Pysindy: A Comprehensive Python Package for Robust Sparse System Identification," *arXiv preprint arXiv:2111.08481* (2021).

39. R. Chartrand, "Numerical Differentiation of Noisy, Nonsmooth Data," *International Scholarly Research Notices* 2011, (2011): 164564.

40. K. Taira and T. Colonius, "The Immersed Boundary Method: A Projection Approach," *Journal of Computational Physics* 2025, no. 2 (2007): 2118–2137.

41. 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, no. 25–28 (2008): 2131–2146.

Appendix A

Additional Results for the Chaotic Lorenz 3 System

We provide additional results for the Chaotic Lorenz 3 system (see Figures A1, A2 and Tables A1, A2).

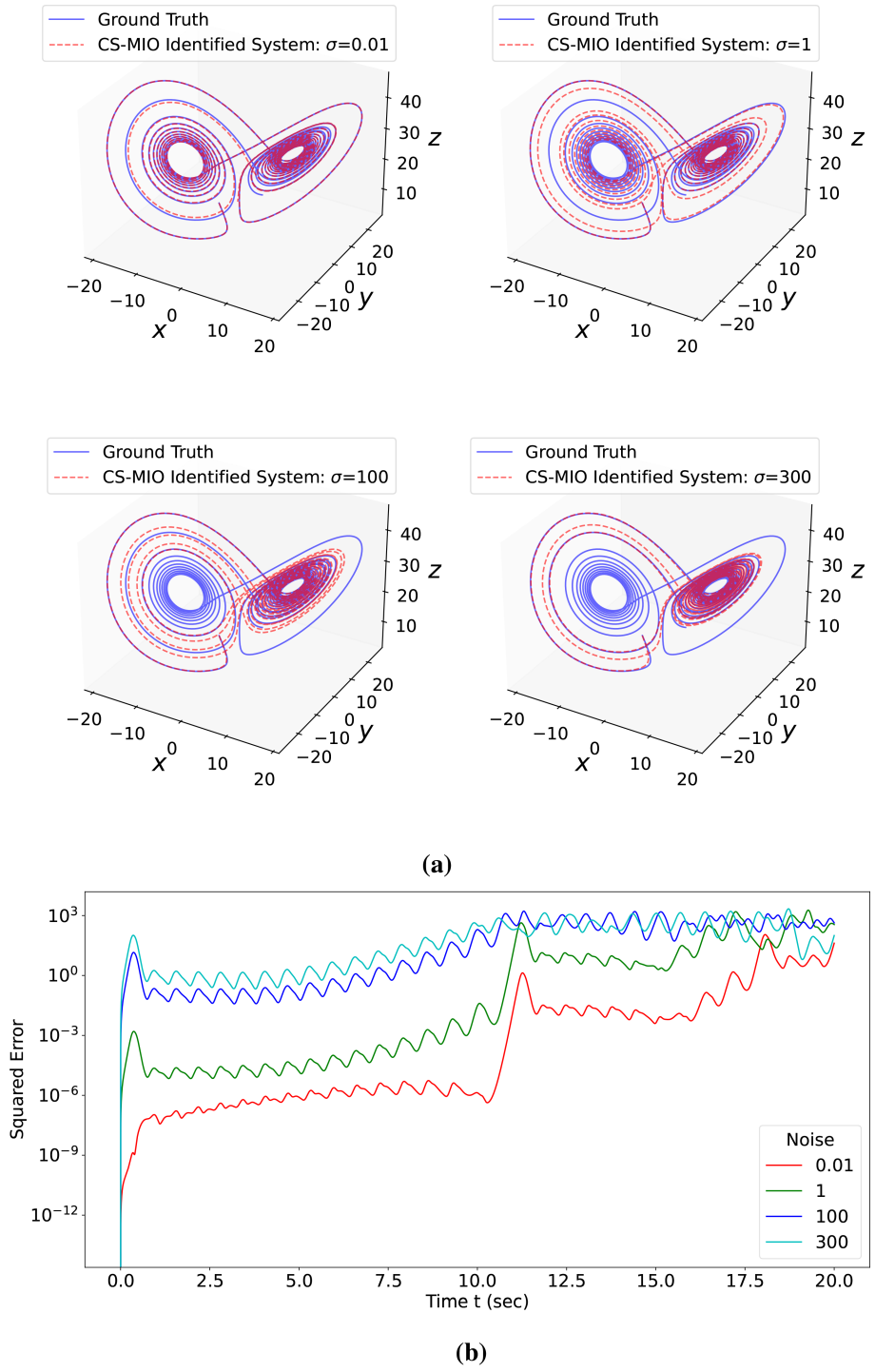
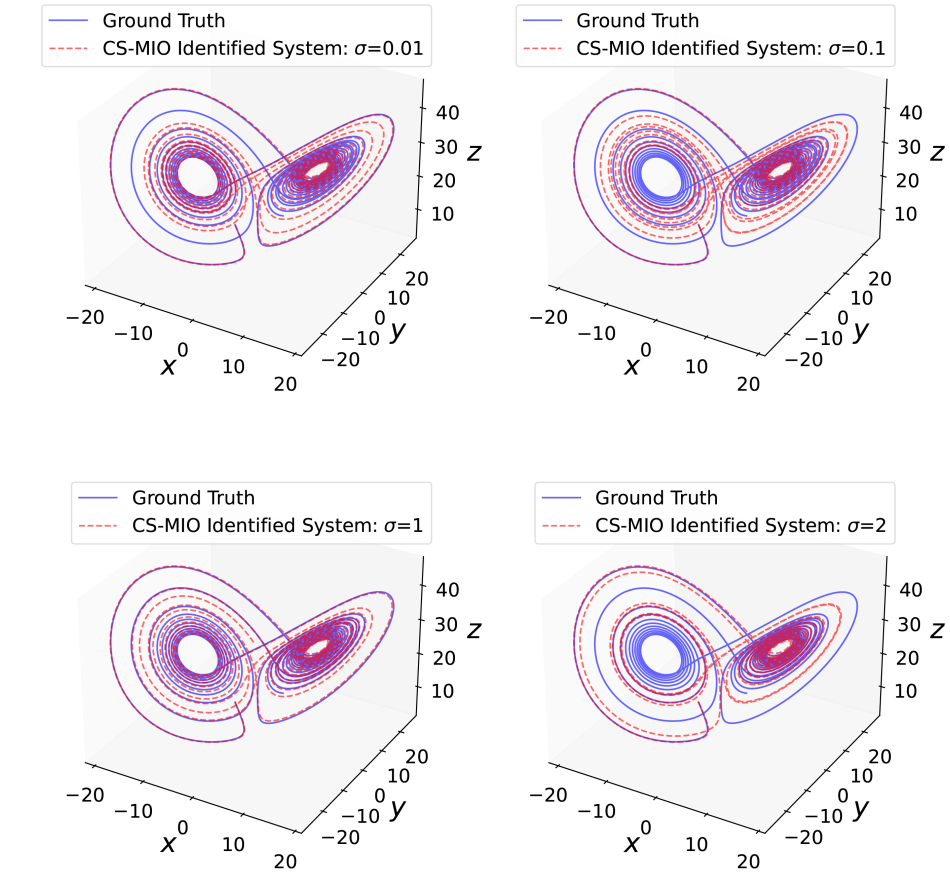
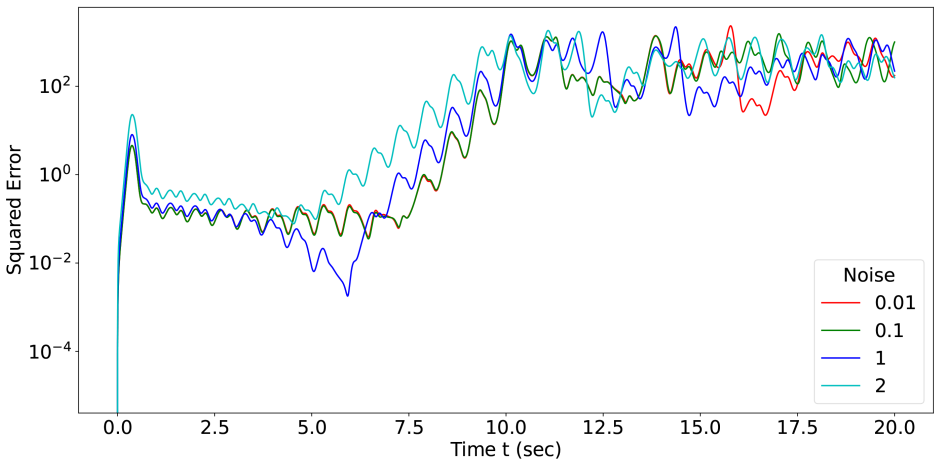


FIGURE A1 | Simulation results of the CS-MIO identified Lorenz 3 system comparing to the ground truth from $t = 0$ to $t = 20$ under Type 1 noise with four noise magnitudes σ: 0.01, 1, 100, and 300. The exact recovery fails when σ is larger than 300. (a) Trajectories of the CS-MIO identified system (red dashed) and ground truth (blue solid). (b) ℓ_2 error vs time of the trajectories of the recovered Lorenz 3 system ($\hat{\mathbf{x}}(t)$) comparing to the ground truth ($\mathbf{x}(t)$), that is, $||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||_2^2$ as a function of t from $t = 0$ to $t = 20$. (**a**) Trajectories of the CS-MIO identified system and the ground truth. (**b**) ℓ_2 error between the CS-MIO identified system and the ground truth.



