A joint optimization approach to identifying sparse dynamics using least squares kernel collocation

Alexander W. Hsu*† Ike W. Griss Salas* Jacob M. Stevens-Haas* J. Nathan Kutz*
Aleksandr Aravkin* Bamdad Hosseini*

Abstract

We develop an all-at-once modeling framework for learning systems of ordinary differential equations (ODE) from scarce, partial, and noisy observations of the states. The proposed methodology amounts to a combination of sparse recovery strategies for the ODE over a function library combined with techniques from reproducing kernel Hilbert space (RKHS) theory for estimating the state and discretizing the ODE. Our numerical experiments reveal that the proposed strategy leads to significant gains in terms of accuracy, sample efficiency, and robustness to noise, both in terms of learning the equation and estimating the unknown states. This work demonstrates capabilities well beyond existing and widely used algorithms while extending the modeling flexibility of other recent developments in equation discovery.

1 Introduction

The identification of ordinary differential equations (ODEs) and dynamical systems is a fundamental problem in control [32, 59, 60], data assimilation [42, 84], and more recently in scientific machine learning (ML) [11, 72, 74]. While algorithms such as Sparse Identification of Nonlinear Dynamics (SINDy) and its variants [46] are widely used by practitioners, they often fail in scenarios where observations of the state of the system are scarce, indirect, and noisy. In such scenarios modifications to SINDy-type methods are required to enforce additional constraints on the recovered equations to make them consistent with the observational data. Put simply, traditional SINDy-type methods work in two steps: (1) the data is used to filter the state of the system and estimate the derivatives, and (2) the filtered state is used to learn the underlying dynamics. In the regime of scarce, noisy and incomplete data, step 1 is inaccurate, which can propagate to poor results in the subsequent step 2.

In this paper, we propose an all-at-once approach to filtering and equation learning based on collocation in a reproducing kernel Hilbert space (RKHS) which we term Joint SINDy (JSINDy), and shows that the issues above can be mitigated by performing both steps together. This joins a broader class of dynamics-informed methods that integrate the governing equations directly into the learning objective, either as hard constraints or as least-squares relaxations, which couples the problems of state estimation and model discovery. Representative examples include physics-informed and sparse-regression frameworks based on neural networks, splines, kernels, finite differences, and adjoint methods [21, 27, 39, 41, 72, 73, 88].

*Department of Applied Mathematics, University of Washington, Seattle, WA, USA
†Corresponding author: owlx@uw.edu.

1.1 Problem setup

Consider a process $\boldsymbol{x}(t) = (x_1(t), ..., x_d(t)) \in C^p([0,T], \mathbb{R}^d)$, for $d \geq 1$, along with the observables $(\boldsymbol{y}_n)_{n=1}^N$ that are connected via the model

$$\frac{d^p}{dt^p}\boldsymbol{x}(t) = f(D[\boldsymbol{x}](t)), \quad \boldsymbol{y}_n = \boldsymbol{V}_n^\top \boldsymbol{x}(t_n) + \boldsymbol{\epsilon}_n, \tag{1}$$

$$\text{where} \quad D[\boldsymbol{x}](t) := \left(\boldsymbol{x}(t), \frac{d}{dt}\boldsymbol{x}(t), ..., \frac{d^{p-1}}{dt^{p-1}}\boldsymbol{x}(t)\right), \tag{2}$$

with known measurement matrices $\boldsymbol{V}_n$ (taken to be the identity when the entire state is observed), observation times $\boldsymbol{t} = (t_n)_{n=1}^N$, and measurement noise $\boldsymbol{\epsilon}_n$ which are assumed to be independent with zero mean and bounded variance. Given these observations, our goal is to estimate the nonlinear map $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ from an instance of the $\boldsymbol{y}_n$, i.e., to find $\widehat{f} \approx f$, and simultaneously infer the process $\boldsymbol{x}(t)$, i.e., find $\widehat{\boldsymbol{x}}(t) \approx \boldsymbol{x}(t)$. Additionally, we ask that our representation of $\widehat{f}$ is *sparse* in some basis or library of functions $\Phi = \{\phi_1, ..., \phi_Q\}$. That is, we assume

$$\widehat{f}(D[\boldsymbol{x}](t))_m = \sum_{q=1}^{Q} \widehat{\theta}_{m,q}\phi_q\left(D[\boldsymbol{x}](t)\right), \quad \widehat{f}(D[\boldsymbol{x}]) = (\widehat{f}(D[\boldsymbol{x}](t))_1, ..., \widehat{f}(D[\boldsymbol{x}](t))_d), \tag{3}$$

where only a small number of $\widehat{\theta}_{m,q}$ are nonzero in order to have an interpretable symbolic form, following the seminal work [11, 79, 81, 96]. As we do not know *a priori* which features ϕ_q are active, we parametrize $\widehat{f}$ using the full feature library and promote sparsity in the coefficients $\widehat{\boldsymbol{\theta}} := (\widehat{\theta}_{m,q}), m = 1, \cdots, d, q = 1, \cdots Q$.

