Compressive-sensing-assisted mixed integer optimization for dynamical system discovery with highly noisy data

Zhongshun Shi\textsuperscript{a}, Hang Ma\textsuperscript{a}, Hoang Tran\textsuperscript{b}, Guannan Zhang\textsuperscript{b}

\textsuperscript{a}Department of Industrial and Systems Engineering, University of Tennessee Knoxville, Knoxville, TN 37996
\textsuperscript{b}Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831

Abstract

The identification of governing equations for dynamical systems is everlasting challenges for the fundamental research in science and engineering. Machine learning has exhibited great success to learn and predict dynamical systems from data. However, the fundamental challenges still exist: discovering the exact governing equations from highly noisy data. In 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 hereby 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 comparing with state-of-the-art method. We also show its effectiveness for high-dimensional dynamical system identification through the chaotic Lorenz 96 system.

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

1. Introduction

Governing equations of the ubiquitous dynamical systems are of critical significance to shape our comprehension of the physical world. Traditional regime of obtaining the equations respects to 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 dataset due to the advances of sensors and technology, a new paradigm of discovering governing
equations purely from data has been evolved. Machine learning plays the pivotal role under this paradigm with a wide scope of methods including symbolic regression [4, 31], Gaussian processes [25], deep neural network [1, 28, 24, 9, 26], Bayesian inference [36], etc.

Even though neural networks have been proved to be an effective tool in learning and predicting trajectories 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 exploit sparse regression approaches [5, 27, 6, 17, 7, 18]. 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 [5] introduced a novel architecture called Sparse Identification of Nonlinear Dynamical Systems (SINDy), which used a sequential threshold least squares (or ridge regression [11]) 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. On this regard, studies have been conducted from many perspectives, including Lasso-based approach with a dictionary of partial derivatives [29], \( \ell_2,1 \) norm for data with highly corrupted segments [34], weak SINDy and discretization accounting for white noise [20], integral formulation of the differential equation [30], weak formulation with the orthogonal matching pursuit [23] and compressed sensing technique [35], to name a few. However, these methods perform terms selection essentially via imposing penalty on coefficients, subject to which they are usually sensitive to noise, and unable to control the exact level of sparsity for differential 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 \( \ell_0 \) norm constrained MIO problem is non-convex and \( \mathcal{NP} \)-hard [22], corresponding to the best subset selection in the larger statistics community [21, 2]. The \( \mathcal{NP} \)-hardness of the problem has contributed to the belief that discrete optimization problems were intractable [3]. For this reason, plenty of impressive sparsity-promoting techniques have focused on computationally feasible algorithms for solving the approximations, including Lasso [33], Elastic-net [37], non-convex regularization [15, 19] and stepwise regression [13]. These approximations induce obscure sparsity via regularization that often includes a large set of active terms (many are correlated terms and the coefficients are shrunk 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 to control the exact level of sparsity via setting the value of \( k \). When 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 [2]. Indeed, there is nothing more important than correct terms 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 coefficients shrinkage for solving the sparse regression problem in governing equations identification.

We propose a discrete optimization based method for exact recovery of differential equations under large noise. Our method takes advantages 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 terms 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 [5], 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 the proposed method can recover exact governing equations with up to two orders of magnitude larger noise comparing with state-of-the-art method. This shows 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 system

We introduce the data-driven dynamical system discovery problem from the perspective of sparse recovery [5]. We define 

\[ J = \{1, 2, \cdots, J\} \]

for any \( J \in \mathbb{Z}_+ \) throughout this paper.

We consider the following dynamical system consisting of \( J \) state variables, i.e.,

\[ \frac{d}{dt} \mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)), \quad (1) \]

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 \( \mathbf{f}(\mathbf{x}(t)) = [f_1(\mathbf{x}(t)), \cdots, f_J(\mathbf{x}(t))] \in \mathbb{R}^{1 \times J} \) represents the forcing term with \( f_j(\mathbf{x}(t)) \) being the forcing term of the \( j \)-th state variable \( x_j \) for \( j \in J \). A dictionary \( \theta(\mathbf{x}) \) consisting a total of \( P \) terms, denoted by

\[ \theta(\mathbf{x}) := [\theta_1(\mathbf{x}), \cdots, \theta_P(\mathbf{x})], \quad (2) \]

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

This work is based upon two assumptions that were also used in [5]. The first is that we assume the right-hand-side \( \mathbf{f} \) in Eq. (1) lives in the function space expanded by the dictionary \( \theta(\mathbf{x}) \) in Eq. (2). In other words, there exists a coefficient matrix \( \Xi := [\xi_1, \cdots, \xi_J] \in \mathbb{R}^{P \times J} \) such that

\[ f_j(\mathbf{x}) = \theta(\mathbf{x}) : \xi_j, \quad \text{for } j = 1, \cdots, J. \quad (3) \]

We remark that the definition of the dictionary \( \theta(\mathbf{x}) \) 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 \( \theta(\mathbf{x}) \). Since this work is to study sparse recovery of \( \mathbf{f}(\mathbf{x}) \), how to properly choosing \( \theta(\mathbf{x}) \) to ensure Eq. (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, i.e., very sparse in the function space expanded by the dictionary \( \theta(\mathbf{x}) \), regardless of the dimensionality \( J \). Specifically, to indicate the presence of
each term of $\Theta(x)$ in the right hand side $f$, we introduce the following indicator matrix

$$\Gamma := [\gamma_1, \cdots, \gamma_J] = \begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1J} \\ \vdots & \ddots & \vdots \\ \gamma_{P1} & \cdots & \gamma_{PJ} \end{bmatrix}, \quad \gamma_{pj} := \begin{cases} 1, & \text{if } f_j(x) \text{ includes } \theta_p(x), \\ 0, & \text{otherwise}, \end{cases}$$

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

$$k = [k_1, k_2, \cdots, k_J],$$

where $k_j$ is the number of non-zeros in $\gamma_j$. When the dynamical system satisfies the above two assumptions, Eq. (1) can be written as

$$\dot{x}(t) = \Theta(x(t))(\Gamma \circ \Xi),$$

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

2.2. The noisy data

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

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

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

$$X^{\text{noisy}} := X + U \quad \text{and} \quad \dot{X}^{\text{noisy}} := \dot{X} + V,$$

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

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

$$\Theta^{\text{noisy}} := [\theta_1(X^{\text{noisy}}), \ldots, \theta_P(X^{\text{noisy}})].$$

Similar to the standard SINDy method in [5], we assume the entries of the noise matrices $U$ and $V$ in Eq. (8) are independent and identically distributed (i.i.d.) Gaussian random variables with zero mean and standard deviation $\sigma$. In this work, we are particularly interested in the scenario with relatively large standard deviation of the noises, i.e., low signal-to-noise ratio. Details about the definition of the noises are give in Section 4.
The goal of sparse recovery of the dynamical system in Eq. (6) is to correctly identify $\Gamma$ and calculate the non-zero elements of $\Xi$ from measurement data of $x$ and/or $\dot{x}$. As discussed in Section 1, existing work on sparse recovery of dynamical systems, e.g., [5, 27, 6, 17, 7, 18, 35], perform term selection and promote sparsity by imposing penalties on the coefficients. In other words, these methods try to recover the product $\Gamma \circ \Xi$ 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 $f$ in the dictionary $\theta(x)$. The motivation of this work is to recover $\Gamma$ and $\Xi$ separately, where $\Gamma$ is recovered by solving a compressive-sensing-assisted mixed integer optimization, 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 Eq. (6) can be set up as follows,

$$
\min_{\Gamma, \Xi} \| \dot{X}^{\text{noisy}} - \Theta^{\text{noisy}} (\Gamma \circ \Xi) \|_2^2, \quad \text{s.t.} \quad \Gamma^T e \leq k^{\text{max}},
$$