(a)



(b)

FIGURE A2 | Simulation results of the CS-MIO identified Lorenz 3 system comparing to the ground truth from $t = 0$ to $t = 20$ under Type 2 noise with four noise magnitudes σ: 0.01, 0.1, 1 and 2. The exact recovery fails when σ ls larger than 2. (a) Trajectories of the CS-MIO identified system (red dashed) and ground truth (blue solid). (b) ℓ_2 error vs time of the trajectories of the recovered Lorenz 3 system ($\hat{\mathbf{x}}(t)$) comparing to the ground truth ($\mathbf{x}(t)$), that is, $||\hat{\mathbf{x}}(t) - \mathbf{x}(t)||_2^2$ as a function of t from $t = 0$ to $t = 20$. (**a**) Trajectories of the CS-MIO identified system and the ground truth. (**b**) ℓ_2 error between the CS-MIO identified system and the ground truth.

TABLE A1 | Identified coefficients of Lorenz 3 system using CS-MIO under Type 1 noise.

Noise: σ	SNR	x x z	y y xy	xz
0.01	41914317.129	−10.0000	10.0000	−1.0000
		28.0000	−1.0000	
		−2.6667	1.0000	
0.1	419143.171	−9.9999	9.9999	−1.0000
		28.0004	-1.0001	
		−2.6667	1.0000	
1	4191.621	−9.9991	9.9990	−1.0001
		28.0043	−1.0010	
		−2.6666	1.0000	
10	41.916	−9.9908	9.9900	−1.0009
		28.0426	−1.0103	
		−2.6658	1.0000	
50	1.677	−9.9540	9.9499	−1.0044
		28.2131	−1.0516	
		−2.6624	1.0001	
100	0.419	−9.9081	9.8999	−1.0089
		28.4263	−1.1032	
		−2.6581	1.0001	
150	0.186	−9.8621	9.8498	−1.0133
		28.6394	−1.1548	
		−2.6538	1.0002	
200	0.105	−9.8161	9.7998	−1.0177
		28.8525	−1.2065	
		−2.6495	1.0002	
250	0.067	−9.7701	9.7497	−1.0222
		29.0657	−1.2581	
		−2.6452	1.0003	
300	0.047	−9.7242	9.6997	−1.0266
		29.2788	−1.3097	
		−2.6409	1.0003	

TABLE A2 | Identified coefficients of Lorenz 3 system using CS-MIO under Type 2 noise.

Noise: σ	SNR	x	y	xz
		x	y	
		z	xy	
0.01	729427.159	−9.9851	10.0000	
		27.6974	−0.8682	−0.9939
		−2.6602	0.9997	
0.05	29178.677	−9.9852	9.9999	
		27.6968	−0.8682	−0.9939
		−2.6602	0.9997	
0.1	7295.616	−9.9851	9.9997	
		27.6954	−0.8680	−0.9938
		−2.6603	0.9997	
0.5	292.848	−9.9791	9.9934	
		27.6635	−0.8596	−0.9929
		−2.6608	0.9999	
1	73.982	−9.9573	9.9730	
		27.5722	−0.8335	−0.9906
		−2.6605	0.9997	
2	19.255	−9.8670	9.8916	
		27.2258	−0.7319	−0.9823
		−2.6571	0.9984	

Appendix B

Additional Results for the Chaotic Lorenz 96 System

See Figures B1, B2 and Tables B1, B2.