Our setup poses a number of fundamental challenges:

1. Broadly speaking we would like to find $\widehat{f}$ such that

$$\widehat{f}(D[\boldsymbol{x}](t)) \approx \frac{d^p}{dt^p}\boldsymbol{x}(t).$$

   However, the input to $\widehat{f}$ as well as the right hand side of the above display depend on $\boldsymbol{x}(t)$ and its derivatives, both of which are unknown and not observed directly.
2. Exacerbating this issue is the fact that the measurements $\boldsymbol{y}_n$ are sparse, indirect, and noisy, which leads to further challenges in estimating x and its derivatives.

Estimating and dealing with the errors in the inputs to $\widehat{f}$ and the target $\frac{d^p}{dt^p}\boldsymbol{x}(t)$ is the fundamental property of system identification problems, which differentiates it from standard approximation or ML tasks. If *all* of these variables were fully observed, i.e., we have direct access to the exact values of $((D[\boldsymbol{x}](t_n))_{n=1}^N$, as well as unbiased estimates of $\frac{d^p}{dt^p}\boldsymbol{x}(t_n)$, then this would reduce to a standard sparse regression problem.

We overcome the above challenges by proposing a joint estimation approach where an approximation $\widehat{\boldsymbol{x}}(t)$ to the true state $\boldsymbol{x}(t)$ is obtained at the same time as the approximation $\widehat{f}(t)$ to the nonlinear dynamics. The key idea in our approach is to enforce self consistency of the $\widehat{\boldsymbol{x}}$ and $\widehat{f}$, i.e., $\widehat{\boldsymbol{x}}$ should approximately satisfy the ODE defined by $\widehat{f}$ while matching the observed data. Since the data is limited and the problem is severely ill-posed, we propose an optimization approach to solve the problem with appropriate regularization terms for both the $\boldsymbol{\theta}$ parameters and the approximate state $\widehat{\boldsymbol{x}}$, in particular, we use a sparse regularizer for the former and a reproducing kernel Hilbert space (RKHS) penalty for the latter.

1.2 Summary of contributions

In view of SINDy and other existing works on equation discovery [11, 21, 27, 38, 39, 88], we summarize our contributions as follows:

1. We develop a RKHS regularized least-squares approach to learning general ODEs directly from sparse and partial observations. (see Section 2).
2. We establish a representer theorem that converts the infinite-dimensional problem into a finite-dimensional nonlinear least-squares system, and design an alternating optimization algorithm based on the Levenberg–Marquardt (LM) method and sparsifying iterations for support selection (see Sections 2.2 and 3).
3. We demonstrate the flexibility and robustness of JSINDy across a broad range of problems, including scarce and partial observations, higher-order systems, and misspecified models, demonstrating accurate recovery in various challenging data regimes (see Section 4).

Our formulation of the problem differs from previous works in two important ways. We allow for arbitrary order ODEs, while previous works are tailored to $p = 1$, and we specifically consider partial, scarce, and scattered observations while previous works often assume access to high resolution full state observations, i.e., $\boldsymbol{y}_n = \boldsymbol{x}(t_n) + \boldsymbol{\epsilon}_n$ on a fine grid.

1.3 Literature Review

Learning continuous time dynamical systems from time series data has been a prominent problem in system identification [5, 47, 60]. Seminal works using symbolic regression [8, 82] through genetic programming sought to discover physical laws from observational data. The works [11, 79, 81, 96] developed and popularized the use of a sparsity prior over a feature library to promote parsimony in the discovered model, drawing on ideas from compressive sensing[13, 14, 23]. Bayesian approaches have also been considered, which allow for uncertainty quantification and motivate new strategies for model selection [38, 65–67, 100, 101].

Many of these works consider a two-step framework which first estimates time-derivatives, then performs regression to estimate the dynamics. These approaches are straightforward to implement and often succeed when measurements of the process are sufficiently dense but fail when the data is scarcely sampled and noisy. These approaches first solve a smoothing problem to estimate the state and its derivatives, using methods such as total variation regularization [18], kernel regression [61], Savitzky–Golay filters, smoothing splines [95], or Kalman smoothing [87]. These approaches inevitably trade bias for robustness: as shown in [94], noise-resistant estimators lie on a Pareto frontier balancing error magnitude and error correlation due to implicitly assuming an inaccurate underlying dynamical model. An alternative to estimating derivatives, but that still lies within a two-step framework is to adopt a weak-form formulation [63], which avoids explicit differentiation by matching integrals against test functions. However, this still requires estimating integrals from noisy and scarcely sampled data, a problem that remains challenging despite being more well posed than differentiation. Consequently, any two-step approach to learning dynamics is inherently limited by the absence of a proper dynamical model in its initial estimation stage.