where $e \in \mathbb{R}^{P \times 1}$ is a vector with all the entries to be one, such that the product $\Gamma^T e$ are exactly the cardinality constraints to indicate the active terms in each equation, and $k^{\text{max}} = [k_1^{\text{max}}, \ldots, k_J^{\text{max}}]$ 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., $\Gamma$) the and the corresponding coefficients (i.e., $\Xi$) in a two stage manner. The indicator matrix $\Gamma$ is determined by mixed integer optimization. Once $\Gamma$ is chosen, we can estimate the corresponding components of $\Xi$ using the standard least-squares method. Nevertheless, when the size of the original dictionary, i.e., the number of columns of $\Gamma$, is large, it is computationally intractable for the state-of-the-art MIO algorithms. To resolve this issue, we propose to use compressive sensing, i.e., $\ell_1$ 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 $x$ in Eq. (1) as an example in the following derivation, which means we intend to use the $j$-th column of the data matrices $X^{\text{noisy}}$ and $\dot{X}^{\text{noisy}}$ to infer the $j$-th columns of $\Gamma$ and $\Xi$. For notational simplicity, we omit the subscript $j$ and use $\dot{x}^{\text{noisy}}, \gamma, \xi$ to represent the $j$-column of $\dot{X}^{\text{noisy}}, \Gamma$ and $\Xi$, respectively.

3.1. Compressive sensing for reducing the size of the dictionary $\Theta^{\text{noisy}}$

The goal of this subsection is to reduce the size of the original dictionary $\Theta^{\text{noisy}}$ in Eq. (9), so that the modern integer optimization solvers, e.g., CPLEX or Gurobi, can be used to determine the indicator vector $\gamma$. To this end, we first solve the following $\ell_1$ minimization problem:

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

where $\| \cdot \|_1$ is the $\ell_1$ norm and $\xi^{\text{CS}}$ is the recovered coefficient by the $\ell_1$ minimization. In this paper, we used LARS algorithm in [14] for $\ell_1$ minimization. Then, we define a subset, denoted by $S$, of $P = [1, 2, \ldots, P]$ based on the magnitude of the components of $\xi^{\text{CS}}$, i.e.,

$$
S := \left\{ i \in P \mid |\xi_i^{\text{CS}}| \geq \varepsilon, \xi_i^{\text{CS}} \in \xi^{\text{CS}} \right\},
$$
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
\[
\Theta_{\text{noisy}}^S := \{ \theta_i(\mathbf{X}^\text{noisy}) \in \Theta^\text{noisy} \mid i \in S \}, \\
\gamma_S := \{ \gamma_i \in \gamma \mid i \in S \}, \\
\xi_S := \{ \xi_i \in \xi, \mid i \in S \},
\]
respectively. We emphasize that the recovered coefficients $\xi^{\text{CS}}$ in solving the $\ell_1$ minimization problem is not used to determine the final estimation of the coefficients. Instead, it is only used to help screening and narrowing down the range of candidate terms for high-dimensional problems with large $P$.

### 3.2. Mixed-integer optimization for determining the indicator $\gamma_S$

We start from converting the problem in Eq. (10) into an MIO problem. Then, the MIO problem constrained by a given sparsity can be written as
\[
\begin{align*}
\min_{\xi_S, \gamma_S} & \quad \| \dot{\mathbf{x}}^\text{noisy} - \Theta^\text{noisy}_S (\gamma_S \circ \xi_S) \|_2^2 + \lambda_2 \| \xi_S \|_2^2 \\
\text{s.t.} & \quad \| \xi_S \|_\infty \leq B, \\
& \quad \gamma_S^T \mathbf{e} = k,
\end{align*}
\]
where $k$ denotes the sparsity of $\gamma_S$ and $B$ is the upper bound of the coefficient $\xi_S$, and the $L^2$ regularization term $\lambda_2 \| \xi_S \|_2^2$ is commonly added to help alleviate the influence of the measurement noises on the MIO optimization.

**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 $\gamma_S$. To resolve this issue, we standardize the data $\Theta^\text{noisy}$ and $\dot{\mathbf{x}}^\text{noisy}$, and use the standardized data in MIO.

**Remark 2.** We emphasize that splitting the coefficient of $\Theta^\text{noisy}_S$ into $\gamma_S$ and $\xi_S$ is to indicate that the goal of solving the MIO is only to determine $\gamma_S$, i.e., identify the correct terms in the reduced dictionary $\Theta^\text{noisy}_S$. Even though an MIO algorithm, e.g., the CPLEX, will also provide an estimate of $\xi_S$, we will not use the estimate as our final solution.

The goal of this subsection is to determine $\gamma_S$ by solving the MIO problem in Eq. ($P_0$). However, there are two hyperparameters, i.e., the sparsity $k$ and the $\ell_2$-norm weight $\lambda_2$, 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 $\gamma_S$.

We first define a tensor grid of $(k, \lambda_2)$. The grid for $k$ is easily defined as $\{1, 2, \ldots, k_{\text{max}}\}$ based on the maximum allowable sparsity $k_{\text{max}}$. The upper bound for $\lambda_2$, denoted by $\lambda_2^{\text{max}}$, is defined by the norm
\[
\lambda_2^{\text{max}} := \| (\Theta^\text{noisy})^T \dot{\mathbf{x}}^\text{noisy} \|_\infty.
\]
This is followed by setting a small ratio $r$ of $\lambda_2^{\text{max}}$ to set the minimum allowed value $\lambda_2^{\text{min}}$, i.e., $\lambda_2^{\text{min}} = r \lambda_2^{\text{max}}$. Empirically, if $N > P$, we set $r = 0.0001$; otherwise $r = 0.01$. Afterwards, we
uniformly sample $m$ values from interval $[\log (\lambda_{2\text{min}}), \log (\lambda_{2\text{max}})]$, where $m$ is practically set to 50 or 100. Then by taking exponentials of the sampled values, we obtain a set of $\ell_2$-norm weight, denoted by $\Lambda = \{\lambda_1^2, \ldots, \lambda_m^2\}$.

We next perform the cross validation to choose the best hyperparameters from the tensor grid of $(k, \lambda_2)$. Specifically, we evenly partition the data set $\{\hat{x}_{\text{noisy}}, \Theta_{\text{noisy}}\}$ with a total of $N$ measurements into $T$ disjoint subsets, denoted by $\{\hat{x}_{\text{noisy}}^1, \Theta_{\text{noisy}}^1\}, \ldots, \{\hat{x}_{\text{noisy}}^T, \Theta_{\text{noisy}}^T\}$, respectively. For one subset $\{\hat{x}_{\text{noisy}}^t, \Theta_{\text{noisy}}^t\}$ and one pair of $(k, \lambda_2)$, the error of the MIO solution is defined by

$$
e_t(k, \lambda_2) := \|\hat{x}_{\text{noisy}}^t - \Theta_{\text{noisy}}^t (\gamma_{S,t} \circ \xi_{S,t})\|_2^2,$$

where $\gamma_{S,t}$ and $\xi_{S,t}$ are obtained by solving the MIO problem in Eq. (P_0) using the complementary data set $\{\hat{x}_{\text{noisy}} \setminus \hat{x}_{\text{noisy}}^t, \Theta_{S} \setminus \Theta_{S,t}\}$. The errors for other subsets and choices of $k, \lambda_2$ can be obtained similarly. Then the total error for the pair $(k, \lambda_2)$ is defined by

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

and the best hyperparameters are obtained by

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

The final step in this subsection is to solve the MIO problem with the best hyperparameters $(k^*, \lambda_2^*)$ to obtain the optimized indicator vector, denoted by $\gamma_S^*$.