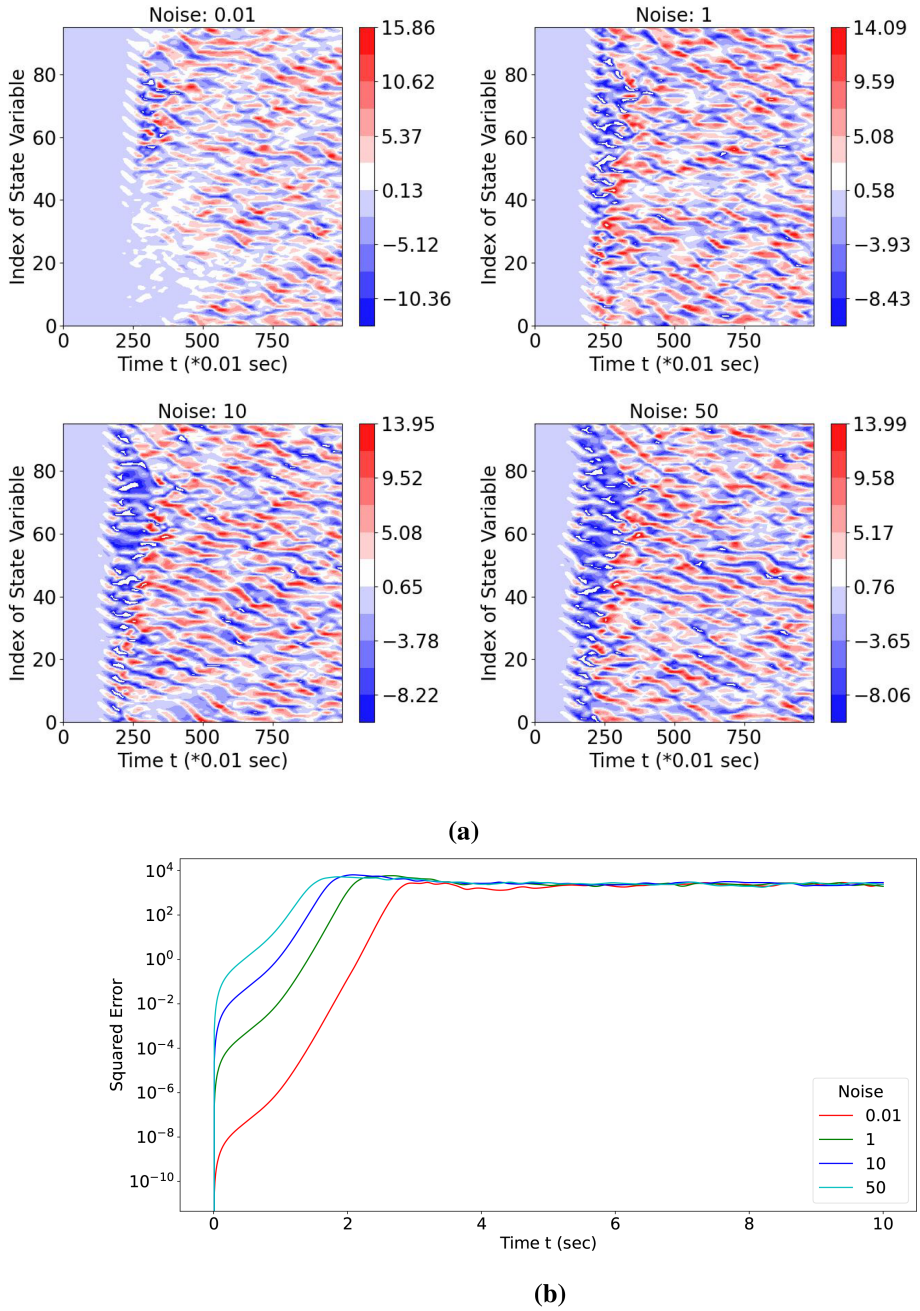
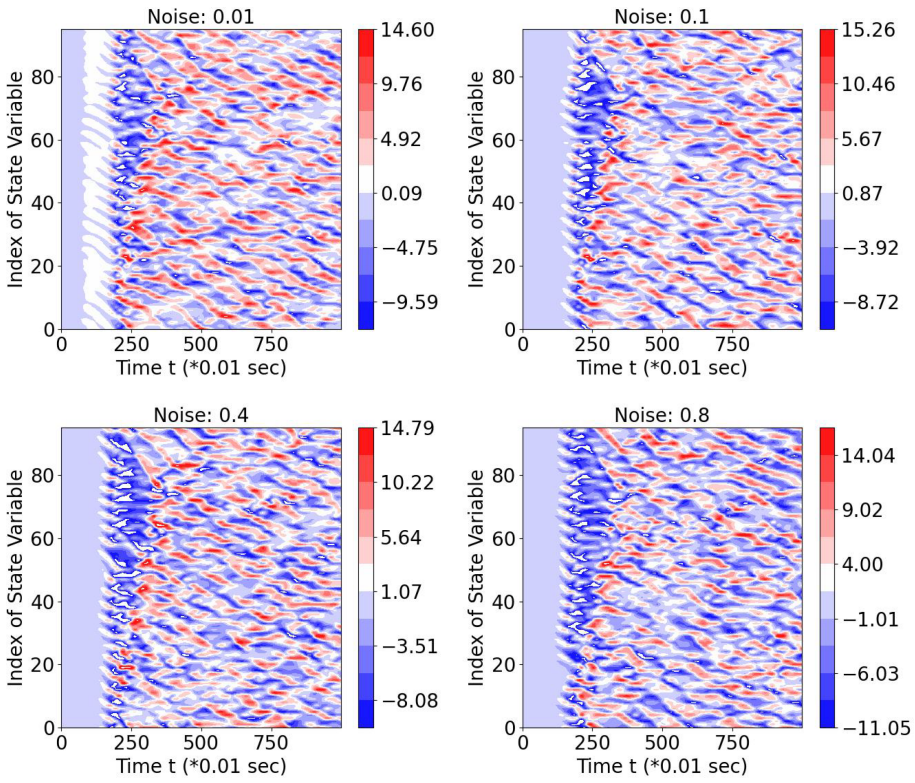
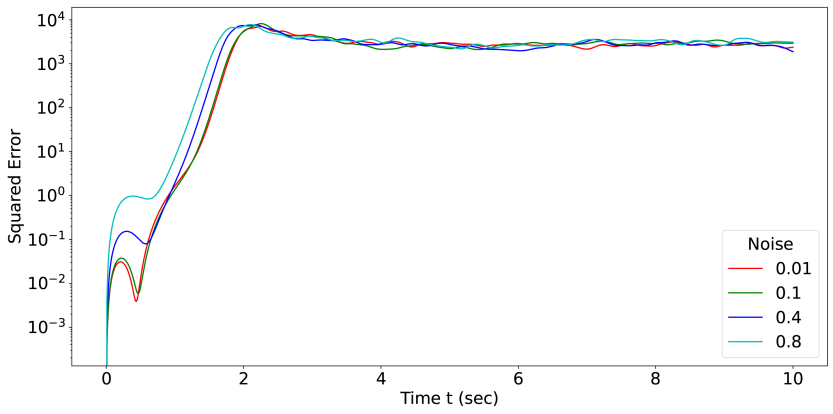