To address limitations of two-step approaches, dynamics-informed approaches jointly estimate the dynamics and the state, allowing each to reinforce the other. These include Physics-Informed Neural Networks-Sparse Regression (PINN-SR) [21] and Physics-Informed Spline Learning (PiSL) [88] which use least squares collocation applied to neural networks and splines respectively, while obtaining sparsity in the coefficients of the ODE by alternating between applying variants of gradient descent and sequentially thresholded ridge regression. [41] takes a least squares collocation approach using RKHS methods, while also representing the unknown differential equation in an RKHS. The ODR-BINDy method in [27] applies a least squares relaxation to a finite difference discretization, and

handles feature selection by greedily maximizing a Bayesian evidence over models defined by feature sets. [77] also applies a least squares relaxation to a finite difference discretization, but optimizes using a variational annealing approach, and focuses on identifying dynamics with hidden variables. Notably, our objective function (8) is closely related to that of [27, 52, 77, 88] and [21], though we apply different discretization, variable selection, and optimization strategies. [39] applies a finite difference discretization but imposes hard ODE constraints by solving the KKT system associated to the discretized problem. They promote sparsity via ℓ_p regularization, and optimize using iteratively reweighted least squares.

Aside from [39], which considers decreasing sampling frequencies, and [77], which considers hidden variables, none of these works deal with very low sampling frequencies, partial observations through general linear measurements, and higher order dynamics in the context of ODEs. These issues are, however, considered heavily in [21] and [41] in the context of PDEs, where partial information on the state (being infinite dimensional) and higher order derivatives are fundamental to the problem. [39] crucially point out that decoupling the level of discretization in their finite difference stencil from the sampling rate of the data is required for mitigating bias in estimating the ODE when the sampling rate of the data is low, and demonstrate this in a restricted setting on the the Van der Pol system. We generally consider sampling rates over an order of magnitude lower than other works and flexibly handle partial observations and higher order ODEs.

1.4 Outline

We derive JSINDy in Section 2, describe the algorithmic implementation in Section 3, give extensive numerical results in Section 4, and discuss the ODE relaxation in Section 5.

1.5 Notation

We will use boldface for vectors, matrices, and vector-valued trajectories and use subscripts to denote indices, e.g., $\boldsymbol{x}(t) = (x_1(t), x_2(t), ..., x_d(t))$. We adopt semicolon notation for parametrized functions, such as $\boldsymbol{x}(t; \boldsymbol{z})$, and use a dot $(\cdot)$ in place of an argument in order to treat an object as a function. For a function k of two arguments, and vectors $\boldsymbol{t}, \boldsymbol{\tau}$, we will write $k(\boldsymbol{t}, \boldsymbol{\tau})$ to mean the matrix with entries $k(t_i, \tau_m)$. We will write ∂_j^r to denote r-th derivative with respect to the j-th argument.

2 JSINDy: Joint State Estimation and System Identification

In this section, we formulate our approach as an infinite dimensional optimization problem in the product of an RKHS with $\mathbb{R}^{d\times Q}$. We discretize this objective via a quadrature rule and develop a representer theorem which reduces the optimization to a finite dimensional subspace, and prove existence of solutions to both the discretized and continuous problems.

2.1 The abstract formulation

Our approach centers around the joint computation of the estimated state $\widehat{\boldsymbol{x}}(t)$, henceforth called *state estimation*, as well as the right hand side $\widehat{f}(x)$, henceforth called *system identification*. We will require that the pair $\widehat{\boldsymbol{x}}$ and $\widehat{f}$ are *self consistent*, in the sense that $\widehat{\boldsymbol{x}}$ should (approximately) satisfy the ODE defined by $\widehat{f}$:

$$\frac{d^p}{dt^p}\widehat{\boldsymbol{x}}(t) = \widehat{f}\left(\widehat{\boldsymbol{x}}(t), \frac{d}{dt}\widehat{\boldsymbol{x}}(t), \ldots, \frac{d^{p-1}}{dt^{p-1}}\widehat{\boldsymbol{x}}(t)\right).$$

To this end, we impose self-consistency using a least squares penalty method [3, 54–56, 73, 99], which is discretized via collocation in an RKHS. Imposing that $\widehat{f}$ has a sparse representation in the library

$\Phi = \{\phi_1, \ldots, \phi_Q\}$, we adopt a parametrized notation for how it operates on $\boldsymbol{x}$ and its derivatives:

$$f(D[\boldsymbol{x}](t); \boldsymbol{\theta}) := \left(\sum_{q=1}^{Q} \theta_{1,q} \phi_q \left(D[\boldsymbol{x}](t) \right), \ldots, \sum_{q=1}^{Q} \theta_{d,q} \phi_q \left(D[\boldsymbol{x}](t) \right) \right), \quad \boldsymbol{\theta} \in \mathbb{R}^{d \times Q}. \tag{4}$$

For instance, if we consider the first order constant coefficient linear ODE $\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}$, then we $D[\boldsymbol{x}](t) = \boldsymbol{x}$, and we would be looking for $\boldsymbol{\theta} = \boldsymbol{A}$.