After the optimal $\gamma_S^*$ is determined using the MIO method, we use the standard least-squares approach to estimate the coefficient $\xi_S$. In this case, we use the original data, not the standarized data used in the MIO method, to solve the following least-squares problem

$$\xi_S^* = \arg \min_{\xi_S} \|\hat{x}_{\text{noisy}} - (\Theta_{\text{noisy}} \gamma_S^*) \xi_S\|_2^2,$$

where the matrix $\Theta_{\text{noisy}} \gamma_S^*$ only contains the columns of $\Theta_{\text{noisy}}$ identified by $\gamma_S^*$.

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 present study, we fully take advantages of state-of-the-art algorithm in modern optimization solver CPLEX for solving the MIO problems. 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 experiment studies.
Algorithm 1: The CS-MIO algorithm

Input: The noisy data $X_{\text{noisy}}$, $\dot{X}_{\text{noisy}}$

1. Construct matrix $\Theta_{\text{noisy}}$ by evaluating $\theta(x)$ at the data points in $X_{\text{noisy}}$;
2. Standardize the columns of $\Theta_{\text{noisy}}$ and $\dot{X}_{\text{noisy}}$ to have zero means and unit variance;
3. for $j \in J$ do
4. if $P > P_{\text{max}}$ then /* Compressive sensing-base dictionary reduction */
5. Construct the reduced dictionary $S$ in Eq. (13) by solving Eq. (12)
6. end
7. for $k = 1, 2, \ldots, k_{\text{max}}$ do /* MIO for determining $\gamma_S$ in Eq. $(P_0)$ */
8. Construct a grid $\Lambda$ of $\lambda_2$ in Eq. $(P_0)$;
9. Divide $\{x_{\text{noisy}}, \Theta_{\text{noisy}}S\}$ into $T$ disjoint subsets;
10. for $\lambda_2 \in \Lambda$ do
11. for $t = 1, \ldots, T$ do
12. Solve the MIO problem in Eq. $(P_0)$ using $k$ and $\lambda_2$;
13. Compute the error $e_t(k, \lambda_2)$ in Eq. (16);
14. end
15. Compute total error $E(k, \lambda_2)$ in Eq. (17);
16. end
17. Find the best hyperparameters $(k^*, \lambda_2^*) = \arg \min_{(k, \lambda_2)} E(k, \lambda_2)$
18. Identify the optimal indicator $\gamma_S^*$ by solving the problem $(P_0)$ using $(k^*, \lambda_2^*)$;
19. Determine the optimal coefficient $\xi_S^*$ by solving the problem in Eq. (19);
20. Set $\gamma_S^*$ and $\xi_S^*$ as the $j$-th column of $\Gamma^*$ and $\Xi^*$, respectively;
21. end
22. Return $\dot{x} = \theta(x)(\Gamma^* \circ \Xi^*)$

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 [5] as the testing problems, including chaotic Lorenz 3 system, vortex shedding after a cylinder, bifurcation dynamical systems like Hopf normal form and logistic map. In addition, we also study the high-dimensional Lorenz 96 system. We compare CS-MIO with state-of-the-art method SINDy, specifically, the Python version solver PySINDy [12, 16]. For all the example systems, the experiments are deployed on a mobile workstation with Intel(R) Xeon(R) W-10885M CPU @ 2.40GHz, 128 GB memory, 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 nu-
merica 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,

\[
\text{SNR} := \frac{1}{J} \sum_{j=1}^{J} \frac{\text{Var}(S_j)}{\text{Var}(Z_j)},
\]

(20)

where \( S_j \in \{X_j, \dot{X}_j\} \) is the signal data, i.e., the \( j \)-th column of matrices \( X \) or \( \dot{X} \), and \( Z_j \in \{U_j, V_j\} \) are the additive Gaussian noise, i.e., the \( j \)-th column of matrices \( U \) or \( V \) in Eq. (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 \( x \) and time derivatives \( \dot{x} \) can be measured; Gaussian noise is added to \( \dot{x} \).

- **Type 2 Noise**: Only state variables \( x \) can be measured. Gaussian noise is added to \( x \). The time derivatives \( \dot{x} \) are computed by total variation derivative (TVD) [8].

We use state-of-the-art algorithm in modern optimization solver CPLEX (Python package docplex) for solving the MIO problems. 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 Eq. (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 \( B \geq \|\xi^\star\|_{\infty} \). This setting is, however, not applicable because \( \xi^\star \) is not known a prior. Some methods have been studied to set \( B \) values by finding the upper bound of \( \xi^\star \) using data-driven manners such as cumulative coherence function and solving convex optimization methods [2]. In this paper, we use a losing upper bound \( B = 1000 \) for all the examples. Besides, we set the timelimit to be 600 seconds, and the mipgap to be 0 for the invoked branch-and-cut algorithm in docplex. This refers to that if the branch-and-cut finds a solution within 600 seconds, 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 \( \Gamma \) in Eq. (10) by the number of exactly recovered equations of the target dynamical system, defined by

\[
A(\Gamma) := \sum_{j=1}^{J} 1_{\gamma_j = \gamma_j^\dagger}, \quad \text{with} \quad 1_{\gamma_j = \gamma_j^\dagger} = \begin{cases} 
1, & \text{if } \gamma_j := \gamma_j^\dagger, \\
0, & \text{if } \gamma_j \neq \gamma_j^\dagger.
\end{cases}
\]

(21)
where $\gamma_j^\dagger$ is the ground truth and $\gamma_j$ is recovered by a method. The exact recovery for the entire dynamical system occurs when $A(\Gamma) = J$. When the exact $\Gamma$ can be recovered, we evaluate the accuracy of the approximation of the coefficients in $\Xi$ in Eq. (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:

\[ \dot{x} = \alpha (y - x), \]
\[ \dot{y} = x(\rho - z) - y, \]
\[ \dot{z} = xy - \beta z. \]

(22)
\[ (23) \]
\[ (24) \]

With $\sigma = 10$, $\beta = 8/3$ and $\rho = 28$, the Lorenz 3 system performs chaotically. We generate the data using initial point $(x, y, z) = (-8,8,27)$ with time step $\Delta t = 0.001$ in $t \in [0, 60]$. A set of noise standard deviation $\sigma$ 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 $\dot{x}$ with $\sigma$ ranging from 1 to 3000. SNR is computed by the added noise and $\dot{x}$. When under Type 2 noise, the Gaussian noise is added to $x$ with $\sigma$ ranging from 0.01 to 20, and the SNR is computed by the added noise and $x$. In this case, $\dot{x}$ is smoothed using total variation derivative (TVD) of [8]. The comparison results of CS-MIO and PySINDy for both cases are presented in Tables 1a and 1b, respectively.

Table 1: Comparison of the number of exactly recovered equations, i.e., the metric $A(\Gamma)$ in Eq. (21), for the Lorenz 3 system. Compared with PySINDy, our CS-MIO method can correctly recover all the three equations, i.e., identifying the correcting $\Gamma$ in Eq. (6), under smaller SNR values.

| 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      |

| Noise std | SNR    | PySINDy | CS-MIO |
|-----------|--------|---------|--------|
| 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      |

(a) Results under Type 1 noise. (b) Results under Type 2 noise.

Table 1a and 1b 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 differential 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 white noise added to $x$ is no longer Gaussian after using numerical differentiation and is difficult to handle. Thus, the performance of both CS-MIO and PySINDy is downgraded at smaller SNRs.
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, but PySINDy picked up incorrect terms.

|                | Ground Truth | PySINDy | CS-MIO |
|----------------|--------------|---------|--------|
|                | $\dot{x} = -10x + 10y$ | $\dot{x} = -6.62 - 13.62x + 11.80y + 0.38z - 0.14x^2 + 0.23xy + 0.11xz - 0.11y^2 - 0.05yz$ | $\dot{x} = -9.72x + 9.70y$ |
|                | $\dot{y} = 28x - y - xz$ | $\dot{y} = 3.46 + 29.32x - 1.32y - 0.05z - 1.03xz$ | $\dot{y} = 29.28x - 1.31y - 1.03xz$ |
|                | $\dot{z} = -\frac{8}{7}z + xy$ | $\dot{z} = 7.14 - 0.13x + 0.15y - 2.28z - 0.08x^2 + 1.05xy$ | $\dot{z} = -2.64z + 1.00xy$ |

(a) Results under Type 1 noise with $\sigma=300$.

