

MIT Open Access Articles

Learning sparse nonlinear dynamics via mixed-integer optimization

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Bertsimas, Dimitris and Gurnee, Wes. 2023. "Learning sparse nonlinear dynamics via mixed-integer optimization."

As Published: <https://doi.org/10.1007/s11071-022-08178-9>

Publisher: Springer Netherlands

Persistent URL: <https://hdl.handle.net/1721.1/147108>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of use: Creative Commons Attribution





Learning sparse nonlinear dynamics via mixed-integer optimization

Dimitris Bertsimas · Wes Gurnee

Received: 3 June 2022 / Accepted: 8 December 2022
© The Author(s) 2023

Abstract Discovering governing equations of complex dynamical systems directly from data is a central problem in scientific machine learning. In recent years, the sparse identification of nonlinear dynamics (SINDy) framework, powered by heuristic sparse regression methods, has become a dominant tool for learning parsimonious models. We propose an exact formulation of the SINDy problem using mixed-integer optimization (MIO-SINDy) to solve the sparsity constrained regression problem to provable optimality in seconds. On a large number of canonical ordinary and partial differential equations, we illustrate the dramatic improvement in our approach in accurate model discovery while being more sample efficient, robust to noise, and flexible in accommodating physical constraints.

Keywords Sparse regression · Optimization · System identification

Mathematics Subject Classification 37M10 · 62J05 · 65K05 · 90C11

D. Bertsimas
Sloan School of Management and Operations Research
Center, Massachusetts Institute of Technology, Cambridge,
Massachusetts, USA
e-mail: dbertsim@mit.edu

W. Gurnee (✉)
Operations Research Center, Massachusetts Institute of
Technology, Cambridge, Massachusetts, USA
e-mail: wesg@mit.edu

1 Introduction

Advances in machine learning (ML) combined with the exponential growth of data and computing power are enabling new paradigms of data-driven science and engineering. In particular, emerging techniques for learning dynamic patterns directly from data are poised to revamp fields where data are abundant, but traditional static analysis methods have failed to generate useful models. While accurate dynamical models are important, the ultimate goal is to advance scientific understanding by discovering interpretable models that are as simple as possible, but no simpler.

The modern era of data-driven system discovery started in earnest with the work of [14,56] on symbolic regression. Since then, probabilistic methods [26,46,53,67] and deep neural networks [3,20,40,48,49,65,66] have proved to be effective tools for modeling high-dimensional complex dynamical systems. However, it is the seminal work of [16] on the Sparse Identification of Nonlinear Dynamics (SINDy) framework that serves as the foundation for our approach. SINDy casts system identification as a sparse regression problem over a large set of nonlinear library functions to find the fewest active terms which accurately reconstruct the system dynamics. Such a technique is especially useful for finding highly interpretable models and performs well even with limited training data (in particular, much less than what a neural network would require). Its success has inspired a large number

of extensions and variants tailored for more specific problems [17, 25, 30, 31, 52].

The enabling technology underlying all of these methods is the optimization algorithm that selects and fits the best set of library terms to reconstruct the dynamics. The original SINDy paper [16] used the sequential threshold least squares algorithm which finds a sparse solution by iteratively fitting a least squares regression on the candidate library and removing terms with coefficients below a specified threshold. However, thresholding is problematic for recovering small model coefficients and does not easily allow adding additional structure on the coefficients as might be needed for enforcing an arbitrary physical constraint. These limitations motivated the development of algorithms with different sparse regularizers that could incorporate constraints such as the sparse relaxed regularized regression (SR3) [21, 68] and conditional gradients-based approaches [19]. There is also a long history of greedy algorithms and convex relaxations designed to solve sparse regression and related subset selection problem [47, 61, 62]. For system identification problems, we claim that such relaxations and heuristics are unnecessary, and ultimately inadequate.

Concurrent with the developments in system identification and increase in scientific data, advances in hardware and software have led to over a one trillion times speed up in mixed-integer optimization (MIO) solvers since 1990 [1, 13, 35]. This fact necessitates researchers revisit their preconceptions about the tractability of MIO in machine learning contexts. Indeed, there has been substantial work on viewing the full slate of classical machine learning algorithms under a modern optimization lens [7], often yielding state-of-the-art results with practical computational budgets. Most relevant to system identification is the recent progress on high-dimensional sparse regression which solves the NP-hard feature selection problem exactly [6, 8–11, 29, 62]. In addition to superior performance, these modern formulations inherit the full generality of MIO, empowering domain experts with a rich modeling language to express a vast range of model desiderata as arbitrary linear, quadratic, and semidefinite constraints on both the coefficients and sparsity structure.

The objective of this work is to bridge the gap between the system identification and discrete optimization literatures and demonstrate the effectiveness of learning sparse nonlinear dynamics via MIO-

SINDy. We begin by reviewing MIO for sparse regression and then adapt a formulation utilizing specially ordered sets to the basic SINDy framework and its relevant extensions. We then systematically illustrate the contrast in performance between heuristic and MIO sparse regression (MIOSR) methods on a wide range of canonical dynamical systems, including both ordinary and scalar partial differential equations. Our main contribution is the optimal MIO-SINDy formulation utilizing a MIOSR optimizer for which we establish the following results:

1. *Tractable and provably optimal* MIOSR terminates when the objective of the incumbent solution matches the dual lower bound, that is, when the gap between the objective upper bound and lower bound vanishes, yielding both an optimal solution and a proof of optimality. Despite solving the NP-hard subset selection problem exactly, in Sect. 3.3 we show that the extra computational cost of MIOSR is minimal and scales favorably with additional data and compute.
2. *Sample efficient and noise robust* This theoretical optimality buys practical performance in more challenging statistical regimes. In particular, we study the low data limit in Sect. 3.2 and the high noise setting in Sect. 3.5 where we find MIOSR outperforms heuristic methods, especially in learning the true sparse form of the dynamics.
3. *Customizable* Due to the flexibility of MIO as a modeling framework, MIOSR can be endlessly customized to impose additional structure on the learning problem to further improve sample efficiency, enforce physically realistic models, or incorporate other domain tailored model requirements. We discuss this flexibility and a few relevant extensions in Sect. 2.3 and then demonstrate the benefits of incorporating known physics as constraints in Sect. 3.4.
4. *Consistent interface* We provide an implementation of our algorithm which adheres to the PySINDy interface [33, 58], both computationally and conceptually. Therefore, our algorithm is compatible with other advancements in the SINDy framework (e.g., preprocessing, library construction, outer loop algorithms) and seamlessly integrates into existing tools and workflows.

2 Methods

We begin by reviewing the SINDy problem, its extension to partial differential equations (PDEs), and the weak form of SINDy for noisy data. We then discuss the most appropriate MIO formulations for sparse regression that solve the SINDy problem and some relevant extensions.

2.1 SINDy

The original SINDy framework [16] was designed to recover systems of ordinary differential equations (ODEs) of the form

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

where $x(t) \in \mathbb{R}^d$ is the state of the system at time t and the function $f(x(t))$ encodes the dynamics of the system. Implicitly, $f(\cdot)$ is also assumed to be sparse. This is justified because most physical systems are known to have sparse dynamics when represented in suitable coordinates. Sparsity also acts as a natural and effective regularizer [36] while yielding more interpretable models.

Given n measurements of a system of interest, the three required inputs are a state time matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ where x_{ij} is the state of system variable j at time i , the measured or numerically approximated time derivatives of the state variables $\dot{\mathbf{X}} \in \mathbb{R}^{n \times d}$, and a candidate library of nonlinear functions $\Theta(\mathbf{X}) = [\theta_1(\mathbf{X}), \dots, \theta_L(\mathbf{X})] \in \mathbb{R}^{n \times D}$. As an example, we could consider second-order polynomials $\theta_\ell(\mathbf{X}) = \mathbf{X}^2$ which would yield $d(d-1)/2$ candidate terms of elementwise products for every pair $\mathbf{X}_i \odot \mathbf{X}_j$. In all of our experiments, we will use a library of low-order polynomials, but other natural choices are trigonometric, logarithmic, or exponential functions.

With these ingredients, we seek a solution to

$$\dot{\mathbf{X}} = \Theta(\mathbf{X})\boldsymbol{\Xi} \quad (2)$$

for $\boldsymbol{\Xi} = [\xi^{(1)} \xi^{(2)} \dots \xi^{(d)}] \in \mathbb{R}^{D \times d}$ to learn the dynamics of each state variable $\dot{X}_i = f_i(\mathbf{X}) = \Theta(\mathbf{X})\xi^{(i)}$. Framed as an optimization problem, the standard objective is to

$$\min_{\boldsymbol{\Xi}} \|\dot{\mathbf{X}} - \Theta(\mathbf{X})\boldsymbol{\Xi}\|^2 + \lambda R(\boldsymbol{\Xi}) \quad (3)$$

where $R(\cdot)$ is a sparsity promoting regularization function which may also include an l_2 ridge regularization

term to improve the conditioning and add robustness [5].

Given the rapid growth of the number of library functions with regard to the input data dimensionality (e.g., $D = O(d^p)$ for an order p polynomial library), SINDy is best suited for analysis of low-dimensional data sets. Therefore, to learn the dynamics of high-dimensional systems, a dimensionality reduction technique such as proper orthogonal decomposition (POD) is first applied to the data [4], and then, SINDy is applied to the reduced data space. SINDy also makes the implicit assumptions that the data contain the relevant governing variables and are represented in coordinates which allow for sparsely representing the dynamics as the sum of only a few elementary functions, and that the library contains these elementary functions.

SINDy for PDEs This core framework can be further extended to the automatic discovery of PDEs by including partial derivatives in the candidate library [52, 54]. Concretely, spatiotemporal data of m spatial locations measured over n time slices are arranged into a length mn column vector \mathbf{U} . Then, a candidate library $\Theta(\mathbf{U}) \in \mathbb{R}^{mn \times D}$ is constructed as before except here we consider functions of both the system state and system spatial derivatives (which have to be numerically approximated). That is, in addition to library functions like \mathbf{U}^2 , we also might include $\mathbf{U}\mathbf{U}_x$ and potentially higher-order derivatives like \mathbf{U}_{xxx} . Finally, as before, we seek coefficients $\boldsymbol{\Xi}$ which accurately reconstruct the temporal dynamics $\mathbf{U}_t = \Theta(\mathbf{U})\boldsymbol{\Xi}$.

Weak form One core drawback with SINDy, especially when applied to PDEs or noisy data, is the need to numerically estimate derivatives because numerical differentiation compounds any noise present in the underlying measurement data. When differentiating multiple times, as is necessary for higher-order PDEs, the estimates can become unusable, even while using more robust differentiation techniques like smoothed finite difference or polynomial interpolation.