We define our estimators $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ as solutions to the following optimization problem.

$$\begin{aligned}
&\underset{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\theta} \in \mathbb{R}^{d \times Q}}{\text{minimize}} \quad \mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}) \\
&\text{subject to} \quad \operatorname{supp}(\boldsymbol{\theta}) \subset \mathcal{S}\left(\boldsymbol{x}\right), \\
&\qquad \mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}) := \alpha \, \mathcal{L}_D(\boldsymbol{x}) + \beta \, \mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta}) + \frac{\lambda}{2} \|\boldsymbol{x}\|_{\mathcal{X}}^2 + \frac{\mu}{2} \|\boldsymbol{\theta}\|_F^2, \\
&\qquad \mathcal{L}_D(\boldsymbol{x}) := \frac{1}{2} \sum_{n=1}^{N} \left| \boldsymbol{V}_n^\top \boldsymbol{x}(t_n) - \boldsymbol{y}_n \right|^2, \\
&\qquad \mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta}) := \frac{1}{2} \int_0^T \left| \frac{d^p}{dt^p} \boldsymbol{x}(t) - f\left(D[\boldsymbol{x}](t); \boldsymbol{\theta}\right) \right|^2 dt.
\end{aligned} \tag{5}$$

where

- $\alpha, \beta, \lambda, \mu > 0$ are relative weights trading off data fidelity, self consistency through the ODE, and regularization of the trajectory estimate and coefficients.
- $\mathcal{L}_D$ is a data misfit term enforcing that the estimated trajectory $\widehat{\boldsymbol{x}}$ is consistent with the observed data.
- $\mathcal{L}_C$ is a least squares penalty enforcing that $\widehat{\boldsymbol{x}}$ approximately satisfies the ODE induced by $\widehat{f}$.
- $D[\boldsymbol{x}](t)$, defined in eq. (2) expands the trajectory $\boldsymbol{x}(t)$ into the states and derivatives.
- $\mathcal{X}$ is a vector-valued RKHS used to represent the trajectories, whose elements are smooth enough such that $D[\boldsymbol{x}](t)$ is well defined pointwise for all $t \in [0, T]$, and $\|\boldsymbol{x}\|_{\mathcal{X}}$ promotes regularity in the estimates of the trajectory. See the discussion below and Section A.2 for more technical details.
- $\mathcal{S}$ is a feature selection procedure for choosing active terms from the feature library Φ given a fixed trajectory estimate, and maps to a subset of indices. This can be seen as applying a traditional SINDy procedure, with an arbitrary choice of sparsifier, and using exact derivatives of the approximation of the trajectory in order to choose the subset of features which are active. Because we handle sparsity through the feature selection operator $\mathcal{S}$ we do not need to apply regularization to $\boldsymbol{\theta}$ to promote sparsity. In principal, however, the Frobenius norm $\frac{\mu}{2}\|\boldsymbol{\theta}\|_F^2$ could be replaced by a more general sparsity promoting regularizer $R(\boldsymbol{\theta})$ which would supplant $\mathcal{S}$.

The inclusion of a feature selection procedure is motivated by a natural algorithmic idea for sparse nonlinear optimization. Consider the fixed point iteration

$$\begin{aligned}
(\boldsymbol{x}^{(k+1)}, \boldsymbol{\theta}^{(k+1)}) &\in \begin{cases} \underset{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\theta} \in \mathbb{R}^{d \times Q}}{\arg\min} & \mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} & \operatorname{supp}(\boldsymbol{\theta}) \subset S^{(k)} \end{cases} \\
S^{(k+1)} &= \mathcal{S}\left(\boldsymbol{x}^{(k+1)}\right),
\end{aligned} \tag{6}$$

which alternates between optimizing on an active set of features, and choosing features to use in the active set. Given a solution $(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$ to (5), define $S^* = \mathcal{S}(\boldsymbol{x}^*)$. If

$$(\boldsymbol{x}^*, \boldsymbol{\theta}^*) \in \underset{\substack{\boldsymbol{x} \in \mathcal{X}, \ \boldsymbol{\theta} \in \mathbb{R}^{d \times Q} \\ \operatorname{supp}(\boldsymbol{\theta}) \subset S^\star}}{\arg\min} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\theta}), \tag{7}$$

then the pair $(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$ is a fixed point of (6). This is a strong condition, but is likely to hold whenever $\mathcal{S}$ is a reasonably good sparsifier that is consistent with the collocation loss $\mathcal{L}_C$. This is also a desirable property in itself; we want for our estimate fixed to the active set to be best estimate possible.

The ADO algorithms of [21, 88] are exactly of the form (6) and motivate it as a surrogate to ℓ_0 minimization. Similarly, the greedy evidence maximization in [27] is related in that it solves a sequence of fixed-support nonlinear least squares problems, though in each iteration, separate subproblems are solved for each candidate feature to remove. This has the benefit of minimizing a fixed objective function, and ensuring convergence due to the monotone feature removal process. This is discussed in detail algorithmically in Section 3.1 and Section 3.3.

In our theoretical analysis, we will remove the constraint and work with

$$\underset{\boldsymbol{x}\in\mathcal{X},\boldsymbol{\theta}\in\mathbb{R}^{d\times Q}}{\text{minimize}} \quad \mathcal{L}(\boldsymbol{x},\boldsymbol{\theta}), \tag{8}$$

which describes the first piece of the fixed point iteration and still entails solving a challenging nonlinear programming problem.