(b) Results under Type 2 noise with $\sigma=2$.

Tables 2a and 2b show the discovered equations by CS-MIO and PySINDy under Type 1 noise (at noise magnitude 300) and Type 2 noise (at noise magnitude 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 remains small deviation 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 PySINDy identified system under Type 1 noise at $\sigma=300$. It is seen the trajectory starts to deviate the ground truth right at the beginning, shown by the red dot. In contrast, the trajectory of CS-MIO identified system can coincide for longer time well with the ground truth, as shown in Figure 1b. Appendix A gives more details of identified models of Lorenz 3 system by CS-MIO.

Figure 1: (a) and (b) show the trajectories of PySINDy and CS-MIO discovered equations in Table 2a, respectively. The trajectory of 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 in Table 2b, respectively.
4.3. The chaotic Lorenz 96 system

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

\[
\dot{x}_j = (x_{j+1} - x_{j-2})x_{j-1} - x_j + F,
\]

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 \( x(0) = 1 \) with a small perturbation 0.01 added to \( x_1(0) \) to generate the dataset with time step \( \Delta t = 0.01 \) in \( t \in [0, 600]. \) We use second order polynomials in CS-MIO for the Lorenz 96 system with 96 variables, which results in 4752 polynomial terms. This leads to huge difficulties to deal with the high dimension.

For this high-dimensional Lorenz 96 system, we use compressive sensing approach as described in Eq. (12) for pre-selecting a subset of at most \( S = 100 \) significant terms from 4752 candidate terms. We use the LassoLars algorithm from 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 times, the subset of terms resulting from 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 \( S. \)

Table 3: Comparison of the number of exactly recovered equations, i.e., the metric \( A(\Gamma) \) in Eq. (21), for the Lorenz 96 system. Compared with PySINDy, our CS-MIO method can correctly recover all the 96 equations, i.e., identifying the correcting \( \Gamma \) in Eq. (6), under smaller SNR values.

| Noise std | SNR | The metric \( A(\Gamma) \) | PySINDy | CS-MIO |
|-----------|-----|---------------------------|--------|--------|
| 1         | 352.147 | 96                       | 96     |        |
| 10        | 3.521   | 96                       | 96     |        |
| 20        | 0.880   | 20                       | 96     |        |
| 30        | 0.391   | 0                         | 96     |        |
| 40        | 0.220   | 0                         | 96     |        |
| 50        | 0.141   | 0                         | 88     |        |
| 70        | 0.072   | 0                         | 9      |        |
| 150       | 0.016   | 0                         | 9      |        |
| 230       | 0.007   | 0                         | 0      |        |
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 do not show the figure for SINDy since its identified equations result in unstable attractor. More details for the identified models of Lorenz 96 system by CS-MIO are in Appendix B.

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 [5]. The first is the 2D Hopf normal form with bifurcation parameter $\mu$,

\begin{align}
\dot{x} &= \mu x - \omega y + Ax(x^2 + y^2), \\
\dot{y} &= \omega x + \mu y + Ay(x^2 + y^2).
\end{align}

To handle the bifurcation behaviors, the $\mu$ in the Hopf normal form is treated as additional state variables by adding dummy differential equation $\dot{\mu} = 0$ to the system [5]. By adopting this setting, we used 14 values of $\mu$ to generate 14 datasets, with 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 a function of state $x$ and bifurcation parameter $\mu$, i.e., $\dot{x} = f(x, \mu)$. 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 the lowest SRN can be as low as 0.015 for CS-MIO to exactly recovered 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. More details for the identified models of Hopf normal form by CS-MIO are in Appendix C.

Table 4: Comparison of the number of exactly recovered equations, i.e., the metric \( A(\Gamma) \) in Eq. (21), for Hopf Normal form. Compared with PySINDy, our CS-MIO method can correctly recover all the two equations, i.e., identifying the correcting \( \Gamma \) in Eq. (6), under smaller SNR values.

| Noise std | SNR | PySINDy | CS-MIO | Noise std | SNR | PySINDy | CS-MIO |
|-----------|-----|---------|--------|-----------|-----|---------|--------|
| 0.1       | 13.758 | 2       | 2      | 0.001     | 120306.233 | 2     | 2      |
| 0.3       | 1.529  | 2       | 2      | 0.003     | 13367.359  | 2     | 2      |
| 0.5       | 0.550  | 1       | 2      | 0.005     | 4812.249   | 2     | 2      |
| 0.6       | 0.281  | 1       | 2      | 0.007     | 2455.229   | 2     | 2      |
| 1         | 0.138  | 0       | 2      | 0.010     | 1203.062   | 0     | 2      |
| 2         | 0.034  | 0       | 2      | 0.013     | 711.871    | 0     | 2      |
| 3         | 0.015  | 0       | 2      | 0.015     | 534.694    | 0     | 2      |
| 4         | 0.009  | 0       | 0      | 0.017     | 416.285    | 0     | 0      |

Figure 3: Trajectory 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 \( \sigma=0.015 \).

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

\[
x_{n+1} = rx_n(1 - x_n) + \eta_n.
\]  

(28)

where \( \eta_n \) is the stochastic forcing and \( r \) is the bifurcation parameter. Similar manner is imposed to the bifurcation parameter \( r \) in the logistic map with 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 1000 discrete steps. Note the logistic map is a discrete time dynamical system, so that there is only one manner for adding noise (herein \( \eta_n \)) to the state variables \( x \). 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
Table 5: Comparison of the number of exactly recovered equations, i.e., the metric $A(\Gamma)$ in Eq. (21), for the logistic map. Compared with PySINDy, our CS-MIO method can correctly recover the single equation, i.e., identifying the correcting $\Gamma$ in Eq. (6), under smaller SNR values.

| 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      |

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

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. As discussed in [5], 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.

\[
\dot{x} = \mu x - \omega y + A x z, \quad (29) \\
\dot{y} = \omega x + \mu y + A y z, \quad (30) \\
\dot{z} = -\lambda (z - x^2 - y^2). \quad (31)
\]

Herein we use the same dataset used in [5], which is originally generated using direct numerical simulations of the 2D Navier-Stokes equations originally by [32, 10]. We do not
Table 6: Identified coefficients of 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 $\Gamma$ in comparison with PySINDy.

| Term | Equation 1 | Equation 2 | Equation 3 |
|------|------------|------------|------------|
|      | 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          | 0          |
| $xz$ | 0.0002     | 0.0002     | 0.0022     | 0.0022     | 0          | 0          |
| $y^2$| 0          | 0          | 0          | 0          | 0          | 0          |
| $yz$ | -0.0019    | -0.0019    | -0.0018    | -0.0018    | 0          | 0          |
| $z^2$| 0          | 0          | 0          | 0          | -0.0011    | -0.0010    |

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.