This drawback motivates the weak form of SINDy [45, 50, 55] (which also generalizes to the PDE case [27, 44]), where both sides of Equation 2 are integrated over a random collection of K temporal subdomains (spatiotemporal for PDEs). That is, for a random subdomain Ω_k , a candidate library function θ_i , and a weight vector \mathbf{w} , we compute

$$q_i^k = \int_{\Omega_k} \mathbf{w}^T \theta_i d\Omega \quad (4)$$

for every library term and subdomain. Each of the elements is then organized into a data matrix $\mathbf{Q} \in \mathbb{R}^{K \times D}$. By also integrating the left hand side of Equation 1 over the same subdomains, we get a linear system $\mathbf{q}_0 = \mathbf{Q}\boldsymbol{\xi}$ amenable to sparse regression without needing to differentiate noisy data.

2.2 Mixed-integer sparse regression

The seminal paper on best subset selection using MIO [8] proposed two primal formulations for sparse regression. Both of these formulations use binary variables to encode the support of the coefficients, one using big-M constraints, and the other using type-1 specially ordered sets (SOS-1). Because the system identification problem is coordinate separable, we can fit each dimension independently, resulting in d smaller subproblems which can be solved directly using these techniques.

For system dimension j , with user-specified target sparsity k_j , the big-M formulation with ridge regularization for the SINDy problem is

$$\min_{\xi, z} \|\dot{\mathbf{X}}_j - \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\xi}\|_2^2 + \lambda \|\boldsymbol{\xi}\|_2^2 \quad (5)$$

$$\text{s.t. } M_i^\ell z_i \leq \xi_i \leq M_i^U z_i \quad i = 1, \dots, D \quad (6)$$

$$\sum_{i=1}^D z_i \leq k_j \quad (7)$$

$$\xi_i \in \mathbb{R}, \quad z_i \in \{0, 1\} \quad i = 1, \dots, D \quad (8)$$

where M_i^ℓ , M_i^U are lower and upper bounds on the coefficients. This formulation is solved for each $j \in [1, d]$, where we simply stack the coefficient vectors to recover the full system dynamics $\boldsymbol{\xi} = [\xi^{(1)} \ \xi^{(2)} \ \dots \ \xi^{(d)}]$.

While the theory and practice of solving MIO problems are deep [12], the basic solution technique relies on the linear programming (LP)-based branch-and-bound algorithm to avoid performing the full combinatorial search. The hard part of MIO problems is the discrete variables which make the problem non-convex and NP-hard. Hence, branch and bound relies on solving the polynomial-time linear relaxation of the problem by allowing integer variables to take continuous values, to obtain bounds on the optimal integral solution. Branch and bound maintains a tree of solutions to the LP relaxation where, at each branch of the tree, an integer variable is fixed to an integer value,

while maintaining global upper and lower bounds on the optimal objective value of an integral solution. The algorithm continues expanding the tree, by branching on integer variables with fractional LP optimal values in promising partial solutions, while pruning other branches outside the solution bounds, until the lower and upper bounds converge, yielding an optimal integral solution. We rely on modern optimization solvers such as Gurobi [28] or CPLEX [23] to both solve the problem and present this certificate of optimality.

The effectiveness of big-M modeling relies on the tightness of coefficient bounds as otherwise the linear relaxations are too weak to efficiently prune the branch-and-bound tree. [8] derive a number of ways to obtain such bounds; however, these approaches can add significant overhead to the solution times and don't generalize well in the presence of arbitrary constraints. This motivates a nonlinear approach which circumvents the need to calculate these $2D$ different bounds. [8] also proposed adding the cardinality constraint via type-1 specially ordered sets (SOS-1) [12]. An SOS-1 constraint on a set of variables enforces that no more than one variable within the set is nonzero enabling branching on multiple variables for each branch of the branch-and-bound tree. In this case,

$$(1 - z_i)\xi_i = 0 \iff \{\xi_i, 1 - z_i\} : \text{SOS-1}$$

correctly captures the support of ξ . By replacing the support constraint, we get

$$\min_{\xi, z} \|\dot{\mathbf{X}}_j - \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\xi}\|_2^2 + \lambda \|\boldsymbol{\xi}\|_2^2 \quad (9)$$

$$\text{s.t. } \{\xi_i, 1 - z_i\} : \text{SOS-1} \quad i = 1, \dots, D \quad (10)$$

$$\sum_{i=1}^D z_i \leq k_j \quad (11)$$

$$\xi_i \in \mathbb{R}, \quad z_i \in \{0, 1\} \quad i = 1, \dots, D. \quad (12)$$

More explicitly, the main objective term is

$$\|\dot{\mathbf{X}}_j - \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\xi}\|_2^2 \quad (13)$$

$$= \boldsymbol{\xi}^T \boldsymbol{\Theta}(\mathbf{X})^T \boldsymbol{\Theta}(\mathbf{X}) \boldsymbol{\xi} - 2\langle \boldsymbol{\Theta}(\mathbf{X})^T \dot{\mathbf{X}}_j, \boldsymbol{\xi} \rangle + \dot{\mathbf{X}}_j^T \dot{\mathbf{X}}_j. \quad (14)$$

If we remove the constant term $\dot{X}_j^T \dot{X}_j$ and add the regularization term, we get our final objective

$$\begin{aligned} & \xi^T \Theta(X)^T \Theta(X) \xi - 2\langle \Theta(X)^T \dot{X}_j, \xi \rangle + \lambda \xi^T \xi \quad (15) \\ & = \xi^T \left(\Theta(X)^T \Theta(X) + \lambda I \right) \xi - 2\langle \Theta(X)^T \dot{X}_j, \xi \rangle. \quad (16) \end{aligned}$$

For a problem with n temporal observations and library size D , our final formulation has D continuous variables, D binary variables, D corresponding SOS-1 constraints, and one knapsack constraint. The objective has $\frac{D(D+1)}{2}$ quadratic terms and D linear terms. Notably, the formulation size is independent of n which yields very favorable scaling properties as we discuss in Sect. 3.3.

A question that naturally arises is whether to enforce sparsity as a hard constraint or promote sparsity as an objective penalty. While it may be tempting to use a penalty term and let the model balance sparsity, it is known that constrained problems enjoy more favorable statistical properties [57]. In particular, while an optimal solution to the sparsity regularized problem is always obtainable by the constrained problem, the converse is not true in general (see [34] Sect. 2.2). This is especially true when the data matrix exhibits high multicollinearity which is common in system discovery because the library terms are usually strongly correlated.

Finally, we note that we adopt the SOS-1 formulation because we found it to be the most numerically stable and most flexible in including other potential model desiderata (e.g., satisfaction of physical constraints). However, it is not the most scalable. Modern general-purpose optimization solvers can only handle sparse regression problems with up to a few thousand SOS-1 constraints. This is in stark contrast with very recent tailored sparse regression solution techniques such as the outer approximation method [9], coordinate descent based branch and bound [29], and the backbone method [6] which can scale to the high-dimensional regime with dimension $O(10^7)$. Given that even a six-dimensional system with a fifth-order polynomial library is only of dimension 462, the more stable and flexible general-purpose solvers are preferable for our circumstances.

2.3 Extensions

In many physical systems where something is known about the underlying physics, we can incorporate this knowledge as constraints on the model coefficients [21, 32, 37]. However, these constraints generally apply to the system as a whole (e.g., conservation of energy), so it is no longer possible to fit one coordinate at a time. Therefore, we fit all coordinates jointly using objective

$$\min_{\xi} \left\| \begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \\ \vdots \\ \dot{X}_d \end{bmatrix} - \begin{bmatrix} \Theta(X) & 0 & \dots & 0 \\ 0 & \Theta(X) & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \dots & \Theta(X) \end{bmatrix} \begin{bmatrix} \xi^{(1)} \\ \xi^{(2)} \\ \vdots \\ \xi^{(d)} \end{bmatrix} \right\|_2^2 \quad (17)$$

$$= \min_{\xi} \sum_{i=1}^d \xi^{(i)T} \Theta(X)^T \Theta(X) \xi^{(i)} - 2\langle \Theta(X)^T \dot{X}_i, \xi^{(i)} \rangle. \quad (18)$$

Now in addition to the sparsity constraint, we can add arbitrary constraints $A\xi \leq b$ where ξ is the vectorized coefficient matrix of length Dd . Of course, because we inherit the full generality of MIO, these constraints can be anything that modern optimization solvers can handle (e.g., linear, quadratic, semidefinite, equality, inequality).

While such a formulation increases the dimension of the regression problem by a factor of d (which is potentially quite costly, both in terms of computational time and sample efficiency), it is sometimes more natural and offers several additional benefits. The first is based on the fact that the coordinates used in SINDy often represent the spatial modes of a high-dimensional discretized PDE simulation computed using a dimensionality reduction technique like proper orthogonal decomposition [22]. Each of these spatial modes has an associated energy λ_i designated by their singular values. Consequently, the quality of the high-dimensional reconstruction does not depend uniformly on the accuracy of each of the individual spatial modes, but in proportion to their energies. In this setting, we can weight each inner term in the sum of the objective function (17) by the energy of the respective spatial mode to recover the terms which maximize the quality of the full reconstruction, rather than the average dimension-wise reconstruction. Another advantage is instead of having to run parameter tuning on different values of

k for each system variable, we can specify and cross-validate one global value of the desired sparsity, and the model will automatically determine the correct level of sparsity per dimension.

While not pursued here, there are many extensions that could be appropriate for more specific circumstances. For instance, it is possible to add lower and upper bounds on the magnitudes of coefficients, add more sophisticated conditional logic on the relationship between nonzero coefficients, put controls on the level of multicollinearity [9], require coefficients be statistically significant [9], automatically prune outliers [21,60], and much more. In short, as such a general framework, MIO-SINDy fully empowers the researcher to express a vast range of model desiderata and impose additional structure to aid the learning process.

3 Results

We benchmark our approach on nine canonical dynamical systems across a wide variety of statistical regimes. Our analysis focuses on attributes that most differentiate SINDy from alternative techniques and then illustrates how using optimal methods furthers these advantages. In particular, we study sample efficiency, robustness, constraint enforcement, and computational efficiency where our evaluation focuses on identifying correct sparse models.

All of our experiments are built off of the open source PySINDy library [33,58]. We use a shared university cluster with heterogeneous hardware where individual trials are confined to one CPU core with sufficient RAM. Our implementation of MIO sparse regression utilizes Gurobi 9.5.0 [28] to optimize and prove optimality. We make all of our code, data, and results publicly available at our Github repository https://github.com/wesg52/sindy_mio_paper.