This approach combines regularization on the trajectory estimate with a least squares relaxation of the differential equation, which prevents trajectory estimates from diverging away from the observed data. Further, this relaxation allows the observed data itself to guide the optimization in the early iterations, pulling our state estimates towards a neighborhood of the true trajectory even before our estimates effectively approximate the solution to the ODE.

To represent the d-dimensional trajectory $\boldsymbol{x} : [0,T] \to \mathbb{R}^d$, we work in a vector-valued RKHS $\mathcal{X}$ of functions $[0,T] \mapsto \mathbb{R}^d$ generated by the d-fold Cartesian product of an RKHS $\mathcal{H}_k$, associated to a symmetric positive definite kernel k. This is equivalent to modeling each coordinate $x_i(t)$ of the trajectory $\boldsymbol{x}(t)$ separately in the RKHS $\mathcal{H}_k$, or by taking $\mathcal{X}$ to be the vector-valued RKHS with matrix valued kernel $K(t,t') = k(t,t')I_d$. We will require that for derivative orders $j = 0, ..., p$, and times $t \in [0,T]$, the derivative evaluation map $\boldsymbol{x} \mapsto \frac{d^j}{dt^j}\boldsymbol{x}(t)$ is a bounded linear operator on $\mathcal{X}$ (see Section A.2). It suffices to take k such that $\mathcal{H}_k$ is compactly embedded in $C^p([0,T])$, which is satisfied if $k \in C^{p,p}([0,T]\times[0,T])$ [103].

We choose the kernel k to be the sum of a high-order Matérn kernel [75] and a constant kernel to account for systems where the time-average is nonzero. More precisely, with h_ν denoting the ν-order Matérn covariance function and ℓ as a length scale parameter, we use kernels of the form

$$k(t,t') = c_0 + c_1 h_\nu\left(\frac{|t-t'|}{\ell}\right), \quad h_\nu(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu}r\right)^\nu K_\nu\left(\sqrt{2\nu}\,r\right) \tag{9}$$

where K_ν is the modified Bessel function of the second kind, and $c_0, c_1 > 0$. These kernels induce RKHSs which satsify the boundedness condition of the derivative evaluation maps whenever $\nu > p + \frac{1}{2}$. See Section A.2 for more details on kernels and smoothness.

As a preliminary step in our model, we choose the kernel hyperparameters c_0, c_1, ℓ, and the variance of the observation noise $\sigma^2 = \mathbb{V}[\epsilon]$ using maximum likelihood and treat the distribution of ϵ as Gaussian, as is common in Gaussian process regression; see for example [75]. From there, $\|\boldsymbol{x}\|_{\mathcal{X}}^2$ corresponds to the regularization induced by the RKHS norm associated to this fitted Gaussian process.

2.2 Discretization

In order to obtain a computable objective function, we discretize the ODE residual $\mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta})$ using a quadrature rule, setting

$$\mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2}\int_0^T \left| \frac{d^p}{dt^p}\boldsymbol{x}(\tau) - f\left(D[\boldsymbol{x}](\tau); \boldsymbol{\theta}\right)\right|^2 d\tau \tag{10}$$

$$\approx \frac{1}{2}\sum_{m=1}^{M} w_m \left| \frac{d^p}{dt^p}\boldsymbol{x}(\tau_m) - f\left(D[\boldsymbol{x}](\tau_m); \boldsymbol{\theta}\right)\right|^2 := \mathcal{L}_C^{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{\theta}) \tag{11}$$

where we refer to $\boldsymbol{\tau} = (\tau_m)_{m=1}^M$ as collocation points, and $\boldsymbol{w} = (w_m)_{m=1}^M$ as collocation weights. In all of our examples, we take uniformly spaced collocation nodes and uniform weights; this is discussed further in Section 5. Further, for notational convenience and without loss of generality, we assume that the observation times $\boldsymbol{t} = (t_n)_{n=1}^N$ are a subset of the collocation points $\boldsymbol{\tau}$, which can be enforced by augmenting $\boldsymbol{\tau}$, and taking some w_m to be zero.

Substituting $\mathcal{L}_C^{\boldsymbol{\tau}}$ for $\mathcal{L}_C$ in eq. (8), we obtain the optimization problem

$$\begin{aligned}\underset{\boldsymbol{x}\in\mathcal{X},\, \boldsymbol{\theta}\in\mathbb{R}^{d\times Q}}{\text{minimize}}\ \mathcal{L}^{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{\theta}) := {}& \frac{\alpha}{2}\sum_{n=1}^{N}\left|\boldsymbol{V}_n^\top \boldsymbol{x}(t_n) - \boldsymbol{y}_n\right|^2 \\ &+ \frac{\beta}{2}\sum_{m=1}^{M} w_m \left| \frac{d^p}{dt^p}\boldsymbol{x}(\tau_m) - f\left(D[\boldsymbol{x}](\tau_m); \boldsymbol{\theta}\right)\right|^2 \\ &+ \frac{\lambda}{2}\|\boldsymbol{x}\|_{\mathcal{X}}^2 + \frac{\mu}{2}\|\boldsymbol{\theta}\|_F^2\end{aligned} \tag{12}$$