| Term  | Equation 1 | Equation 2 | Equation 3 |
|-------|------------|------------|------------|
|       | PySINDy    | CS-MIO     | PySINDy    | CS-MIO     | PySINDy    | CS-MIO     |
| Bias  | 0          | 0          | 0          | 0          | -9.66082   |
| $x$   | 0          | 0          | 0          | 1.02896    | 0          |
| $y$   | -1.04203   | -0.21545   | 0.00621    | 0.24547    | 0.00025    |
| $z$   | 0.00002    | 0          | -0.00004   | 0          | 0.47502    |
| $x^2$ | 0          | 0          | 0          | 0          | 0.00006    |
| $xy$  | 0          | 0          | 0          | 0          | 0.00047    |
| $xz$  | 0.00138    | 0.00275    | -0.00744   | 0.00022    | 0          |
| $y^2$ | 0          | 0          | 0          | 0          | -0.00019   |
| $yz$  | -0.00367   | 0.00396    | -0.00366   | 0          | 0          |
| $z^2$ | 0          | 0          | 0          | 0          | -0.00006   |
| $x^3$ | 0          | 0          | 0          | 0          | 0.00532    |
| $x^2y$| 0          | -0.00004   | 0          | -0.00001   | 0          |
| $x^2z$| 0          | 0          | 0          | 0          | -0.00003   |
| $xy^2$| 0          | 0          | 0          | 0          | 0.00005    |
| $xyz$ | 0          | 0          | 0          | 0          | -0.00002   |
| $xz^2$| 0.00001    | 0.00002    | -0.00002   | 0          | 0          |
| $y^3$ | 0          | -0.00004   | 0          | -0.00001   | 0          |
| $y^2z$| 0          | 0          | 0          | 0          | -0.00002   |
| $yz^2$| 0          | -0.00002   | 0          | -0.00002   | 0          |
| $z^3$ | 0          | 0          | 0          | 0          | 0.00001    | 0.00001    |
employ either Type 1 or Type 2 noise instead of using this single dataset. We identify the
differential 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 differential equations. However, it is 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 less nonlinear terms to include all those in the ground truth than PySINDy.

If $\lambda$ is large, then the dynamics on $z$ 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 $z$ coordinate and then correcting to the parabolic slow manifold. If substituting $z = x^2 + y^2$
into Equations 29 and 30, we obtain a Hopf normal form system on the slow manifold,
which include 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 the CS-MIO identified system
agree 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 gov-
erning 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 grounded on the important foundation,
that is, the identification of terms in the governing equations is essentially a discrete opti-
mization 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 penalty on the coefficients. We also
combine the mixed-integer optimization with compressive sensing and other regularization

![Figure 5: Trajectories of the full simulation and the CS-MIO identified system for cylinder dynamics.](image)
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 outperforms 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.

Acknowledgement

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.

References

[1] 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(31):15344–15349, 2019.

[2] D. Bertsimas, A. King, R. Mazumder, et al. Best subset selection via a modern optimization lens. *Annals of statistics*, 44(2):813–852, 2016.

[3] D. Bertsimas, J. Pauphilet, and B. Van Parys. Rejoinder: Sparse regression: Scalable algorithms and empirical performance. *Statistical Science*, 35(4):623–624, 2020.

[4] J. Bongard and H. Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

[5] 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(15):3932–3937, 2016.

[6] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Sparse identification of nonlinear dynamics with control (sindyc). *IFAC-PapersOnLine*, 49(18):710–715, 2016.

[7] 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(45):22445–22451, 2019.

[8] R. Chartrand. Numerical differentiation of noisy, nonsmooth data. *International Scholarly Research Notices*, 2011, 2011.
[9] Z. Chen and D. Xiu. On generalized residual network for deep learning of unknown dynamical systems. *Journal of Computational Physics*, 438:110362, 2021.

[10] 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(25-28):2131–2146, 2008.

[11] B. de 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, 2020.

[12] B. de 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(49):2104, 2020.

[13] N. R. Draper and H. Smith. *Applied regression analysis*, volume 326. John Wiley & Sons, 1998.

[14] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[15] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

[16] A. A. Kaptanoglu, B. M. de Silva, U. Fasel, K. Kaheman, A. J. Goldschmidt, J. L. Callaham, C. B. Delahunt, Z. G. Nicolaou, K. Champion, J.-C. Loiseau, J. N. Kutz, and S. L. Brunton. Pysindy: A comprehensive python package for robust sparse system identification. *arXiv preprint arXiv:2111.08481*, 2021.

[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*, 844:459–490, 2018.

[18] 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(2223):20180534, 2019.

[19] R. Mazumder, J. H. Friedman, and T. Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495):1125–1138, 2011.

[20] D. A. Messenger and D. M. Bortz. Weak sindy: Galerkin-based data-driven model selection. *Multiscale Modeling & Simulation*, 19(3):1474–1497, 2021.

[21] A. Miller. *Subset selection in regression*. CRC Press, 2002.

[22] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[23] Y. Pantazis and I. Tsamardinos. A unified approach for sparse dynamical system inference from temporal measurements. *Bioinformatics*, 35:3387 – 3396, 2019.
[24] T. Qin, Z. Chen, J. D. Jakeman, and D. Xiu. Data-driven learning of nonautonomous systems. *SIAM Journal on Scientific Computing*, 43(3):A1607–A1624, 2021.

[25] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[26] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Multistep neural networks for data-driven discovery of nonlinear dynamical systems. *arXiv preprint arXiv:1801.01236*, 2018.

[27] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[28] 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*, 396:483–506, 2019.

[29] H. Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[30] H. Schaeffer and S. G. McCalla. Sparse model selection via integral terms. *Phys. Rev. E*, 96:023302, Aug 2017.

[31] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[32] K. Taira and T. Colonius. The immersed boundary method: a projection approach. *Journal of Computational Physics*, 225(2):2118–2137, 2007.

[33] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[34] G. Tran and R. Ward. Exact recovery of chaotic systems from highly corrupted data. *Multiscale Modeling & Simulation*, 15(3):1108–1129, 2017.

[35] 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 15:154101, 2011.

[36] 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(2217):20180305, 2018.

[37] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.
Appendix A  Additional results for the Chaotic Lorenz 3 System

We provide additional results for the Chaotic Lorenz 3 system.

Table 8: Identified coefficients of Lorenz 3 system using CS-MIO under Type 1 noise.

| Noise: $\sigma$ | SNR        | $x$     | $y$     | $x$ | $y$ | $xz$ |
|-----------------|------------|---------|---------|-----|-----|------|
| 0.01            | 4191.4317.129 | -10.0000 | 10.0000 | 28.0000 | -1.0000 | -1.0000 |
|                 |             | -2.6667 | 1.0000  |     |     |      |
| 0.1             | 4191.43.171 | -9.9999 | 9.9999  | 28.0043 | -1.0010 | -1.0000 |
|                 |             | -2.6667 | 1.0000  |     |     |      |
| 1               | 4191.621    | -9.9991 | 9.9990  | 28.0034 | -1.0001 | -1.0001 |
|                 |             | -2.6666 | 1.0000  |     |     |      |
| 10              | 41.916      | -9.9908 | 9.9908  | 28.0426 | -1.0103 | -1.0009 |
|                 |             | -2.6658 | 1.0000  |     |     |      |
| 50              | 1.677       | -9.9540 | 9.9499  | 28.2131 | -1.0516 | -1.0044 |
|                 |             | -2.6624 | 1.0001  |     |     |      |
| 100             | 0.419       | -9.9081 | 9.8999  | 28.4263 | -1.0320 | -1.0089 |
|                 |             | -2.6581 | 1.0001  |     |     |      |
| 150             | 0.186       | -9.8621 | 9.8498  | 28.6394 | -1.1548 | -1.0133 |
|                 |             | -2.6538 | 1.0002  |     |     |      |
| 200             | 0.105       | -9.8161 | 9.7998  | 28.8525 | -1.2065 | -1.0177 |
|                 |             | -2.6495 | 1.0002  |     |     |      |
| 250             | 0.067       | -9.7701 | 9.7497  | 29.0657 | -1.2581 | -1.0222 |
|                 |             | -2.6452 | 1.0003  |     |     |      |
| 300             | 0.047       | -9.7242 | 9.6997  | 29.2788 | -1.3097 | -1.0266 |
|                 |             | -2.6409 | 1.0003  |     |     |      |
Figure 6: 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 \( \sigma \): 0.01, 1, 100, and 300. The exact recovery fails when \( \sigma \) is larger than 300. (a) Trajectories of the CS-MIO identified system (red dashed) and ground truth (blue solid). (b) \( \ell_2 \) error vs time of the trajectories of the recovered Lorenz 3 system (\( \hat{x}(t) \)) comparing to the ground truth (\( x(t) \)), i.e., \( \|\hat{x}(t) - x(t)\|_2^2 \) as a function of \( t \) from \( t = 0 \) to \( t = 20 \).
Table 9: Identified coefficients of Lorenz 3 system using CS-MIO under Type 2 noise.