3.1 Experimental overview

Most of our experiments follow the same high level structure. We vary a quantity of interest (e.g., data length, data noise) for 50 random initial conditions, each with additive Gaussian noise scaled to be a certain percentage of the l_2 norm of the training data. Then, for each sampled trajectory, we split the data into a training

and a validation segment, using the validation segment to select the hyperparameters (see Sect. 3.1.1) of each of the baseline algorithms (Sect. 3.1.2). We follow the standard practice of unbiasing the final model by refitting an unregularized least squares regression on the selected coefficients. This final model is then evaluated on a suite of metrics (Sect. 3.1.3), with a focus toward identifying the correct coefficient support.

While the specifics of our results are somewhat sensitive to the details of our experimental procedure, we take steps to ensure our conclusions are robust to such design choices. For instance, we use random initial conditions, whereas many papers in the SINDy literature report results using a fixed initial condition. While this aids reproducibility, we found performance to be sensitive to initial conditions for many systems, especially in the low data limit. Additionally, for each experiment we test our approach on multiple systems, each of which raises different qualitative behavior, to better understand the factors which differentiate performance.

3.1.1 Model selection

A critical component of learning parsimonious models is in choosing hyperparameters that appropriately tradeoff model fit with sparsity, since adding more degrees of freedom monotonically decreases in-sample error. We strike this balance by selecting parameters which minimize the Akaike information criterion (AIC) metric [2,42].

In our setting of sparse regression, for a learned model $\hat{\mathbf{z}}$, the corrected AIC is given by

$$AIC_c = m \ln(RSS/m) + 2k + \frac{2(k+1)(k+2)}{m-k-2} \quad (19)$$

where RSS is the residual sum of squared errors $\sum_{i=1}^n \sum_{j=1}^d (\dot{\mathbf{X}} - \boldsymbol{\Theta}(\mathbf{X})\hat{\mathbf{z}})_{ij}^2$, $m = n \times d$ is the total number of measurements, k is the sparsity of the solution, and the last term is the correction for finite samples.

Unless otherwise noted, for every trial of every experiment discussed below, we run the following model selection procedure. Split the sampled trajectory into a train and a validation interval, typically the first 2/3 and the last 1/3, respectively. For each algorithm and choice of hyperparameters, train on the training split and compute the AIC_c with respect to the validation data. The final model is the one which minimizes the AIC_c metric on the validation data. Note, for algo-

gorithms which fit each dimension separately, we compute the AIC_c dimensionwise and combine the best coefficients per dimension to create the final model.

3.1.2 Algorithms

Given our focus on accurate support recovery, and for sake of consistent comparison, we restrict our baselines to other l_0 regularized or constrained sparse regression algorithms. This notably excludes l_1 regularized sparse regression methods like LASSO and its variants [19, 61], probabilistic methods, and deep learning-based approaches which are less suitable for accurate variable selection [9, 21]. Specifically, we compare our MIO-based approach (MIOSR) to four common optimizers in the SINDy literature: sequential threshold least squares (STLSQ) [16], sparse relaxed regularized regression (SR3) [21], stepwise sparse regression (SSR) [15], and ensembling using STLSQ (E-STLSQ) [25].

For every experiment trial, we perform a grid search over the relevant hyperparameters of each algorithm to find the set which minimize the AIC_c as described above. Each algorithm has a hyperparameter corresponding to regularization strength and sparsity promotion. MIOSR, STLSQ, SSR, and E-STLSQ all have an explicit ridge regression penalty while SR3 has a relaxation parameter ν . For MIOSR and SSR, we tune the sparsity of each dimension, where we control the can control k exactly. For (E-)STLSQ and SSR, we tune the threshold parameter which acts as a proxy for the true sparsity. Parameter ranges for each algorithm are included in appendix.

For SSR, we use the greedy criterion of removing the smallest magnitude coefficient at each iteration. For SR3, we run until convergence up to a max 10000 iterations. For MIOSR, we set a timeout of 30 sec. per dimension to prove convergence. For E-STLSQ, we use robust bagging (bragging) where we randomly sample time slices with replacement to train 50 different models, and then use the median of the coefficients as the final model.

3.1.3 Evaluation metrics

To evaluate each algorithm, we focus on three common metrics: true positivity rate (TPR), normalized coefficient error (NCE), and root mean squared error (RMSE) of the estimated dynamics. Throughout, let Ξ be the

true dynamics of the system and $\hat{\Xi}$ be the estimated dynamics.

The true positivity rate measures the ability to identify the correct nonzero terms of the dynamics and importantly to not select superfluous terms. Specifically, the true positivity rate is the ratio of intersection over union of nonzero coefficients

$$TPR = \frac{|\{i : \Xi_i \neq 0\} \cap \{i : \hat{\Xi}_i \neq 0\}|}{|\{i : \Xi_i \neq 0\} \cup \{i : \hat{\Xi}_i \neq 0\}|}, \quad (20)$$

or equivalently, the ratio of true coefficients identified to the combined number of true coefficients, false zero coefficients, and false nonzero coefficients identified.

The normalized coefficient error simply measures the normalized Euclidean distance between the true coefficients and the learned coefficients

$$NCE = \frac{\|\Xi - \hat{\Xi}\|_2}{\|\Xi\|_2}. \quad (21)$$

This metric is less punitive than the true positivity rate because it mainly captures the difference in large coefficients and minimally penalizes small nonzero terms.

Finally, to contextualize how the coefficient error actually impacts the quality of the recovered model, we also report the root mean squared error of the estimated derivatives on a clean test set. That is, for a testing trajectory of length n we calculate

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{nd} \sum_{i=1}^n \sum_{j=1}^d (\Theta(X_{test})\Xi - \Theta(X_{test})\hat{\Xi})_{ij}^2}. \end{aligned} \quad (22)$$

To test that the model truly generalizes, we independently sample 10 initial conditions, each run for 10 sec., and take the average RMSE across each of these trajectories. Due to the chaotic nature of the systems we study, we only compare the derivatives and not the trajectories of the forward models, since initial errors will rapidly compound regardless of the accuracy of the underlying model.

In the results that follow, we will typically report the log RMSE and log l_2 coefficient errors, averaged over 50 trials. When calculating the mean and standard errors, we do so on the log of these statistics. This avoids one outlier from dragging up the mean by many orders of magnitude and makes the standard errors symmetric in log space.

3.2 Sample efficiency

A key advantage of SINDy over deep learning techniques is the reduced data requirements to learn high-quality sparse models. To test how MIOSR extends this advantage, we study the effect of varying the length of the training trajectory on model recovery for three canonical dynamical systems studied in the SINDy literature: the Lorenz model [38], the Hopf system [43], and a triadic magnetohydrodynamical (MHD) model [18].

The Lorenz model is the classic example of a chaotic system and is given by

$$\dot{x} = \sigma(y - x) \quad (23a)$$

$$\dot{y} = x(\rho - z) - y \quad (23b)$$

$$\dot{z} = xy - \beta z \quad (23c)$$

where we use the standard parameters $\sigma = 10$, $\beta = 8/3$, $\rho = 28$. Another canonical system in the study of nonlinear dynamics is the Hopf system given by

$$\dot{x} = \mu x + \omega y - Ax(x^2 + y^2) \quad (24a)$$

$$\dot{y} = -\omega x + \mu y - Ay(x^2 + y^2) \quad (24b)$$

where we set $\mu = -0.05$, $\omega = A = 1$. Finally, we consider a simplified plasma model of a joint velocity and magnetic field with 0 fluid viscosity or resistivity

$$\dot{V}_1 = 4(V_2 V_3 - B_2 B_3) \quad (25a)$$

$$\dot{V}_2 = -7(V_1 V_3 - B_1 B_3) \quad (25b)$$

$$\dot{V}_3 = 3(V_1 V_2 - B_1 B_2) \quad (25c)$$

$$\dot{B}_1 = 2(B_3 V_2 - V_3 B_2) \quad (25d)$$

$$\dot{B}_2 = 5(V_3 B_1 - B_3 V_1) \quad (25e)$$

$$\dot{B}_3 = 9(V_1 B_2 - B_1 V_2). \quad (25f)$$

Figure 1 depicts our results for each combination of algorithm and system for varying lengths of training data. For each system, we sample trajectories with 0.002 second time granularity with 0.2% added Gaussian noise. In anticipating the assumption that exact optimization methods are not scalable, we use oversized libraries: 5th-order polynomials for the Lorenz and Hopf systems and 3rd-order polynomials for MHD.

This yields libraries with dimension 56, 21, and 84, respectively.

The high-level conclusion is that MIOSR consistently outperforms all other methods by finding more accurate models with less data and with less variance in the quality of the fit. While the gap is more muted in the Lorenz case, the stark differences in the Hopf and MHD systems surface two distinct scenarios where heuristics can fail: the presence of small coefficients and large libraries. For Hopf, with bifurcation parameter $\mu = -0.05$, thresholding techniques cannot select such a small coefficient in the presence of even modest noise. Indeed, the original SINDy paper used multiple independent trajectories to learn the Hopf dynamics, because the system quickly converges to a fixed point. In the low data limit for MHD, a larger 6-dimensional system, there are many combinations of the large library of terms which fit the small training set well. With so many degrees of freedom, iterative methods break down by taking incorrect intermediate steps, but by taking a global view, MIOSR can identify the true model.

In this low data regime, some baselines completely fail, in particular SR3. This is perhaps unsurprising because SR3 fits all coordinates jointly and therefore is solving a more difficult, higher-dimensional optimization problem. While sometimes the baselines methods achieve comparable test RMSE, they often do so by overfitting as evidenced by the low true positivity rate. This largely defeats the purpose of SINDy in discovering robust, interpretable, and scientifically illuminating models.

Many of the aforementioned difficulties are partially ameliorated by using a smaller library. In appendix, we perform the same experiment with “tight” libraries, those which don’t include higher-order polynomials than are necessary to express the system (Fig. 6). While MIOSR maintains a clear edge, the difference is less striking. Of course for novel systems, this information is not available a priori, and therefore, this represents the most idealized scenario.

3.3 Computational efficiency

Nearly every SINDy paper contains a sentence that justifies the need for sparse regression heuristics by claiming that the feature selection problem is computationally intractable due to the combinatorial nature.