which we use to compute our estimators. At this point, the reproducing property and boundedness of derivative evaluations in the RKHS $\mathcal{X}$ give a natural parametrization of $\widehat{\boldsymbol{x}}$ which is justified by the following *representer theorem*, which converts optimization over the infinite-dimensional space $\mathcal{X}$ into a problem on a finite-dimensional subspace. This result is more general than the typical representer theorems often found in ML, such as in [75], which only make use of pointwise evaluations, but is very similar to the representer theorem given in [20], and follows from the more general results of [70]. The purpose of this result is to reduce the optimization problem over the infinite-dimensional space $\mathcal{X}$ into an optimization problem over the finite-dimensional subspace spanned by an explicit set of basis functions.

Theorem 2.1. *Consider the optimization problem (12) for $\alpha, \beta, \lambda, \mu > 0$. Let $\mathcal{X}$ be the vector-valued RKHS associated to the d-fold Cartesian product of an RKHS $\mathcal{H}_k$ associated to the positive definite kernel k. Assume that linear operators of the form $\boldsymbol{x} \mapsto \frac{d^r}{dt^r}\boldsymbol{x}(\tau)$ for $r = 0, ..., p$ and $\tau \in [0, T]$ are bounded in the RKHS $\mathcal{X}$. Also, assume that all functions ϕ_q in the library Φ are continuous on $\mathbb{R}^{dp}$.*

1. *There exists a minimizer $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ to the optimization problem*

$$\underset{\boldsymbol{x}\in\mathcal{X},\, \boldsymbol{\theta}\in\mathbb{R}^{d\times Q}}{\text{minimize}}\ \mathcal{L}^{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{\theta}).$$

2. *Define the basis functions*

$$D_{\boldsymbol{\tau}}[k](t) := \left(k(t, \tau_1), \partial_2 k(t, \tau_1), \ldots, \partial_2^p k(t, \tau_1), \ldots, k(t, \tau_M), \ldots, \partial_2^p k(t, \tau_M)\right) \in \mathbb{R}^{(p+1)M}, \tag{13}$$

and for $\boldsymbol{z} \in \mathbb{R}^{(p+1)M\times d}$, set

$$\boldsymbol{x}(t; \boldsymbol{z}) := D_{\boldsymbol{\tau}}[k](t)^\top \boldsymbol{z}. \tag{14}$$

Then problem (12) *is equivalent to*

$$\underset{\boldsymbol{z}\in\mathbb{R}^{(p+1)M\times d},\,\boldsymbol{\theta}\in\mathbb{R}^{d\times Q}}{\text{minimize}}\ \mathcal{L}^{\boldsymbol{\tau}}\big(\boldsymbol{x}(\cdot;\boldsymbol{z}),\boldsymbol{\theta}\big), \tag{15}$$

in the sense that they have the same optimal value and $(\widehat{\boldsymbol{x}},\widehat{\boldsymbol{\theta}})$ *is a minimizer if and only if* $\widehat{\boldsymbol{x}}$ *admits a representation* $\widehat{\boldsymbol{x}}(t) = D_{\boldsymbol{\tau}}[k](t)^\top \widehat{\boldsymbol{z}}$ *for some* $\widehat{\boldsymbol{z}}$*, where* $(\widehat{\boldsymbol{z}},\widehat{\boldsymbol{\theta}})$ *is a solution to* (15)*.*

3. *For all* $\boldsymbol{z}\in\mathbb{R}^{(p+1)M\times d}$*,*

$$\|\boldsymbol{x}(\cdot;\boldsymbol{z})\|_{\mathcal{X}}^2 = \operatorname{tr}\big(\boldsymbol{z}^\top \boldsymbol{K}\,\boldsymbol{z}\big),\quad \boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{\tau},\boldsymbol{\tau}) & \partial_2^1 k(\boldsymbol{\tau},\boldsymbol{\tau}) & \cdots & \partial_2^p k(\boldsymbol{\tau},\boldsymbol{\tau}) \\ \partial_1^1 k(\boldsymbol{\tau},\boldsymbol{\tau}) & \partial_1^1\partial_2^1 k(\boldsymbol{\tau},\boldsymbol{\tau}) & \cdots & \partial_1^1\partial_2^p k(\boldsymbol{\tau},\boldsymbol{\tau}) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_1^p k(\boldsymbol{\tau},\boldsymbol{\tau}) & \partial_1^p\partial_2^1 k(\boldsymbol{\tau},\boldsymbol{\tau}) & \cdots & \partial_1^p\partial_2^p k(\boldsymbol{\tau},\boldsymbol{\tau}). \end{bmatrix} \in \mathbb{R}^{(p+1)M\times(p+1)M}. \tag{16}$$

Proof. We have that

$$\mathcal{L}^{\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{\theta}) = \alpha\mathcal{L}_D(\boldsymbol{x}) + \beta\mathcal{L}_C^{\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{\theta}) + \frac{\lambda}{2}\|\boldsymbol{x}\|_{\mathcal{X}}^2 + \frac{\mu}{2}\|\boldsymbol{\theta}\|_F^2. \tag{17}$$