FIGURE B1 | Simulation results of CS-MIO identified Lorenz 96 system using 60k data and under Type 1 noise with four noise magnitudes, namely 0.01, 1, 10 and 50. (a) Hovermöller plot for difference between the identified system and ground truth of Lorenz 96 system in $t \in [0, 10]$. The vertical axis is the index j of the state variable. The values of the colors refer to the differences between the ground truth states $x_j(t)$ and the evolved states $\hat{x}_j(t)$ using the identified equations by CS-MIO, that is, $\Delta x_j(t) = x_j(t) - \hat{x}_j(t)$ for $j = 1, \cdots, 96$. (b) l_2 error versus time of the trajectories of the CS-MIO recovered Lorenz 96 system from $t = 0$ to $t = 10$. The exact recovery fails when σ is larger than 50. (**a**) Hovermöller plot for difference between the identified system and ground truth. (**b**) ℓ_2 error between the identified system and ground truth.



(a)



(b)

FIGURE B2 | Simulation results of CS-MIO identified Lorenz 96 system using 60 k data and under Type 2 noise with four noise magnitudes, namely 0.01, 0.1, 0.4, and 0.8. (a) Hovermöller plot for difference between the identified system and ground truth of Lorenz 96 system in $t \in [0, 10]$. The vertical axis is the index j of the state variable. The values of the colors refer to the differences between the ground truth states $x_j(t)$ and the evolved states $\hat{x}_j(t)$ using the identified equations by CS-MIO, that is, $\Delta x_j(t) = x_j(t) - \hat{x}_j(t)$ for $j = 1, \cdots, 96$. (b) l_2 error versus time of the trajectories of the recovered Lorenz 96 system from $t = 0$ to $t = 10$. The exact recovery fails when σ is larger than 0.8. (**a**) Hovermöller plot for difference between the identified system and ground truth. (**b**) ℓ_2 error between the identified system and ground truth.

TABLE B1 | Identified coefficients of Lorenz 96 system using CS-MIO under Type 1 noise with magnitude 50.