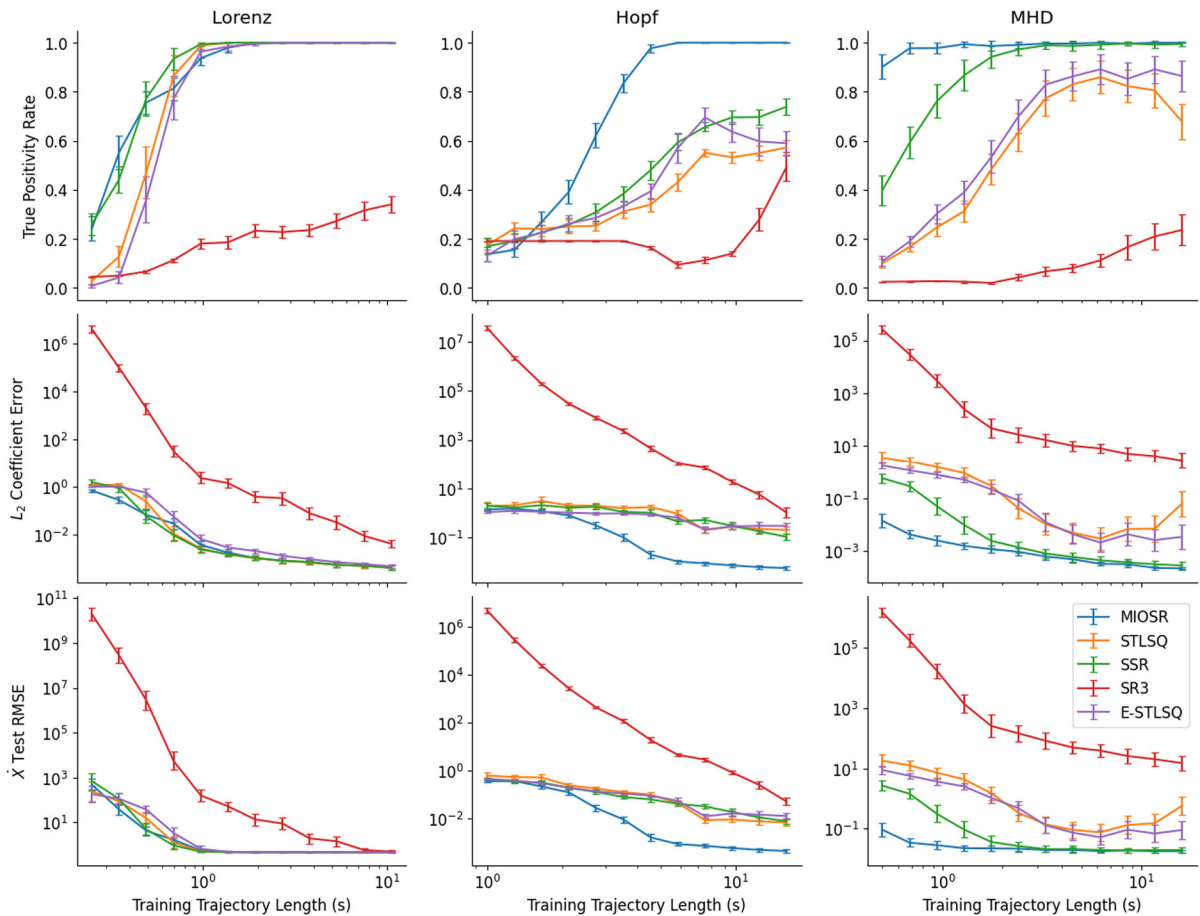


Fig. 1 Performance comparison of sparse regression algorithms for the differential form of SINDy under varying amounts of training data for three different canonical systems: Lorenz, Hopf,

and MHD. Results are averaged over 50 trials with added Gaussian noise of 0.2%

To directly dispel this claim, we compare the wallclock computation time of each of different algorithms for varying library sizes and amounts of training data (see Fig. 2). Similar to the previous experiment, for each system, library size, and amount of data, we train each algorithm on 50 random trajectories with 0.2% additive Gaussian noise with 0.002 sample frequency. We precompute the derivatives and library, so the reported times corresponds to just the regression time, not the whole SINDy pipeline. Unlike the previous experiment, we are not performing hyperparameter tuning and instead use appropriate defaults learned above.

Several points deserve elaboration. Perhaps most surprising to those unfamiliar with MIO-based machine learning is that MIOSR is often faster as the amount

of data increases, sometimes significantly [11]. This is partially due to the fact that the final optimization problem has no dependence on n , as we only require a one time $n \times D$ matrix multiplication to initially construct the objective value coefficients. On the other hand, for small n , there are more ways to fit the data, so the bounds in the branch-and-bound tree are weaker, necessitating more node exploration. Combining these results with those from Sect. 3.2, we conclude that either MIOSR takes a comparable amount of time while achieving the same accuracy, or it takes longer, but the extra computational cost buys extra statistical performance. That is, regardless of data size, the cost of MIOSR is justified.

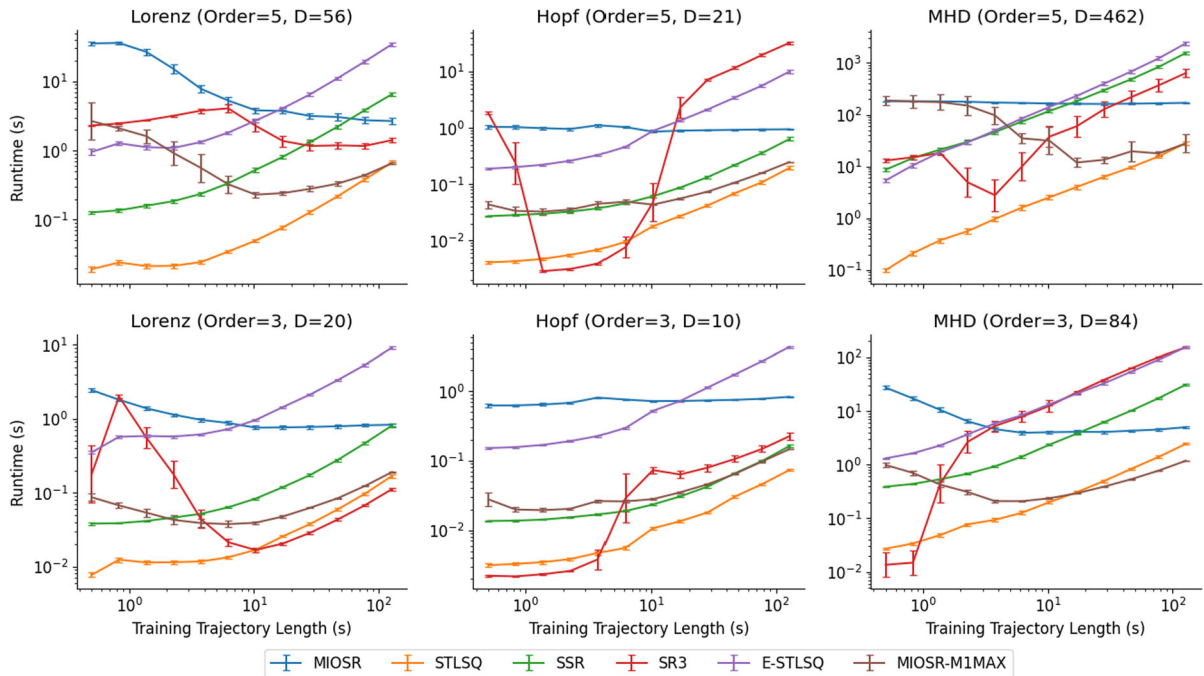


Fig. 2 Comparison of sparse regression algorithm computational efficiency for the differential form of SINDy under varying amounts of training data for three different canonical systems: Lorenz, Hopf, and MHD. The top row uses a fifth-order poly-

nomial library for each of the three systems while the bottom row uses a third-order polynomial library. Results are averaged over 50 trials with added Gaussian noise of 0.2%

Another critical point is the dramatic speed up associated with utilizing more powerful hardware and parallelization. When run on a high-end laptop (2021 MacBook Pro with M1 Max chip), as opposed to a single core of a cluster node, MIOSR can be up to 100x faster, while all other methods only improve by a few percent (see Fig. 7). This is because MIO is a highly parallelizable technology, as multiple threads can explore, expand, and prune different branches of the branch-and-bound tree in parallel. Therefore, one can simply allocate additional compute to achieve increasing levels of performance, and the time gap between optimal methods and heuristics will continue to close as more powerful MIO software and parallel hardware emerge in the future.

In addition to hardware, there are several other factors which understate how efficient MIOSR can be relative to the results reported in Fig. 2. The first is that MIO requires less hyperparameter tuning than other methods because we are directly tuning the sparsity and not a proxy threshold. Second, for especially difficult or large problems, one can use warm starts from heuristic methods to get good initial solutions or reuse

solutions and models from previous steps in the hyperparameter search (which also avoids reconstructing the full optimization model). Finally, much of the computational effort is dedicated toward proving optimality by improving the dual lower bound [11]. Therefore, one could set a short timeout on the solver to get what is likely an optimal solution, but give up on the optimality guarantee if so desired.

While not studied in detail here, it is important to note that runtime also has a dependence on the sparsity parameter k because the number of possible models scales exponentially as $\binom{D}{k}$. However, given the dynamics are assumed to be sparse (e.g., $k < 10$), this scaling limitation is less relevant. We refer the reader to other works [8, 9] on general MIOSR methods for more extensive runtime benchmarking.

3.4 Physical constraints

A central goal within scientific machine learning is to incorporate existing physical knowledge into the models, both to aid the learning process and to ensure that

physically plausible models are learned. To illustrate the improved capability of MIOSR in service of this goal, we replicate the experiment performed by [21] on the two-dimensional Duffing system.

To briefly describe their setup, the 2D Duffing system is both a Hamiltonian system and a gradient system. These properties induce constraints on the coefficients because each individual governing equation must be a partial derivative of a Hamiltonian or potential function. The full system is described by

$$\dot{x} = X \quad (26a)$$

$$\dot{y} = Y \quad (26b)$$

$$\dot{X} = -\frac{\partial}{\partial x} V(x, y) \quad (26c)$$

$$\dot{Y} = -\frac{\partial}{\partial y} V(x, y) \quad (26d)$$

where x, y give the spatial position, X, Y give the momentum, and the potential function is

$$V(x, y) = -\frac{\omega}{2}(x^2 + y^2) + \frac{\alpha}{4}(x^2 + y^2)^2. \quad (27)$$

We use SINDy to just fit the spatial coordinates and set $\omega = -2$ and $\alpha = 0.1$. We refer the interested reader to [21] for the detailed derivation of the constraints, but for our purposes, the relevant fact is simply that the potential function imposes a set of equality constraints on Ξ . We can then use a vectorized representation of the coefficients $\bar{\xi}$ and add $A\bar{\xi} = b$ to the MIO model where A, b are based on the partial derivatives of $V(x, y)$. Because these constraints extend between dimensions, we use the joint formulation (17) to fit x and y simultaneously.