| Noise: $\sigma$ | SNR     | $x$     | $y$     | $xz$    | $z$     | $xy$     |
|-----------------|---------|---------|---------|---------|---------|---------|
| 0.01            | 729427.159 | -9.9851 | 10.0000 | -0.8682 | -0.9939 | -2.6602 | 0.9997 |
| 0.05            | 29178.677  | -9.9852 | 9.9999  | -0.8682 | -0.9939 | -2.6602 | 0.9997 |
| 0.1             | 7295.616   | -9.9851 | 9.9997  | -0.8680 | -0.9938 | -2.6603 | 0.9997 |
| 0.5             | 292.848    | -9.9791 | 9.9934  | -0.8596 | -0.9929 | -2.6608 | 0.9999 |
| 1               | 73.982     | -9.9573 | 9.9730  | -0.8335 | -0.9906 | -2.6605 | 0.9997 |
| 2               | 19.255     | -9.8670 | 9.8916  | -0.7319 | -0.9823 | -2.6571 | 0.9984 |
Figure 7: 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 $\sigma$: 0.01, 0.1, 1 and 2. The exact recovery fails when $\sigma$ is larger than 2. (a) Trajectories of the CS-MIO identified system (red dashed) and ground truth (blue solid). (b) $\ell_2$ error vs time of the trajectories of the recovered Lorenz 3 system ($\hat{x}(t)$) comparing to the ground truth ($x(t)$), i.e., $||\hat{x}(t) - x(t)||^2_2$ as a function of $t$ from $t = 0$ to $t = 20$. 
Appendix B  Additional results for the Chaotic Lorenz 96 System

(a) Hovermøller plot for difference between the identified system and ground truth.

(b) $\ell_2$ error between the identified system and ground truth.

Figure 8: 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) Hovermøller 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 difference between the ground truth states $x_j(t)$ and the evolved states $\hat{x}_j(t)$ using the identified equations by CS-MIO, i.e., $\Delta x_j(t) = x_j(t) - \hat{x}_j(t)$ for $j \in [96]$. (b) $\ell_2$ error vs time of the trajectories of the CS-MIO recovered Lorenz 96 system from $t = 0$ to $t = 10$. The exact recovery fails when $\sigma$ is larger than 50.

25
Table 10: Identified coefficients of Lorenz 96 system using CS-MIO under Type 1 noise with magnitude 50: Part A.

| Equation Index $j$ | $F$ | $x_{j+1}x_{j-1}$ | $x_{j-2}x_{j-1}$ | $x_j$ |
|------------------|-----|------------------|------------------|------|
| 1                | 7.6636 | 0.9954          | -0.9919          | -0.9053 |
| 2                | 8.2107 | 1.0137          | -0.9957          | -1.0136 |
| 3                | 8.0884 | 1.0031          | -0.9776          | -1.0558 |
| 4                | 7.7594 | 0.9958          | -0.9925          | -1.0204 |
| 5                | 7.6274 | 0.9997          | -1.0020          | -0.9596 |
| 6                | 7.6735 | 1.0009          | -0.9790          | -0.9568 |
| 7                | 8.1661 | 1.0126          | -1.0122          | -1.0837 |
| 8                | 7.7873 | 1.0004          | -0.9930          | -0.9782 |
| 9                | 8.1277 | 0.9978          | -1.0040          | -1.0211 |
| 10               | 8.2015 | 1.0310          | -1.0276          | -0.9769 |
| 11               | 7.7901 | 0.9904          | -1.0143          | -0.8445 |
| 12               | 8.0504 | 0.9844          | -0.9902          | -1.0707 |
| 13               | 7.9518 | 0.9889          | -1.0144          | -1.0927 |
| 14               | 7.8521 | 1.0055          | -0.9897          | -0.9773 |
| 15               | 8.4207 | 0.9905          | -0.9984          | -1.0906 |
| 16               | 8.2164 | 1.0162          | -1.0171          | -0.9698 |
| 17               | 8.3087 | 0.9924          | -0.9893          | -1.0197 |
| 18               | 8.4646 | 1.0119          | -1.0123          | -1.0714 |
| 19               | 8.2501 | 0.9759          | -0.9886          | -1.0430 |
| 20               | 7.8796 | 0.9919          | -0.9873          | -0.9722 |
| 21               | 7.8357 | 0.9929          | -1.0093          | -0.9649 |
| 22               | 8.5176 | 0.9749          | -1.0268          | -1.1041 |
| 23               | 8.4018 | 1.0369          | -1.0221          | -1.0326 |
| 24               | 7.9646 | 0.9937          | -0.9830          | -1.0083 |
| 25               | 7.7756 | 0.9871          | -0.9960          | -0.9909 |
| 26               | 7.6644 | 0.9907          | -1.0124          | -1.0218 |
| 27               | 8.1968 | 1.0030          | -1.0100          | -1.0355 |
| 28               | 8.2314 | 0.9985          | -0.9925          | -1.0586 |
| 29               | 8.2346 | 1.0033          | -1.0061          | -1.0567 |
| 30               | 7.6806 | 1.0109          | -0.9958          | -0.8968 |
| 31               | 8.3632 | 1.0160          | -1.0046          | -1.0581 |
| 32               | 8.1515 | 1.0083          | -0.9858          | -0.9228 |
| 33               | 8.0453 | 1.0109          | -1.0234          | -0.9918 |
| 34               | 7.8578 | 1.0029          | -0.9876          | -0.9044 |
| 35               | 7.8325 | 1.0070          | -1.0029          | -0.9443 |
| 36               | 8.0977 | 0.9873          | -0.9925          | -0.9986 |
| 37               | 8.6882 | 0.9952          | -1.0037          | -1.0654 |
| 38               | 7.7677 | 0.9983          | -0.9685          | -1.0238 |
| 39               | 8.0614 | 0.9718          | -0.9919          | -0.9915 |
| 40               | 7.8614 | 1.0046          | -1.0125          | -1.0181 |
| 41               | 7.2833 | 0.9881          | -0.9548          | -0.9379 |
| 42               | 8.0186 | 0.9879          | -0.9965          | -1.0273 |
| 43               | 8.0823 | 0.9994          | -1.0185          | -1.0349 |
| 44               | 7.9811 | 0.9758          | -1.0048          | -1.0275 |
| 45               | 7.9975 | 0.9556          | -0.9970          | -0.9966 |
| 46               | 7.7917 | 0.9986          | -0.9901          | -0.9353 |
| 47               | 7.7668 | 0.9971          | -0.9891          | -1.0075 |
| 48               | 7.5492 | 1.0106          | -0.9717          | -1.0031 |
Table 11: Identified coefficients of Lorenz 96 system using CS-MIO under Type 1 noise with magnitude 50: Part B.