Equation j	F	$x_{j+1}x_{j-1}$	$x_{j-2}x_{j-1}$	x_j	Equation j	F	$x_{j+1}x_{j-1}$	$x_{j-2}x_{j-1}$	x_j
1	7.6636	0.9954	−0.9919	−0.9053	49	8.1873	1.0016	−1.0014	−0.9289
2	8.2107	1.0137	−0.9957	−1.0136	50	7.8786	1.0036	−1.0096	−1.01
3	8.0884	1.0031	−0.9776	−1.0558	51	7.7901	1.0235	−0.9655	−1.0325
4	7.7594	0.9958	−0.9925	−1.0204	52	7.4264	0.9828	−0.9877	−0.8955
5	7.6274	0.9997	−1.002	−0.9596	53	7.728	0.9898	−0.9848	−0.9488
6	7.6735	1.0009	−0.979	−0.9568	54	8.084	0.9995	−0.9977	−0.9669
7	8.1661	1.0126	−1.0122	−1.0837	55	8.8206	0.9805	−1.0038	−1.0711
8	7.7873	1.0004	−0.993	−0.9782	56	8.5845	0.9934	−1.012	−1.0968
9	8.1277	0.9978	−1.004	−1.0211	57	8.1714	0.9805	−0.9903	−0.971
10	8.2015	1.031	−1.0276	−0.9769	58	8.5471	1.0031	−1.0284	−1.0648
11	7.7901	0.9904	−1.0143	−0.8445	59	8.2529	1.0088	−0.9877	−0.9947
12	8.0504	0.9844	−0.9902	−1.0707	60	8.2181	1.0111	−1.0064	−1.0332
13	7.9518	0.9889	−1.0144	−1.0927	61	8.3878	1.031	−1.0099	−1.0172
14	7.8521	1.0055	−0.9897	−0.9773	62	8.4394	0.9898	−1.012	−1.0398
15	8.4207	0.9905	−0.9984	−1.0906	63	8.4044	1.0326	−0.9997	−1.0862
16	8.2164	1.0162	−1.0171	−0.9698	64	8.1365	1.0058	−1.0072	−1.0033
17	8.3087	0.9924	−0.9893	−1.0197	65	7.9563	0.9895	−0.9893	−1.0268
18	8.4646	1.0119	−1.0123	−1.0714	66	8.3239	0.9874	−0.9911	−1.0302
19	8.2501	0.9759	−0.9886	−1.043	67	8.1219	1.0129	−1.0013	−1.0345
20	7.8796	0.9919	−0.9873	−0.9722	68	7.825	1.0037	−0.9819	−0.9692
21	7.8357	0.9929	−1.0093	−0.9649	69	7.8334	1.0085	−1.0088	−0.9828
22	8.5176	0.9749	−1.0268	−1.1041	70	8.0292	1.0082	−0.9822	−1.1236
23	8.4018	1.0369	−1.0221	−1.0326	71	8.1128	0.9926	−1.0058	−1.0314
24	7.9646	0.9937	−0.983	−1.0083	72	8.217	1.0018	−0.9909	−1.088
25	7.7756	0.9871	−0.996	−0.9909	73	8.1572	1.002	−0.9912	−0.9913
26	7.6644	0.9907	−1.0124	−1.0218	74	7.9162	1.0058	−0.9838	−0.98
27	8.1968	1.003	−1.01	−1.0355	75	8.4242	1.0166	−1.0046	−1.1802
28	8.2314	0.9985	−0.9925	−1.0586	76	8.2365	1.0021	−1.0086	−1.0709
29	8.2346	1.0033	−1.0061	−1.0567	77	8.4704	1.0057	−1.0092	−1.0961
30	7.6806	1.0109	−0.9958	−0.8968	78	8.2634	1.0109	−1.0043	−1.03
31	8.3632	1.016	−1.0046	−1.0581	79	8.0802	0.9659	−0.9874	−1.0338
32	8.1515	1.0083	−0.9858	−0.9228	80	8.2453	1.0183	−0.9971	−1.0583
33	8.0453	1.0109	−1.0234	−0.9918	81	8.5512	0.9834	−1.0118	−1.079
34	7.8578	1.0029	−0.9876	−0.9044	82	7.8437	0.9997	−0.9978	−0.9929
35	7.8325	1.007	−1.0029	−0.9443	83	8.134	0.9818	−0.9918	−0.9468
36	8.0977	0.9873	−0.9925	−0.9986	84	8.3957	1.0119	−1.0095	−1.0589
37	8.6882	0.9952	−1.0037	−1.0654	85	8.1792	1.0053	−0.9906	−1.0231
38	7.6767	0.9983	−0.9685	−1.0238	86	8.1009	0.9933	−0.998	−0.9385
39	8.0614	0.9718	−0.9919	−0.9915	87	8.0357	0.9629	−1.0263	−0.8889
40	7.8614	1.0046	−1.0125	−1.0181	88	7.9619	0.9825	−0.9979	−0.9141
41	7.2833	0.9881	−0.9548	−0.9379	89	8.225	0.9839	−0.9932	−1.0779
42	8.0186	0.9879	−0.9965	−1.0273	90	7.8981	0.9894	−0.9881	−0.8814
43	8.0823	0.9994	−1.0185	−1.0349	91	7.7766	0.9819	−1.0157	−0.9386
44	7.9811	0.9758	−1.0048	−1.0275	92	7.9365	1.0125	−1.0174	−0.9876
45	7.9975	0.9956	−0.997	−0.9966	93	8.3197	1.0293	−1.0184	−1.0195
46	7.7917	0.9986	−0.9901	−0.9353	94	7.6349	0.9901	−0.9642	−0.9583
47	7.7668	0.9971	−0.9891	−1.0075	95	7.8329	1.0115	−1.0067	−0.9771
48	7.5492	1.0106	−0.9717	−1.0031	96	7.9165	0.9803	−0.9942	−1.008

TABLE B2 | Identified coefficients of Lorenz 96 system using CS-MIO under Type 2 noise with magnitude 0.8.