As in previous experiments, we sample 50 independent trajectories with random initial conditions, for every combination of training duration and noise depicted in Fig. 3. For every trajectory, we run MIOSR with and without constraints, as well as SR3 with and without constraints, as it is the only baseline which naturally accommodates constraints. To stay consistent with [21], we use a third-order polynomial library with a sample rate of 0.01 sec. Additionally, we do not unbias the coefficients after feature selection and report the average constraint violation $\frac{1}{c}\|A\hat{\xi} - b\|_1$ where c is the number of constraints.

The immediate takeaways from Fig. 3 are that adding constraints help both algorithms, especially with limited data, but that even without constraints,

MIOSR is more accurate. This is in slight contrast to the findings of [21] where “the constrained and unconstrained models have nearly identical $[R^2]$ scores at all noise levels.” However, this is because we study a more difficult statistical regime, one with less data and with coefficients of different magnitudes. In particular, they used training sets that included 20 independent trajectories; hence, the constraints did not add additional information.

Another drawback of SR3 is that it enforces the constraints on a set of relaxed coefficients; hence, the constraints are not strict and there exist nontrivial violations. This occurs most frequently in training regimes with less data and more noise. Unfortunately, these are precisely the regimes where constraints are most useful. MIOSR, in contrast, always satisfies constraints up to solver numerical precision, which can be adjusted or relaxed as desired. Finally, we observe that adding constraints does not add to the solution time of MIOSR, so the runtimes are comparable to those reported in Sect. 3.3, while constraining SR3 increases the runtime by about 30%.

3.5 Robustness

For the remainder of our experiments, we study system recovery under substantial noise, and therefore, we use the weak form of SINDy, where the data matrix is given by Eq. 4. We first study robust recovery of three canonical ODEs: the Van der Pol oscillator [64], Lotka–Volterra equations [39], and the Rössler system [51].

The Van der Pol system is given by

$$\dot{x} = y \quad (28a)$$

$$\dot{y} = \mu(1 - x^2)y - x \quad (28b)$$

where we use $\mu = 3$. The Lotka–Volterra equations, sometimes also known as the predator–prey equations given their origin in modeling wildlife populations, are given by

$$\dot{x} = p_1x - p_2xy \quad (29a)$$

$$\dot{y} = p_2xy - 2p_1y \quad (29b)$$

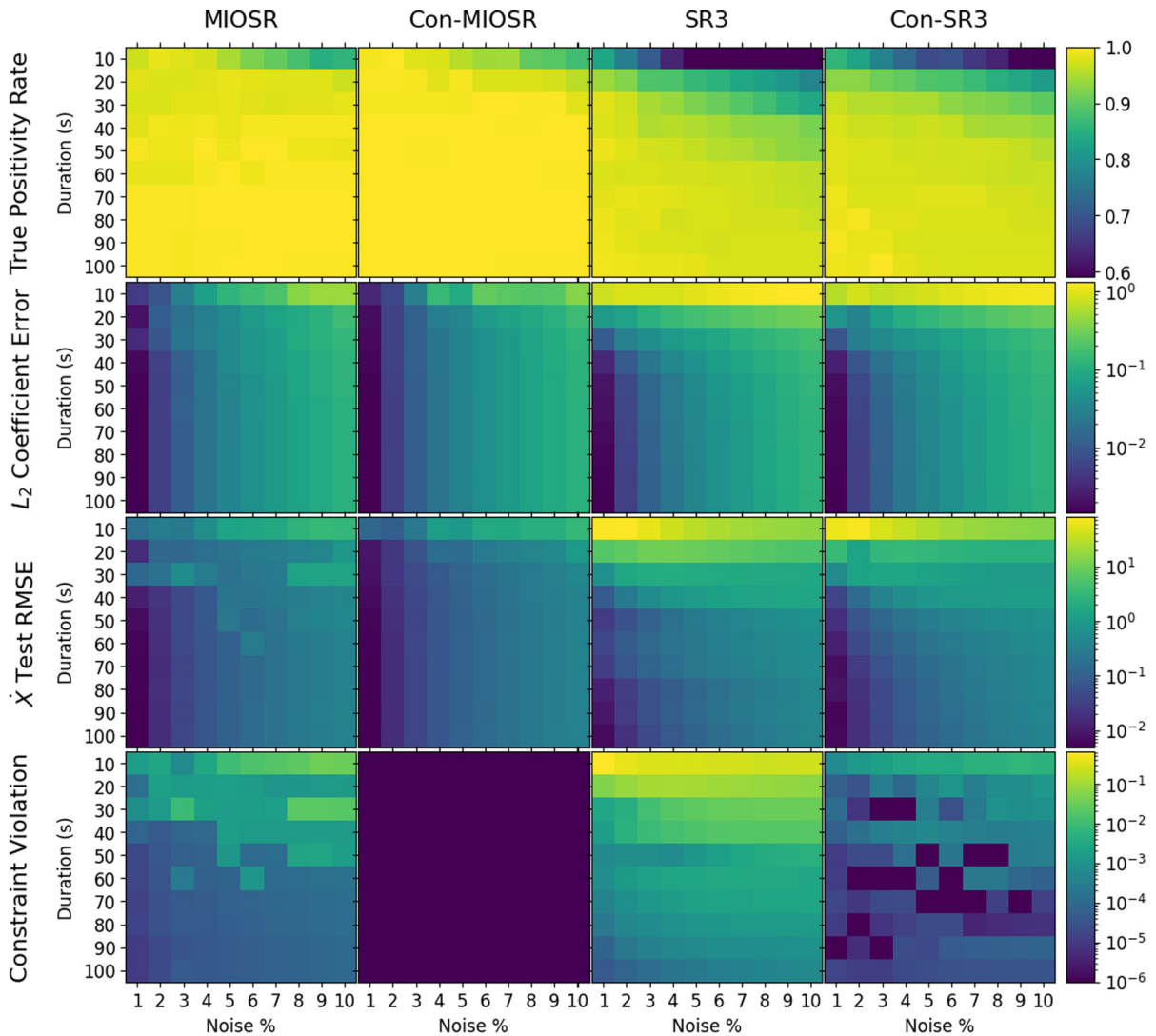


Fig. 3 Performance comparison of constrained versus unconstrained sparse regression for the differential form of SINDy under varying amounts of training data and noise for the 2D

Duffing system. Results are averaged over 50 trials with random initial conditions

where we set $p_1 = 1$ and $p_2 = 10$. Finally, the Rössler system is

$$\dot{x} = -y - z \quad (30a)$$

$$\dot{y} = x + ay \quad (30b)$$

$$\dot{z} = b - cz + xz \quad (30c)$$

where we have $a = b = 0.2$ and $c = 5.7$.

For all three systems, we use a third-order polynomial library with 50 sec. of training data with time inter-

vals of 0.002 sec. (i.e., 25000 total time steps). Our weak libraries are composed of 2400 spatial domains each with 400 points per domain. To validate weak models on noisy data, one can either use the weak form of the validation set or try to differentiate the validation data with more aggressive smoothing. We observe that the weak form of the validation data yields a less reliable tuning signal, partially because the weak form uses randomized domains and leads to less sparse models. However, the numerical derivative also becomes

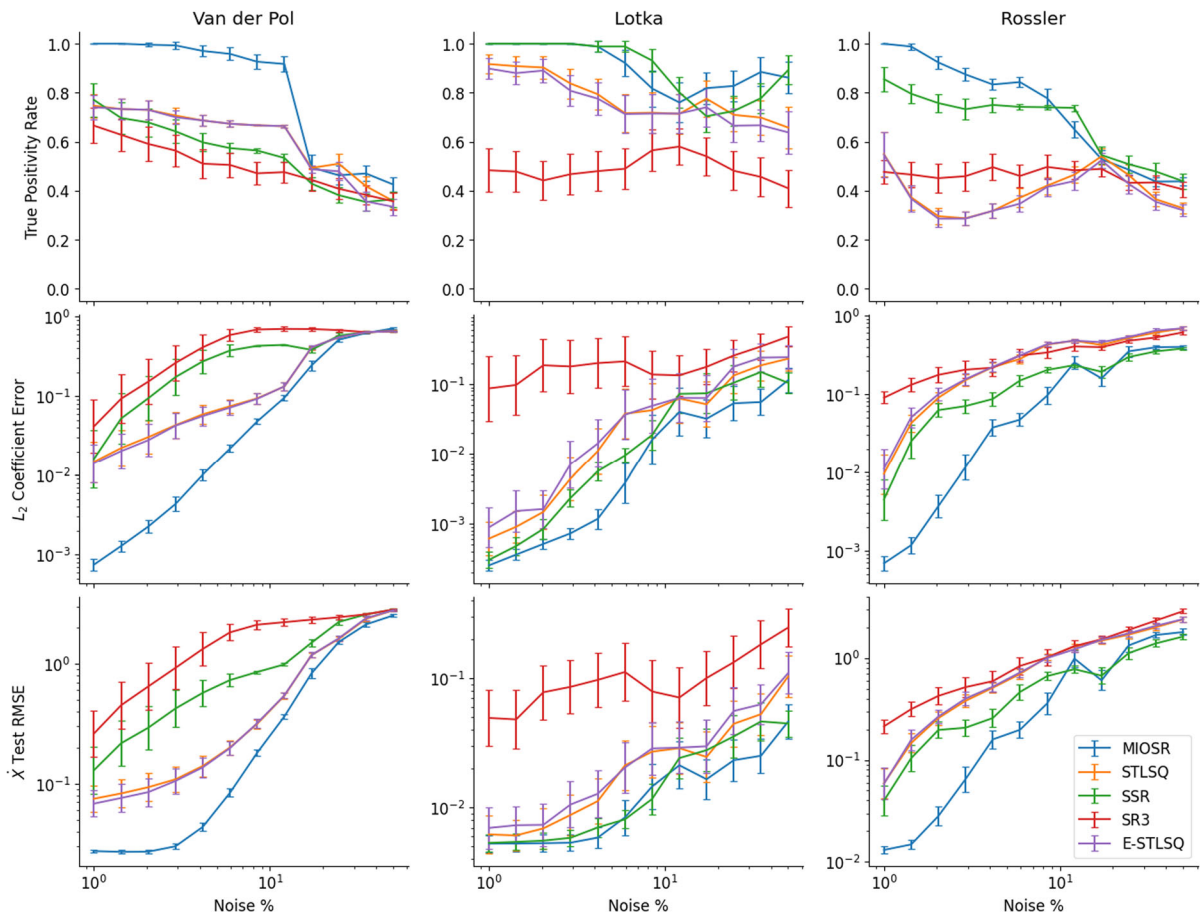


Fig. 4 Performance comparison of sparse regression algorithms for the integral form of SINDy under varying amounts of added Gaussian noise for three different canonical systems: Van der

Pol, Lotka, and Rossler. Results are averaged over 50 trials, each with 50 sec. of training data

increasingly unreliable with more noise. Therefore, under 15% noise we use the smoothed derivative (with window size 21) for validation and use the weak form of the validation data above 15% additive noise.