| Equation Index $j$ | $F$ | $x_{j+1}x_{j-1}$ | $x_{j-2}x_{j-1}$ | $x_j$ |
|-------------------|-----|------------------|------------------|-------|
| 49                | 8.1873 | 1.0016 | -1.0014 | -0.9289 |
| 50                | 7.8786 | 1.0036 | -1.0096 | -1.0100 |
| 51                | 7.7901 | 1.0235 | -0.9655 | -1.0325 |
| 52                | 7.4264 | 0.9828 | -0.9877 | -0.8955 |
| 53                | 7.7280 | 0.9898 | -0.9848 | -0.9488 |
| 54                | 8.0840 | 0.9995 | -0.9977 | -0.9669 |
| 55                | 8.8206 | 0.9805 | -1.0038 | -1.0711 |
| 56                | 8.5845 | 0.9934 | -1.0120 | -1.0968 |
| 57                | 8.1714 | 0.9805 | -0.9903 | -0.9710 |
| 58                | 8.5471 | 1.0031 | -1.0284 | -1.0648 |
| 59                | 8.2529 | 1.0088 | -0.9877 | -0.9947 |
| 60                | 8.2181 | 1.0111 | -1.0064 | -1.0332 |
| 61                | 8.3878 | 1.0310 | -1.0099 | -1.0172 |
| 62                | 8.4394 | 0.9898 | -1.0120 | -1.0398 |
| 63                | 8.4044 | 1.0326 | -0.9997 | -1.0862 |
| 64                | 8.1365 | 1.0058 | -1.0072 | -1.0033 |
| 65                | 7.9563 | 0.9895 | -0.9893 | -1.0268 |
| 66                | 8.3239 | 0.9874 | -0.9911 | -1.0302 |
| 67                | 8.1219 | 1.0129 | -1.0013 | -1.0345 |
| 68                | 7.8250 | 1.0037 | -0.9819 | -0.9692 |
| 69                | 7.8334 | 1.0085 | -1.0088 | -0.9828 |
| 70                | 8.0292 | 1.0082 | -0.9822 | -1.1236 |
| 71                | 8.1128 | 0.9926 | -1.0058 | -1.0314 |
| 72                | 8.2170 | 1.0018 | -0.9909 | -1.0800 |
| 73                | 8.1572 | 1.0020 | -0.9912 | -0.9913 |
| 74                | 7.9162 | 1.0058 | -0.9838 | -0.9800 |
| 75                | 8.4242 | 1.0166 | -1.0046 | -1.1802 |
| 76                | 8.2365 | 1.0021 | -1.0086 | -1.0709 |
| 77                | 8.4704 | 1.0057 | -1.0092 | -1.0961 |
| 78                | 8.2634 | 1.0109 | -1.0043 | -1.0300 |
| 79                | 8.0802 | 0.9659 | -0.9874 | -1.0338 |
| 80                | 8.2453 | 1.0183 | -0.9971 | -1.0583 |
| 81                | 8.5512 | 0.9834 | -1.0118 | -1.0790 |
| 82                | 7.8437 | 0.9997 | -0.9978 | -0.9929 |
| 83                | 8.1340 | 0.9818 | -0.9918 | -0.9468 |
| 84                | 8.3957 | 1.0119 | -1.0095 | -1.0589 |
| 85                | 8.1792 | 1.0053 | -0.9906 | -1.0231 |
| 86                | 8.1009 | 0.9933 | -0.9980 | -0.9385 |
| 87                | 8.0357 | 0.9629 | -1.0263 | -0.8889 |
| 88                | 7.9619 | 0.9825 | -0.9979 | -0.9141 |
| 89                | 8.2250 | 0.9839 | -0.9932 | -1.0779 |
| 90                | 7.8981 | 0.9894 | -0.9881 | -0.8814 |
| 91                | 7.7766 | 0.9819 | -1.0157 | -0.9386 |
| 92                | 7.9365 | 1.0125 | -1.0174 | -0.9876 |
| 93                | 8.3197 | 1.0293 | -1.0184 | -1.0195 |
| 94                | 7.6349 | 0.9901 | -0.9642 | -0.9583 |
| 95                | 7.8329 | 1.0115 | -1.0067 | -0.9771 |
| 96                | 7.9165 | 0.9803 | -0.9942 | -1.0080 |
Figure 9: Simulation results of CS-MIO identified Lorenz 96 system using 60k data and under Type 2 noise with four noise magnitudes, namely 0.01, 0.1, 0.4, and 0.8. (a) Hovermøller 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 difference between the ground truth states \( x_j(t) \) and the evolved states \( \hat{x}_j(t) \) using the identified equations by CS-MIO, i.e., \( \Delta x_j(t) = x_j(t) - \hat{x}_j(t) \) for \( j \in [96] \). (b) \( \ell_2 \) error vs time of the trajectories of the recovered Lorenz 96 system from \( t = 0 \) to \( t = 10 \). The exact recovery fails when \( \sigma \) is larger than 0.8.
Table 12: Identified coefficients of Lorenz 96 system using CS-MIO under Type 2 noise with magnitude 0.8: Part A.

| Equation Index $j$ | $F$ | $x_{j+1}x_{j-1}$ | $x_{j-2}x_{j-1}$ | $x_j$ |
|-----------------|-----|-----------------|-----------------|------|
| 1               | 7.0568 | 0.9459           | -0.9658         | -0.6757 |
| 2               | 7.0112 | 0.9483           | -0.9639         | -0.6958 |
| 3               | 7.0591 | 0.9475           | -0.9581         | -0.6963 |
| 4               | 7.0493 | 0.9571           | -0.9597         | -0.6776 |
| 5               | 7.1518 | 0.9481           | -0.9672         | -0.7060 |
| 6               | 7.0869 | 0.9523           | -0.9635         | -0.6799 |
| 7               | 7.1948 | 0.9430           | -0.9682         | -0.6986 |
| 8               | 7.1661 | 0.9491           | -0.9674         | -0.7077 |
| 9               | 6.9831 | 0.9408           | -0.9571         | -0.6745 |
| 10              | 7.0809 | 0.9559           | -0.9623         | -0.7037 |
| 11              | 7.0943 | 0.9516           | -0.9639         | -0.6838 |
| 12              | 7.2745 | 0.9574           | -0.9713         | -0.7263 |
| 13              | 7.0001 | 0.9630           | -0.9728         | -0.6775 |
| 14              | 7.1846 | 0.9497           | -0.9675         | -0.7080 |
| 15              | 7.1597 | 0.9475           | -0.9624         | -0.6962 |
| 16              | 7.0932 | 0.9526           | -0.9670         | -0.6986 |
| 17              | 7.0520 | 0.9541           | -0.9552         | -0.7014 |
| 18              | 6.8848 | 0.9549           | -0.9643         | -0.6501 |
| 19              | 7.3256 | 0.9529           | -0.9710         | -0.7050 |
| 20              | 7.1578 | 0.9634           | -0.9677         | -0.6955 |
| 21              | 6.9812 | 0.9586           | -0.9734         | -0.6890 |
| 22              | 6.9374 | 0.9463           | -0.9665         | -0.6560 |
| 23              | 7.2987 | 0.9527           | -0.9702         | -0.7081 |
| 24              | 7.0471 | 0.9486           | -0.9610         | -0.6813 |
| 25              | 7.0503 | 0.9581           | -0.9643         | -0.7093 |
| 26              | 6.9403 | 0.9514           | -0.9609         | -0.6487 |
| 27              | 7.2672 | 0.9534           | -0.9608         | -0.7060 |
| 28              | 6.9671 | 0.9564           | -0.9652         | -0.6765 |
| 29              | 7.0891 | 0.9571           | -0.9648         | -0.6691 |
| 30              | 7.0759 | 0.9487           | -0.9712         | -0.7038 |
| 31              | 7.0496 | 0.9579           | -0.9665         | -0.6851 |
| 32              | 6.9661 | 0.9421           | -0.9614         | -0.6288 |
| 33              | 7.2776 | 0.9547           | -0.9588         | -0.7411 |
| 34              | 6.8697 | 0.9531           | -0.9710         | -0.6503 |
| 35              | 7.2938 | 0.9561           | -0.9661         | -0.7087 |
| 36              | 6.9934 | 0.9543           | -0.9610         | -0.7026 |
| 37              | 7.1151 | 0.9545           | -0.9641         | -0.6916 |
| 38              | 7.1185 | 0.9554           | -0.9673         | -0.6762 |
| 39              | 6.9873 | 0.9584           | -0.9661         | -0.6900 |
| 40              | 7.0552 | 0.9493           | -0.9635         | -0.6868 |
| 41              | 7.2372 | 0.9544           | -0.9669         | -0.7220 |
| 42              | 6.9862 | 0.9515           | -0.9619         | -0.6670 |
| 43              | 7.0901 | 0.9551           | -0.9586         | -0.6930 |
| 44              | 7.0438 | 0.9531           | -0.9620         | -0.7080 |
| 45              | 7.0553 | 0.9517           | -0.9625         | -0.6809 |
| 46              | 7.0447 | 0.9509           | -0.9646         | -0.6727 |
| 47              | 7.3077 | 0.9685           | -0.9679         | -0.7102 |
| 48              | 7.0477 | 0.9698           | -0.9671         | -0.6916 |
Table 13: Identified coefficients of Lorenz 96 system using CS-MIO under Type 2 noise with magnitude 0.8:
Part B.