By nesting, problem (12) is equivalent to

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^{d\times Q}}{\text{minimize}}\ \inf_{\boldsymbol{x}\in\mathcal{X}} \mathcal{L}^{\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{\theta}) \tag{18}$$

and we can thus focus solely on the dependence with respect to $\boldsymbol{x}$, studying the inner variational problem, as the outer problem is already finite dimensional. Going term by term, we have

$$\mathcal{L}_D(\boldsymbol{x}) = \frac{1}{2}\sum_{n=1}^{N}\left|\boldsymbol{V}_n^\top \boldsymbol{x}(t_n) - \boldsymbol{y}_n\right|^2, \tag{19}$$

which depends only on point evaluations of the form $\boldsymbol{x}(\tau_m)$ (recall that we assume the observation times $\boldsymbol{t}$ are a subset of $\boldsymbol{\tau}$). Next, we have that

$$\mathcal{L}_C^{\boldsymbol{\tau}}(\boldsymbol{x},\boldsymbol{\theta}) = \frac{1}{2}\sum_{m=1}^{M} w_m\left|\frac{d^p}{dt^p}\boldsymbol{x}(\tau_m) - f\left(D[\boldsymbol{x}](\tau_m);\boldsymbol{\theta}\right)\right|^2, \tag{20}$$

depends on $\boldsymbol{x}$ only through point evaluations of $\boldsymbol{x}$ and its derivatives at $(\tau_m)_{m=1}^M$, i.e., the values of $D[\boldsymbol{x}](\tau_m)$ and $\frac{d^p}{dt^p}\boldsymbol{x}(\tau_m)$.

Recall that linear operators of the form $\boldsymbol{x}\mapsto\frac{d^r}{dt^r}\boldsymbol{x}(\tau)$ for $r=0,...,p$ and $\tau\in[0,T]$ are assumed to be bounded on the RKHS $\mathcal{X}$. Thus, by the reproducing property [103],

$$\frac{d^r}{dt^r}x_i(\tau_m) = \left\langle \partial_2^r k(\cdot,\tau_m), x_i(\cdot)\right\rangle_{\mathcal{H}_k}. \tag{21}$$

Since the objective $\mathcal{L}^{\boldsymbol{\tau}}$ depends only on the values of $\frac{d^r}{dt^r}\boldsymbol{x}(\tau_m)$ for $r=0,1,...,p$, $m=1,...,M$, with the addition of the squared RKHS norm, this implies that

$$\widehat{x}_i \in \operatorname{span}\Big\{\partial_2^r k(\,\cdot\,,\tau_m) : m=1,\dots,M, r=0,\dots,p\Big\}, \tag{22}$$

by Theorem A.1. This is precisely the subspace spanned by the basis $D_{\boldsymbol{\tau}}[k](\cdot)$ in (13). By the reproducing property, we also have that

$$\left\langle \partial_2^r k(\cdot,\tau_m), \partial_2^s k(\cdot,\tau_{m'})\right\rangle_{\mathcal{H}_k} = \partial_1^r\partial_2^s k(\tau_m,\tau_{m'}), \tag{23}$$

which gives the structure of the matrix $\boldsymbol{K}$, and the RKHS norm of individual component of the trajectory $x_i(\cdot) \in \operatorname{span}\{\partial_2^r k(\cdot, \tau_m),\ r = 0, 1, ..., p, m = 1, ..., M\}$. Stacking the individual components together and noting that

$$\boldsymbol{x}(t) = (x_1(t), ..., x_d(t)) \text{ and } \|\boldsymbol{x}\|_{\mathcal{X}}^2 = \sum_{i=1}^{d} \|x_i\|_{\mathcal{H}_k}^2 \tag{24}$$

produces the compact representation in (14) and the formula for the RKHS norm in (16).

We obtain existence of minimizers from continuity and coercivity. The objective is continuous with respect to $\boldsymbol{z}$ and $\boldsymbol{\theta}$ due to the continuity of the feature maps ϕ_q. If $\boldsymbol{K}$ is invertible, then the objective is coercive and thus attains a minimum. If it is not invertible, then we may, without affecting $\mathcal{L}_D$ or $\mathcal{L}_C^{\boldsymbol{\tau}}$, restrict $\boldsymbol{z}$ to the range of $\boldsymbol{K}$; this is essentially applying a finite-dimensional representer theorem to the subspace spanned by $D_{\boldsymbol{\tau}}[k]$. Under this additional constraint, the objective becomes coercive, giving existence of minima. Finally, the equivalence of the optimization problems allows us to transfer existence from problem 15 to problem 12. □