Figure 4 depicts our results. The general trend is that for low to medium amounts of noise, MIOSR is significantly more accurate, often perfectly recovering the underlying model. Even with just 1% noise, the baselines struggle to identify the true model coefficients. However, unlike in Sect. 3.2, the baselines are mostly identifying all of the correct coefficients, but find small false positive coefficients that are fitting noise.

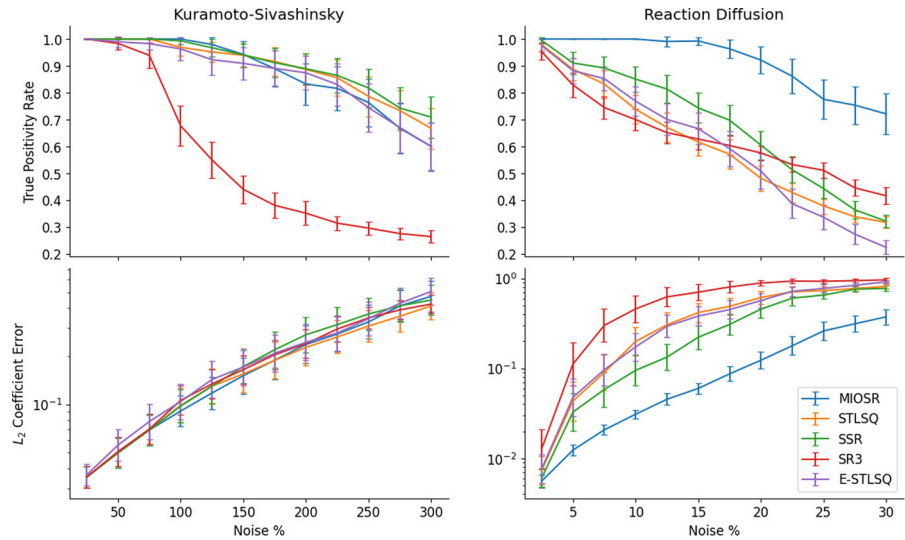
At very high levels of noise, MIOSR starts to break down, converging to the approximate performance of heuristic methods. While more elaborate ensembling could help, we believe better data preparation is likely a

more effective way to learn accurate models, especially with MIOSR. Examples include applying more aggressive smoothing, expanding the number of domains or points per domain in the weak form, or utilizing tailored differentiation techniques [63]. Beyond data preparation, there is also more recent work on more sophisticated iterative schemes to prune the library by regressing on Fourier transforms [24], which could benefit from utilizing optimal methods.

3.6 PDEs

For our last experiment, we study the recovery of scalar PDEs under substantial measurement noise. This is likely the regime most relevant to advancing mod-

Fig. 5 Best PDE model found by sparse regression algorithms for the integral form of SINDy under varying amounts of Gaussian noise for two different canonical PDE systems: Kuramoto–Sivashinsky and reaction diffusion. Results are averaged over 50 trials with random initial conditions



ern science and engineering practice. In particular, we study the one-dimensional Kuramoto–Sivashinsky equation [59], an early model of laminar flame fronts, and the two-dimensional reaction diffusion system, a ubiquitous model in chemistry. The Kuramoto–Sivashinsky in spatial dimension x and time t is given by

$$u_t = -uu_x - u_{xx} - u_{xxx} \quad (31)$$

where the notation u_{xx} denotes the second partial derivative with respect to x . We study the 2D reaction diffusion system given by

$$u_t = \frac{1}{10}u_{xx} + \frac{1}{10}u_{yy} + u - uv^2 - u^3 + v^3 + u^2v \quad (32a)$$

$$v_t = \frac{1}{10}v_{xx} + \frac{1}{10}v_{yy} + v - uv^2 - u^3 - v^3 - u^2v. \quad (32b)$$

For both systems, we use weak third-order polynomial libraries, with up to fourth-order derivatives for Kuramoto–Sivashinsky and second-order derivatives for reaction diffusion, yielding library dimensions of 19 and 109, respectively. For Kuramoto–Sivashinsky, our weak library is composed of 200 spatiotemporal domains, each with 50 points, sampled 10 times a second for 25 sec. on a periodic domain with 1024 spatial points. For reaction diffusion, our weak library is composed of 400 spatiotemporal domains, each with 36 points, sampled 50 times a second for 5 sec. on a 256×256 periodic grid.

Unlike the previous experiments, we do not perform model tuning and selection using AIC, due to the computational cost of fitting a large number of PDEs. Instead, we perform an achievability analysis, loosely inspired by [41], where we choose the algorithm parameters knowing the dynamics, to determine if it is even possible to learn a correct model given an appropriate model selection method. In particular, for MIOSR, we set the sparsity constraint to be the actual dimension-wise sparsity; for STLSQ, we try several thresholds slightly below the actual smallest coefficient; for E-STLSQ, we use library ensembling [25] and take the k terms with the highest inclusion probability, where k is the true sparsity.

Figure 5 depicts our results for PDE learning based on 50 trials with random initial conditions and varying amounts of additive Gaussian noise. These results further underscore how MIOSR has a substantial edge over heuristic methods when using larger libraries or when the system coefficients are of different orders of magnitude; it also further illustrates the converse that problems with smaller libraries and similar magnitude coefficients do not benefit from exact methods because the heuristics converge to the correct solution. Regardless of optimizer, we see how effective the weak SINDy framework can be in identifying noisy systems, with Kuramoto–Sivashinsky often being recovered even with 300% measurement noise.

4 Conclusion

In this work, we demonstrate the superior performance of mixed-integer optimization in learning sparse nonlinear dynamics (MIO-SINDy) as compared with popular heuristic approaches. The biggest advantage is in finding models which are as simple as possible, but not simpler—a model which learns the truth and only the truth. In addition to more accurate support recovery, MIO sparse regression is capable of incorporating a huge range of additional model structure as auxiliary objective terms or constraints, while solving the underlying optimization problem to provable optimality.

Contrary to the predictions of complexity theory, MIOSR is highly tractable and can actually be faster than heuristics for large amounts of data. Indeed, MIOSR runs slower when the regression is harder, that is, when the sample size is small, signal-to-noise ratio is low, or the coefficients span multiple orders of magnitude. However, this is exactly where heuristic methods perform poorly, so MIOSR requires more time when it improves upon heuristic methods while being comparable in running time when the dynamics are more easily recoverable. Given the practicality of the approach, and the theoretical guarantees, we see no reason why MIOSR should not be the default choice of optimizer for real applications.

Due to the modularity of the SINDy framework, MIO-SINDy is compatible with other methodological advancements concerning data preprocessing, library construction, numerical differentiation, and outer loop algorithms. We restricted our study of these extensions to the weak form and PDE learning, but we expect MIOSR to offer similar benefits to other variants like control [17] or identifying implicit equations [30]. We hope domain experts find use for the additional modeling and statistical power afforded by MIO. We believe this is an exciting development that advances the state of the art in system discovery.

Funding The authors declare that no funds, grants, or other support was received during the preparation of this manuscript.

Data availability All data, code, and results are available at our Github repository https://github.com/wesg52/sindy_mio_paper.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

A Additional experiment details

Here we record in greater detail the set of initial conditions we use for each system, the parameter ranges in tuning various algorithms, and other relevant implementation details. Additionally, in our raw results files made available on Github, we include the initial condition, random seed, and chosen hyperparameters for every trial in every experiment.

A1 Sample efficiency

For all systems and algorithms, we use smoothed finite difference differentiation with a smoothing window length of 9. We use the same regularization grid for all systems. For MIOSR, STLSQ, and SSR, we tune over the regularization strength $\alpha \in \{0, 10^{-5}, 10^{-3}, 10^{-2}, 0.05, 0.2\}$. For E-STLSQ, we use the best α for STLSQ. For SR3, we tune relaxation parameter $\nu \in \{\frac{1}{30}, \frac{1}{10}, \frac{1}{3}, 1, \frac{10}{3}\}$. For thresholds, we try to tailor the range based on the system to give the best shot at finding a sparse model (since the heuristics are quite sensitive to the threshold). We choose 50 values uniformly in log space. That is, 10^a for $a \in [b : c : d]$ where $d = 50$ values equally spaced on the interval $[b, c]$. In particular, we use $[-2 : 1 : 50]$, $[-2 : 0 : 50]$, and $[-1.5 : 1.5 : 50]$ for Lorenz, Hopf, and MHD, respectively (where we increase the range by 0.5 for SR3 since it does not use a hard threshold). For MIOSR, we tune the sparsity k for each dimension over integers $k \in [1, 5]$ and SSR by nature fits a model at every level of sparsity between one and the full library size.

Regarding initial conditions, we sample uniformly from a specified volume. For Hopf, we sample in polar coordinates: a radius uniformly at random between 0.75

and 1.25 and an angle at random from 0 to 2π radians. For Lorenz, we sample $x, y \in [-5, 5]$ and $z \in [10, 40]$. For MHD, we sample every coordinate independently from $[-1.5, 1.5]$. We use the same sampling strategy for the runtime experiment.

A2 Physical constraints

For both algorithms, we use smoothed finite difference differentiation with a smoothing window length of 21 to accommodate the noisier data. As before, for SR3 we tune over $\nu \in \{\frac{1}{30}, \frac{1}{10}, \frac{1}{3}, 1, \frac{10}{3}\}$ and 50 thresholds $\lambda = 10^a$, $a \in [-3 : 0 : 20]$. For MIOSR, we tune over a global sparsity constraint $k \in [2, 10]$, and regularizer in $\{0.0001, 0.001, 0.01\}$. As in [21], we sample initial conditions uniformly for each dimension in $[-\pi, \pi]$.

A3 Robustness

For all weak form experiments, we normalize the data matrix to have unit column norm. We use the same values of α and ν as before. Again, in an effort to get the best baseline model, we tailor the thresholds to the system and check that all chosen thresholds fall in the range we tune over. For Van der Pol, Lotka, and Rossler, respectively, we tune the threshold λ over 2^a $a \in [-1 : 5 : 50]$, $[-3 : 4 : 50]$ and $[2 : 6 : 50]$ for STLSQ. For E-STLSQ, we use the same regularization that was optimal for STLSQ.

For Van der Pol, we sample initial conditions from the box $x \in [-1, 1]$, $y \in [-\mu, \mu]$ where we use $\mu = 3$. For Lotka, we sample initial conditions from the box $x, y \in [0, 1]$. For Rossler, we sample uniformly

from the a canonical trajectory with initial condition $(5, 3, 0)$ and add 10% Gaussian noise. For Rossler, we additionally take the absolute value of the z coordinate to preclude unstable trajectories.