Equation j	F	$x_{j+1}x_{j-1}$	$x_{j-2}x_{j-1}$	x_j	Equation j	F	$x_{j+1}x_{j-1}$	$x_{j-2}x_{j-1}$	x_j
1	7.0568	0.9459	-0.9658	-0.6757	49	7.1351	0.9523	−0.9683	−0.6815
2	7.0112	0.9483	−0.9639	−0.6958	50	7.0338	0.9526	−0.9637	−0.6808
3	7.0591	0.9475	−0.9581	−0.6963	51	7.0436	0.9391	−0.9574	−0.6857
4	7.0493	0.9571	−0.9597	−0.6776	52	7.0394	0.9566	−0.9659	−0.6886
5	7.1518	0.9481	−0.9672	−0.706	53	7.1219	0.9587	−0.9692	−0.6887
6	7.0869	0.9523	−0.9635	−0.6799	54	7.1177	0.9464	−0.9631	−0.7365
7	7.1948	0.943	−0.9682	−0.6986	55	7.0162	0.9496	−0.9678	−0.6666
8	7.1661	0.9491	−0.9674	−0.7077	56	7.2848	0.9554	−0.9675	−0.7262
9	6.9831	0.9408	−0.9571	−0.6745	57	7.0027	0.9609	−0.9591	−0.6802
10	7.0809	0.9559	−0.9623	−0.7037	58	7.0277	0.9523	−0.964	−0.6903
11	7.0943	0.9516	−0.9639	−0.6838	59	6.8512	0.9535	−0.9636	−0.6585
12	7.2745	0.9574	−0.9713	−0.7263	60	7.2209	0.9483	−0.9651	−0.6831
13	7.0001	0.963	−0.9728	−0.6775	61	7.0277	0.9493	−0.9647	−0.674
14	7.1846	0.9497	−0.9675	−0.708	62	7.055	0.9545	−0.9701	−0.6907
15	7.1597	0.9475	−0.9624	−0.6962	63	7.0577	0.9549	−0.9617	−0.6875
16	7.0932	0.9526	−0.967	−0.6986	64	7.129	0.9515	−0.9595	−0.6763
17	7.052	0.9541	−0.9552	−0.7014	65	7.0478	0.9457	−0.9674	−0.6722
18	6.8848	0.9549	−0.9643	−0.6501	66	7.1539	0.9407	−0.9648	−0.7014
19	7.3256	0.9529	−0.971	−0.705	67	6.9451	0.9446	−0.9607	−0.6592
20	7.1578	0.9634	−0.9677	−0.6955	68	7.0965	0.9588	−0.9702	−0.6837
21	6.9812	0.9586	−0.9734	−0.689	69	7.034	0.9422	−0.9566	−0.6943
22	6.9374	0.9463	−0.9665	−0.656	70	7.054	0.9546	−0.9654	−0.7106
23	7.2987	0.9527	−0.9702	−0.7081	71	6.9663	0.9495	−0.9619	−0.6307
24	7.0471	0.9486	−0.961	−0.6813	72	7.2584	0.9489	−0.9604	−0.7156
25	7.0503	0.9581	−0.9643	−0.7093	73	7.0076	0.9572	−0.968	−0.6768
26	6.9403	0.9514	−0.9609	−0.6487	74	7.1294	0.945	−0.9585	−0.7171
27	7.2672	0.9534	−0.9608	−0.706	75	7.1466	0.9466	−0.9758	−0.685
28	6.9671	0.9564	−0.9652	−0.6765	76	7.1545	0.956	−0.968	−0.7093
29	7.0891	0.9571	−0.9648	−0.6691	77	6.9454	0.9473	−0.9591	−0.6868
30	7.0759	0.9487	−0.9712	−0.7038	78	7.0249	0.9494	−0.9603	−0.7016
31	7.0496	0.9579	−0.9665	−0.6851	79	7.0989	0.956	−0.9571	−0.68
32	6.9661	0.9421	−0.9614	−0.6288	80	7.1336	0.9661	−0.9697	−0.7055
33	7.2776	0.9547	−0.9588	−0.7411	81	7.1208	0.968	−0.9631	−0.686
34	6.8697	0.9531	−0.971	−0.6503	82	7.1551	0.9685	−0.9713	−0.737
35	7.2938	0.9561	−0.9661	−0.7087	83	6.9564	0.9583	−0.9638	−0.6756
36	6.9934	0.9543	−0.961	−0.7026	84	7.1512	0.9496	−0.9649	−0.6969
37	7.1151	0.9545	−0.9641	−0.6916	85	7.0909	0.945	−0.9696	−0.6817
38	7.1185	0.9554	−0.9673	−0.6762	86	6.9989	0.9461	−0.9636	−0.6907
39	6.9873	0.9584	−0.9661	−0.69	87	7.0673	0.9494	−0.9697	−0.688
40	7.0552	0.9493	−0.9635	−0.6868	88	7.0741	0.9495	−0.9611	−0.6648
41	7.2372	0.9544	−0.9669	−0.722	89	7.1304	0.9542	−0.9704	−0.7008
42	6.9862	0.9515	−0.9619	−0.667	90	7.0863	0.9455	−0.9587	−0.6852
43	7.0901	0.9551	−0.9586	−0.693	91	7.1075	0.9504	−0.962	−0.7066
44	7.0438	0.9531	−0.962	−0.708	92	7.0741	0.9534	−0.9761	−0.6834
45	7.0553	0.9517	−0.9625	−0.6809	93	7.2857	0.9491	−0.9676	−0.6904
46	7.0447	0.9509	−0.9646	−0.6727	94	6.883	0.958	−0.9587	−0.6762
47	7.3077	0.9685	−0.9679	−0.7102	95	7.1757	0.9499	−0.9766	−0.7105
48	7.0477	0.9698	−0.9671	−0.6916	96	7.1995	0.955	−0.9631	−0.6822