| Equation Index | $F$ | $x_{j+1}x_{j-1}$ | $x_{j-2}x_{j-1}$ | $x_j$ |
|----------------|-----|------------------|------------------|-------|
| 49             | 7.1351 | -0.9683 | -0.6815 |
| 50             | 7.0338 | -0.9637 | -0.6808 |
| 51             | 7.0436 | -0.9574 | -0.6857 |
| 52             | 7.0394 | -0.9659 | -0.6886 |
| 53             | 7.1219 | -0.9692 | -0.6887 |
| 54             | 7.1177 | -0.9631 | -0.7365 |
| 55             | 7.0162 | -0.9678 | -0.6666 |
| 56             | 7.2848 | -0.9675 | -0.7262 |
| 57             | 7.0027 | -0.9591 | -0.6802 |
| 58             | 7.0277 | -0.9640 | -0.6903 |
| 59             | 6.8512 | -0.9636 | -0.6585 |
| 60             | 7.2209 | -0.9651 | -0.6831 |
| 61             | 7.0277 | -0.9647 | -0.6740 |
| 62             | 7.0550 | -0.9701 | -0.6907 |
| 63             | 7.0577 | -0.9617 | -0.6875 |
| 64             | 7.1290 | -0.9595 | -0.6763 |
| 65             | 7.0478 | -0.9674 | -0.6722 |
| 66             | 7.1539 | -0.9648 | -0.7014 |
| 67             | 6.9451 | -0.9607 | -0.6592 |
| 68             | 7.0965 | -0.9702 | -0.6837 |
| 69             | 7.0340 | -0.9566 | -0.6943 |
| 70             | 7.0540 | -0.9654 | -0.7106 |
| 71             | 6.9663 | -0.9619 | -0.6307 |
| 72             | 7.2584 | -0.9604 | -0.7156 |
| 73             | 7.0076 | -0.9680 | -0.6768 |
| 74             | 7.1294 | -0.9585 | -0.7171 |
| 75             | 7.1466 | -0.9758 | -0.6850 |
| 76             | 7.1545 | -0.9680 | -0.7093 |
| 77             | 6.9454 | -0.9591 | -0.6868 |
| 78             | 7.0249 | -0.9603 | -0.7016 |
| 79             | 7.0989 | -0.9571 | -0.6800 |
| 80             | 7.1336 | -0.9697 | -0.7055 |
| 81             | 7.1208 | -0.9631 | -0.6860 |
| 82             | 7.1551 | -0.9713 | -0.7370 |
| 83             | 6.9564 | -0.9638 | -0.6756 |
| 84             | 7.1512 | -0.9649 | -0.6969 |
| 85             | 7.0909 | -0.9696 | -0.6817 |
| 86             | 6.9989 | -0.9636 | -0.6907 |
| 87             | 7.0673 | -0.9697 | -0.6880 |
| 88             | 7.0741 | -0.9611 | -0.6648 |
| 89             | 7.1304 | -0.9704 | -0.7008 |
| 90             | 7.0863 | -0.9587 | -0.6852 |
| 91             | 7.1075 | -0.9620 | -0.7066 |
| 92             | 7.0741 | -0.9761 | -0.6834 |
| 93             | 7.2857 | -0.9676 | -0.6904 |
| 94             | 6.8830 | -0.9587 | -0.6762 |
| 95             | 7.1757 | -0.9766 | -0.7105 |
| 96             | 7.1995 | -0.9631 | -0.6822 |
## Appendix C Additional results for the Hopf Normal Form

Table 14: Identified coefficients of Hopf normal form system using CS-MIO under Type 1 noise.

| Noise: $\sigma$ | SNR     | $y$  | $\mu x$ | $x y$ | $x^3$ | $x^2 y$ | $x y^2$ | $y^3$ |
|----------------|----------|------|----------|-------|-------|---------|---------|-------|
| 0.001          | 13758.39 | -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 | -1.0000 |
| 0.1            | 13.758   | -1.0001 | 1.0003 | -0.9992 | -0.9981 | -1.0000 | -1.0000 | -1.0001 |
| 0.3            | 1.529    | -1.0002 | 1.0009 | -0.9975 | -0.9942 | -1.0000 | -1.0000 | -1.0003 |
| 0.5            | 0.550    | -1.0004 | 1.0015 | -0.9959 | -0.9904 | -1.0000 | -1.0000 | -1.0005 |
| 0.7            | 0.281    | -1.0005 | 1.0021 | -0.9942 | -0.9866 | -1.0000 | -1.0000 | -1.0008 |
| 1              | 0.138    | -1.0007 | 1.0031 | -0.9918 | -0.9808 | -1.0000 | -1.0000 | -1.0011 |
| 2              | 0.034    | -1.0014 | 1.0061 | -0.9835 | -0.9616 | -1.0000 | -1.0000 | -1.0022 |
| 3              | 0.015    | -1.0021 | 1.0092 | -0.9753 | -0.9424 | -1.0000 | -1.0000 | -1.0033 |

Table 15: Identified coefficients of Hopf normal form system using CS-MIO under Type 2 noise.

| Noise: $\sigma$ | SNR     | $y$  | $\mu x$ | $x y$ | $x^3$ | $x^2 y$ | $x y^2$ | $y^3$ |
|----------------|----------|------|----------|-------|-------|---------|---------|-------|
| 0.001          | 120306.233 | -0.9951 | 0.9680 | -0.9681 | -0.9680 | -0.9680 | -0.9680 | -0.9680 |
| 0.003          | 13367.359  | -0.9949 | 0.9545 | -0.9542 | -0.9543 | -0.9543 | -0.9543 | -0.9543 |
| 0.005          | 4812.249   | -0.9948 | 0.9289 | -0.9282 | -0.9284 | -0.9284 | -0.9284 | -0.9284 |
| 0.007          | 2455.229   | -0.9946 | 0.8925 | -0.8913 | -0.8915 | -0.8915 | -0.8915 | -0.8915 |
| 0.010          | 1203.062   | -0.9942 | 0.8237 | -0.8215 | -0.8218 | -0.8218 | -0.8218 | -0.8218 |
| 0.013          | 711.871    | -0.9937 | 0.7459 | -0.7425 | -0.7429 | -0.7429 | -0.7429 | -0.7429 |
| 0.015          | 534.694    | -0.9932 | 0.6928 | -0.6887 | -0.6891 | -0.6891 | -0.6891 | -0.6891 |
Appendix D  Additional results for the logistic Map

Table 16: Identified coefficients of the logistic map system using CS-MIO.

| Noise: $\sigma$ | SNR    | $r x_n$   | $r x_n^2$   |
|----------------|--------|-----------|-------------|
| 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     |