A4 PDEs

For achievability analysis, we don't need to tune the sparsity of MIOSR since we can simply use the true sparsity and check if the correct model is learned. However, for (E-)STLSQ the optimal threshold does change because we add substantial noise to the data. Therefore, we still train for thresholds in $\lambda \in [0.4, 2.0, 20]$ and $\lambda \in [0.04, 0.16, 20]$ for Kuramoto–Sivashinsky and reaction diffusion, respectively.

The initial conditions for Kuramoto–Sivashinsky are sampled as $\frac{1}{Z}(\cos(x + r_0) + \sin(4r_1x))$ where $r_0, r_1 \in [0, 1]$ are sampled uniformly at random, x is the 1D mesh on $[0, 2\pi]$ with 1024 grid points, and Z is the normalization factor $\|\cos(x + r_0) + \sin(4r_1x)\|_\infty$. For reaction diffusion, we use a spiral initial condition that is randomly rotated and slightly expanded or contracted. In particular, for X, Y representing the 256×256 spatial mesh

$$u_0 = \tanh((X^2 + Y^2)^{1/2}) \cos((\text{angle}(X + iY) + o) - (s(X^2 + Y^2)^{1/2}))$$

$s \in [0.95, 1.05]$ and $o \in [0, 2\pi]$ both sampled uniformly at random. v_0 is the same but with sin instead of cos.

B Additional results

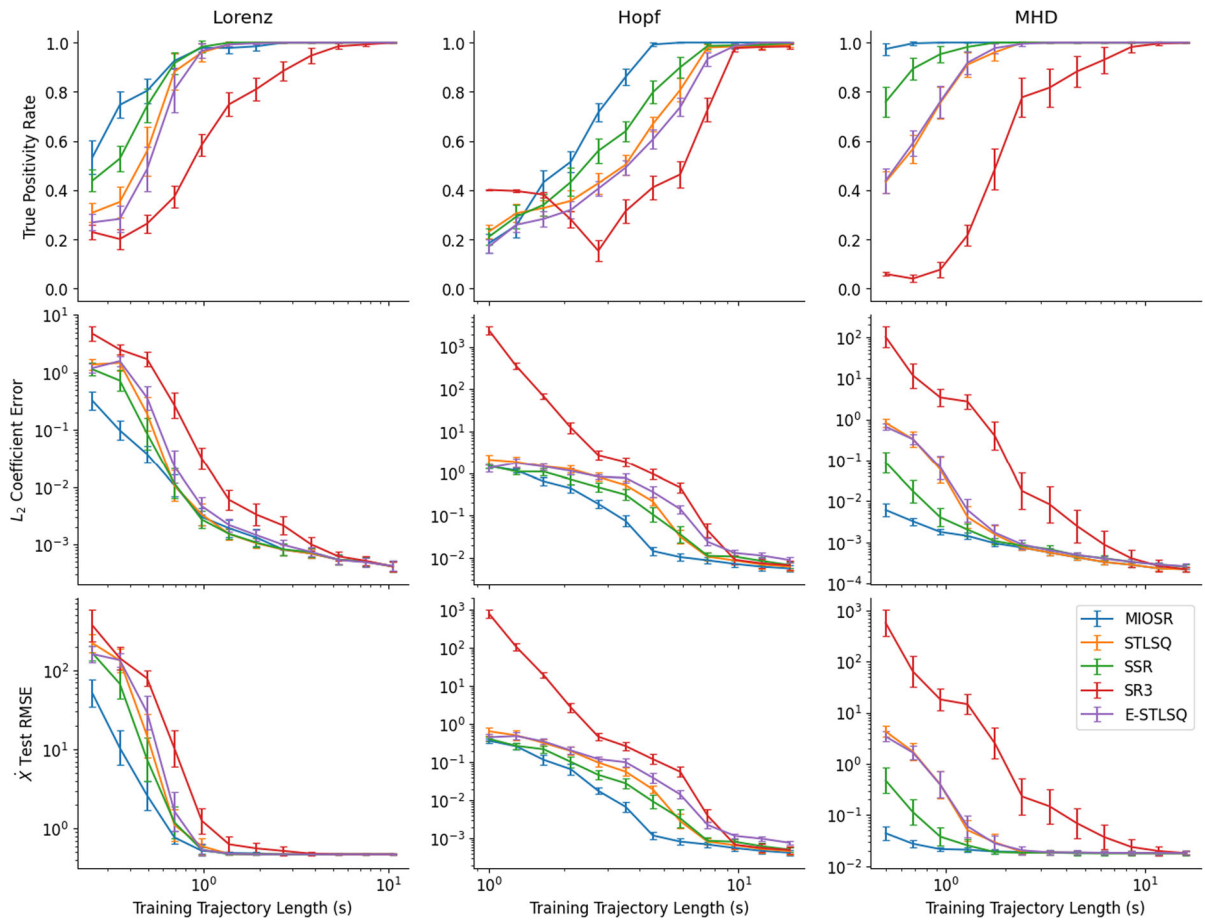


Fig. 6 Performance comparison of sparse regression algorithms for the differential form of SINDy using minimal polynomial libraries under varying amounts of training data for three dif-

ferent canonical systems: Lorenz, Hopf, and MHD. Results are averaged over 50 trials with added Gaussian noise of 0.2%

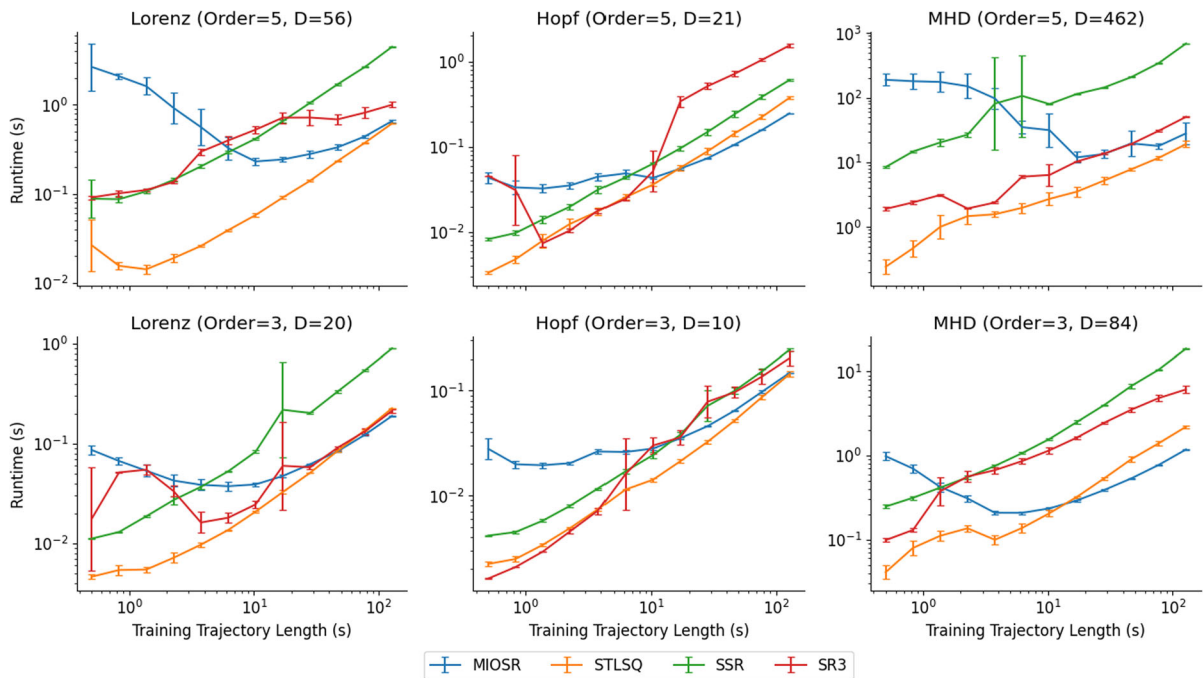


Fig. 7 Comparison of sparse regression algorithm computational efficiency, when executed on a 2021 Macbook Pro, for the differential form of SINDy under varying amounts of training data for three different canonical systems: Lorenz, Hopf, and

MHD. The top row uses a fifth-order polynomial library for each of the three systems while the bottom row uses a third-order polynomial library. Results are averaged over 5 trials with added Gaussian noise of 0.2%

References

- Achterberg, T., Wunderling, R.: Mixed integer programming: analyzing 12 years of progress. In: Facets of Combinatorial Optimization, pp. 449–481. Springer (2013)
- Akaike, H.: A new look at the statistical model identification. *IEEE Trans. Autom. Control* **19**(6), 716–723 (1974)
- Bar-Sinai, Y., Hoyer, S., Hickey, J., Brenner, M.P.: Learning data-driven discretizations for partial differential equations. *Proc. Natl. Acad. Sci.* **116**(31), 15344–15349 (2019)
- Berkooz, G., Holmes, P., Lumley, J.L.: The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rev. Fluid Mech.* **25**(1), 539–575 (1993)
- Bertsimas, D., Copenhaver, M.S.: Characterization of the equivalence of robustification and regularization in linear and matrix regression. *Eur. J. Oper. Res.* **270**(3), 931–942 (2018)
- Bertsimas, D., Digalakis, V.: The backbone method for ultra-high dimensional sparse machine learning. *Mach. Learn.* **111**(6), 1–52 (2022)
- Bertsimas, D., Dunn, J.: *Machine Learning Under a Modern Optimization Lens*. Dynamic Ideas LLC, Belmont (2019)
- Bertsimas, D., King, A., Mazumder, R.: Best subset selection via a modern optimization lens. *Ann. Stat.* **44**(2), 813–852 (2016)
- Bertsimas, D., Li, M.L.: Scalable holistic linear regression. *Oper. Res. Lett.* **48**(3), 203–208 (2020). <https://doi.org/10.1016/j.orl.2020.02.008>
- Bertsimas, D., Pauphilet, J., Van Parys, B.: Sparse regression: scalable algorithms and empirical performance. *Stat. Sci.* **35**(4), 555–578 (2020)
- Bertsimas, D., Van Parys, B.: Sparse high-dimensional regression: exact scalable algorithms and phase transitions. *Ann. Stat.* **48**(1), 300–323 (2020)
- Bertsimas, D., Weismantel, R.: *Optimization Over Integers*. Dynamic Ideas, Boston (2005)
- Bixby, R.E.: Solving real-world linear programs: a decade and more of progress. *Oper. Res.* **50**(1), 3–15 (2002)
- Bongard, J., Lipson, H.: Automated reverse engineering of nonlinear dynamical systems. *Proc. Natl. Acad. Sci.* **104**(24), 9943–9948 (2007)
- Boninsegna, L., Nüske, F., Clementi, C.: Sparse learning of stochastic dynamical equations. *J. Chem. Phys.* **148**(24), 241723 (2018). <https://doi.org/10.1063/1.5018409>
- Brunton, S.L., Proctor, J.L., Kutz, J.N.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci.* **113**(15), 3932–3937 (2016). <https://doi.org/10.1073/pnas.1517384113>
- Brunton, S.L., Proctor, J.L., Kutz, J.N.: Sparse identification of nonlinear dynamics with control (SINDy). *IFAC-PapersOnLine* **49**(18), 710–715 (2016)