Appendix C

Additional Results for the Hopf Normal Form

See Tables C1, C2.

TABLE C1 | Identified coefficients of Hopf normal form system using CS-MIO under Type 1 noise.

Noise: σ	SNR	y x	μx μy	x^3 x^2y	xy^2 y^3
0.001	137583.905	−1.0000	1.0000	−1.0000	−1.0000
		1.0000	1.0000	−1.0000	−1.0000
0.01	1375.839	−1.0000	1.0000	−0.9999	−0.9998
		1.0000	1.0000	−0.9995	−1.0000
0.1	13.758	−1.0001	1.0003	−0.9992	−0.9981
		0.9998	0.9996	−0.9952	−1.0001
0.3	1.529	−1.0002	1.0009	−0.9975	−0.9942
		0.9995	0.9988	−0.9857	−1.0003
0.5	0.550	−1.0004	1.0015	−0.9959	−0.9904
		0.9991	0.9980	−0.9762	−1.0005
0.7	0.281	−1.0005	1.0021	−0.9942	−0.9866
		0.9988	0.9972	−0.9667	−1.0008
1	0.138	−1.0007	1.0031	−0.9918	−0.9808
		0.9983	0.9961	−0.9525	−1.0011
2	0.034	−1.0014	1.0061	−0.9835	−0.9616
		0.9966	0.9921	−0.9049	−1.0022
3	0.015	−1.0021	1.0092	−0.9753	−0.9424
		0.9949	0.9882	−0.8574	−1.0033

TABLE C2 | Identified coefficients of Hopf normal form system using CS-MIO under Type 2 noise.

Noise: σ	SNR	y x	μx μy	x^3 x^2y	xy^2 y^3
0.001	120306.233	−0.9951	0.9680	−0.9681	−0.9680
		0.9951	0.9681	−0.9681	−0.9682
0.003	13367.359	−0.9949	0.9545	−0.9542	−0.9543
		0.9949	0.9555	−0.9554	−0.9552
0.005	4812.249	−0.9948	0.9289	−0.9282	−0.9284
		0.9947	0.9291	−0.9287	−0.9283
0.007	2455.229	−0.9946	0.8925	−0.8913	−0.8915
		0.9945	0.8913	−0.8905	−0.8898
0.010	1203.062	−0.9942	0.8237	−0.8215	−0.8218
		0.9940	0.8201	−0.8187	−0.8175
0.013	711.871	−0.9937	0.7459	−0.7425	−0.7429
		0.9934	0.7404	−0.7381	−0.7364
0.015	534.694	−0.9932	0.6928	−0.6887	−0.6891
		0.9929	0.6865	−0.6836	−0.6816

Appendix D

Additional Results for the Logistic Map

See Table D1.

TABLE D1 | Identified coefficients of the logistic map system using CS-MIO.

Noise: σ	SNR	rx_n	rx_n^2
0.001	48377.506	1.0000	−1.0000
0.01	481.877	0.9999	−0.9999
0.1	8.985	0.9902	−0.9862
0.2	3.619	0.9543	−0.9386
0.3	2.146	0.9212	−0.8956
0.4	1.455	0.8759	−0.8382
0.5	1.098	0.8406	−0.7907
0.6	0.874	0.8031	−0.7443