18. Carbone, V., Veltri, P.: Relaxation processes in magnetohydrodynamics—a triad-interaction model. *Astron. Astrophys.* **259**, 359–372 (1992)
19. Carderera, A., Pokutta, S., Schütte, C., Weiser, M.: CINDy: conditional gradient-based Identification of Nonlinear Dynamics—Noise-robust recovery. [arXiv:2101.02630](https://arxiv.org/abs/2101.02630) (2021)
20. Champion, K., Lusch, B., Kutz, J.N., Brunton, S.L.: Data-driven discovery of coordinates and governing equations. *Proc. Natl. Acad. Sci.* **116**(45), 22445–22451 (2019)
21. Champion, K., Zheng, P., Aravkin, A.Y., Brunton, S.L., Kutz, J.N.: A unified sparse optimization framework to learn parsimonious physics-informed models from data. *IEEE Access* **8**, 169259–169271 (2020)
22. Chatterjee, A.: An introduction to the proper orthogonal decomposition. *Curr. Sci.* 808–817 (2000)
23. Cplex, I.I.: V12. 8: User’s manual for cplex. *Int. Bus. Mach. Corp.* **46**(53), 157 (2017)
24. Delahunt, C.B., Kutz, J.N.: A toolkit for data-driven discovery of governing equations in high-noise regimes. Preprint [arXiv:2111.04870](https://arxiv.org/abs/2111.04870) (2021)
25. Fasel, U., Kutz, J.N., Brunton, B.W., Brunton, S.L.: Ensemble-SINDy: robust sparse model discovery in the low-data, high-noise limit, with active learning and control. *Proc. R. Soc. A* **478**(2260), 20210904 (2022)
26. Fuentes, R., Dervilis, N., Worden, K., Cross, E.J.: Efficient parameter identification and model selection in nonlinear dynamical systems via sparse Bayesian learning. In: *Journal of Physics: Conference Series*, vol. 1264, p. 012050. IOP Publishing (2019)
27. Gurevich, D.R., Reinbold, P.A., Grigoriev, R.O.: Robust and optimal sparse regression for nonlinear PDE models. *Chaos Interdiscip. J. Nonlinear Sci.* **29**(10), 103113 (2019)
28. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2021). <https://www.gurobi.com>
29. Hazimeh, H., Mazumder, R., Saab, A.: Sparse regression at scale: branch-and-bound rooted in first-order optimization. *Math. Program.* (2021). <https://doi.org/10.1007/s10107-021-01712-4>
30. Kaheman, K., Kutz, J.N., Brunton, S.L.: SINDy-PI: A Robust Algorithm for Parallel Implicit Sparse Identification of Nonlinear Dynamics. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **476**(2242), 20200279 (2020). <https://doi.org/10.1098/rspa.2020.0279>. [arXiv: 2004.02322](https://arxiv.org/abs/2004.02322)
31. Kaiser, E., Kutz, J.N., Brunton, S.L.: Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **474**(2219), 20180335 (2018). <https://doi.org/10.1098/rspa.2018.0335>
32. Kaptanoglu, A.A., Morgan, K.D., Hansen, C.J., Brunton, S.L.: Physics-constrained, low-dimensional models for MHD: First-principles and data-driven approaches. *Phys. Rev. E* **104**(1), 015206 (2021). <https://doi.org/10.1103/PhysRevE.104.015206>. [arXiv: 2004.10389](https://arxiv.org/abs/2004.10389)
33. Kaptanoglu, A.A., de Silva, B.M., Fasel, U., Kaheman, K., Goldschmidt, A.J., Callahan, J., Delahunt, C.B., Nicolaou, Z.G., Champion, K., Loiseau, J.C., Kutz, J.N., Brunton, S.L.: Pysindy: a comprehensive python package for robust sparse system identification. *J. Open Source Softw.* **7**(69), 3994 (2022). <https://doi.org/10.21105/joss.03994>
34. Kreber, D.: Cardinality-constrained discrete optimization for regression (2019)
35. Kronqvist, J., Bernal, D.E., Lundell, A., Grossmann, I.E.: A review and comparison of solvers for convex MINLP. *Optim. Eng.* **20**(2), 397–455 (2019)
36. Kutz, J.N., Brunton, S.L.: Parsimony as the ultimate regularizer for physics-informed machine learning. *Nonlinear Dyn.* **107**(3), 1–17 (2022)
37. Loiseau, J.C., Brunton, S.L.: Constrained sparse Galerkin regression. *J. Fluid Mech.* **838**, 42–67 (2018)
38. Lorenz, E.N.: Deterministic nonperiodic flow. *J. Atmos. Sci.* **20**(2), 130–141 (1963)
39. Lotka, A.J.: *Elements of Physical Biology*. Williams & Wilkins, Philadelphia (1925)
40. Lusch, B., Kutz, J.N., Brunton, S.L.: Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* **9**(1), 1–10 (2018)
41. Maddu, S., Cheeseman, B.L., Sbalzarini, I.F., Müller, C.L.: Stability selection enables robust learning of partial differential equations from limited noisy data. [arXiv:1907.07810](https://arxiv.org/abs/1907.07810) (2019)
42. Mangan, N.M., Kutz, J.N., Brunton, S.L., Proctor, J.L.: Model selection for dynamical systems via sparse regression and information criteria. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **473**(2204), 20170009 (2017). <https://doi.org/10.1098/rspa.2017.0009>
43. Marsden, J.E., McCracken, M.: *The Hopf bifurcation and its applications*, vol. 19. Springer Science & Business Media, Cham (2012)
44. Messenger, D.A., Bortz, D.M.: Weak SINDy for partial differential equations. *J. Comput. Phys.* **443**, 110525 (2021). <https://doi.org/10.1016/j.jcp.2021.110525>. [arXiv: 2007.02848](https://arxiv.org/abs/2007.02848)
45. Messenger, D.A., Bortz, D.M.: Weak sindy: Galerkin-based data-driven model selection. *Multiscale Model. Simul.* **19**(3), 1474–1497 (2021)
46. Pan, W., Yuan, Y., Gonçalves, J., Stan, G.B.: A sparse Bayesian approach to the identification of nonlinear state-space systems. *IEEE Trans. Autom. Control* **61**(1), 182–187 (2015)
47. Pati, Y.C., Rezaifar, R., Krishnaprasad, P.S.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44. IEEE (1993)
48. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics informed deep learning (part i): data-driven solutions of nonlinear partial differential equations. Preprint [arXiv:1711.10561](https://arxiv.org/abs/1711.10561) (2017)
49. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Multistep neural networks for data-driven discovery of nonlinear dynamical systems. Preprint [arXiv:1801.01236](https://arxiv.org/abs/1801.01236) (2018)
50. Reinbold, P.A.K., Gurevich, D.R., Grigoriev, R.O.: Using noisy or incomplete data to discover models of spatiotemporal dynamics. *Phys. Rev. E* **101**(1), 010203 (2020). <https://doi.org/10.1103/PhysRevE.101.010203>
51. Rössler, O.E.: An equation for continuous chaos. *Phys. Lett. A* **57**(5), 397–398 (1976)
52. Rudy, S.H., Brunton, S.L., Proctor, J.L., Kutz, J.N.: Data-driven discovery of partial differential equations. *Sci. Adv.* **3**(4), e1602614 (2017)
53. Rudy, S.H., Sapsis, T.P.: Sparse methods for automatic relevance determination. *Phys. D* **418**, 132843 (2021)

54. Schaeffer, H.: Learning partial differential equations via data discovery and sparse optimization. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **473**(2197), 20160446 (2017)
55. Schaeffer, H., McCalla, S.G.: Sparse model selection via integral terms. *Phys. Rev. E* **96**(2), 023302 (2017). <https://doi.org/10.1103/PhysRevE.96.023302>
56. Schmidt, M., Lipson, H.: Distilling free-form natural laws from experimental data. *Science* **324**(5923), 81–85 (2009)
57. Shen, X., Pan, W., Zhu, Y., Zhou, H.: On constrained and regularized high-dimensional regression. *Ann. Inst. Stat. Math.* **65**(5), 807–832 (2013)
58. de Silva, B.M., Champion, K., Quade, M., Loiseau, J.C., Kutz, J.N., Brunton, S.L.: PySINDy: a Python package for the sparse identification of nonlinear dynamics from data. [arXiv:2004.08424](https://arxiv.org/abs/2004.08424) [physics] (2020)
59. Sivashinsky, G.I.: Nonlinear analysis of hydrodynamic instability in laminar flames-i. derivation of basic equations. *Acta Astronaut.* **4**(11), 1177–1206 (1977)
60. Thompson, R.: Robust subset selection. *Comput. Stat. Data Anal.* 107415 (2022)
61. Tibshirani, R.: Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. Ser. B (Methodol.)* **58**(1), 267–288 (1996)
62. Tillmann, A.M., Bienstock, D., Lodi, A., Schwartz, A.: Cardinality minimization, constraints, and regularization: a survey. Preprint [arXiv:2106.09606](https://arxiv.org/abs/2106.09606) (2021)
63. Van Breugel, F., Kutz, J.N., Brunton, B.W.: Numerical differentiation of noisy data: a unifying multi-objective optimization framework. *IEEE Access* **8**, 196865–196877 (2020)
64. Van der Pol, B.: LXXXVIII. on “relaxation-oscillations.” *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **2**(11), 978–992 (1926)
65. Wehmeyer, C., Noé, F.: Time-lagged autoencoders: deep learning of slow collective variables for molecular kinetics. *J. Chem. Phys.* **148**(24), 241703 (2018)
66. Yeung, E., Kundu, S., Hodas, N.: Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In: 2019 American Control Conference (ACC), pp. 4832–4839. IEEE (2019)
67. Zhang, S., Lin, G.: Robust data-driven discovery of governing physical laws with error bars. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **474**(2217), 20180305 (2018)
68. Zheng, P., Askham, T., Brunton, S.L., Kutz, J.N., Aravkin, A.Y.: A unified framework for sparse relaxed regularized regression: Sr3. *IEEE Access* **7**, 1404–1423 (2018)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.