A direct application of Theorem 2.1 gives the finite-dimensional objective function

$$\begin{aligned} \mathcal{J}(\boldsymbol{z}, \boldsymbol{\theta}) := \mathcal{L}^{\boldsymbol{\tau}}(\boldsymbol{x}(\cdot; \boldsymbol{z}), \boldsymbol{\theta}) = &\frac{\alpha}{2} \sum_{n=1}^{N} \left| \boldsymbol{V}_n^\top \boldsymbol{x}(t_n; \boldsymbol{z}) - \boldsymbol{y}_n \right|^2 \\ &+ \frac{\beta}{2} \sum_{m=1}^{M} w_m \left| \frac{d^p}{dt^p} \boldsymbol{x}(\tau_m; \boldsymbol{z}) - f\left(D[\boldsymbol{x}](\tau_m; \boldsymbol{z})\right) \right|^2 \\ &+ \frac{\lambda}{2} \|\boldsymbol{z}\|_{\boldsymbol{K}}^2 + \frac{\mu}{2} \|\boldsymbol{\theta}\|_F^2,\ \|\boldsymbol{z}\|_{\boldsymbol{K}}^2 := \operatorname{tr}(\boldsymbol{z}^T \boldsymbol{K} \boldsymbol{z}) \end{aligned} \tag{25}$$

We also give a theorem on the existence of solutions to the continuous problem eq. (8).

Theorem 2.2. *Consider the optimization problem (8). Let $\alpha, \beta, \lambda, \mu > 0$. Let $\mathcal{X}$ be the vector-valued RKHS associated to the d-fold Cartesian product of an RKHS $\mathcal{H}_k$ associated to the positive definite kernel k. Assume that $\mathcal{H}_k$ is compactly embedded in $C^p([0, T], \mathbb{R}^d)$. Assume that all functions ϕ_q in the library Φ are continuous on $\mathbb{R}^{dp}$. Then there exists a minimizer $(\boldsymbol{x}^*, \boldsymbol{\theta}^*) \in \mathcal{X} \times \mathbb{R}^{d \times Q}$*

Proof. Because the features ϕ_q are assumed to be continuous functions, and point evaluations of derivatives up to order p are bounded on $\mathcal{H}_k$ due to the compact embedding property, $\mathcal{L}_C$ and $\mathcal{L}_D$ are continuous on both $\mathcal{X} \times \mathbb{R}^{d \times Q}$ and $C^p([0, T] \times \mathbb{R}^d)$. Additionally, both are lower bounded by zero. Because R is lower semicontinuous on $\mathbb{R}^{d \times Q}$, we have $\mathcal{L}$ is lower semicontinuous on $C^p([0, T], \mathbb{R}^d)$. Since $\frac{\lambda}{2}\|\boldsymbol{x}\|_{\mathcal{X}}^2 + \frac{\mu}{2}\|\boldsymbol{\theta}\|_F^2$ is coercive on $\mathcal{X} \times \mathbb{R}^{d \times Q}$, we obtain that $\mathcal{L}$ is coercive as well. For a minimizing sequence $(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k)})$, and we may extract a subsequence that strongly converges in $(C^p([0, T], \mathbb{R}^d)$ to a candidate minimizer $(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$. Lower semicontinuity of $\mathcal{L}$ yields that $\lim_{k \to \infty} \mathcal{L}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k)}) = \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\theta}^*)$. □

3 Algorithms and implementation details

Reintroducing the sparse selection procedure, we obtain the optimization problem

$$\begin{aligned} \underset{\boldsymbol{z} \in \mathbb{R}^{(p+1)M \times d}, \boldsymbol{\theta} \in \mathbb{R}^{d \times Q}}{\text{minimize}} \quad & \mathcal{J}(\boldsymbol{z}, \boldsymbol{\theta}) \\ \text{subject to} \quad & \operatorname{supp}(\boldsymbol{\theta}) \subset \mathcal{S}(\boldsymbol{x}). \end{aligned} \tag{26}$$

In order to numerically compute solutions to 26, we alternate between applying a Levenberg-Marquardt (LM) algorithm to solve smooth nonlinear least squares problems with the support of

$\boldsymbol{\theta}$ restricted, and a sparse regression algorithm to choose the support of $\boldsymbol{\theta}$. We first write $\mathcal{J}$ as a regularized nonlinear least squares objective. Define the residual functions

$$\boldsymbol{F}_D(\boldsymbol{z}) := \begin{bmatrix} \boldsymbol{V}_1^\top \boldsymbol{x}(t_1;\boldsymbol{z}) - \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{V}_N^\top \boldsymbol{x}(t_N;\boldsymbol{z}) - \boldsymbol{y}_N \end{bmatrix}, \quad \boldsymbol{F}_C(\boldsymbol{z},\boldsymbol{\theta}) = \begin{bmatrix} \sqrt{w_1}\big(\frac{d^p}{dt^p}\boldsymbol{x}(\tau_1;\boldsymbol{z}) - f\big(D[\boldsymbol{x}](\tau_1;\boldsymbol{z});\,\boldsymbol{\theta}\big)\big) \\ \vdots \\ \sqrt{w_M}\big(\frac{d^p}{dt^p}\boldsymbol{x}(\tau_M;\boldsymbol{z}) - f\big(D[\boldsymbol{x}](\tau_M;\boldsymbol{z});\,\boldsymbol{\theta}\big)\big). \end{bmatrix} \tag{27}$$

With

$$\boldsymbol{F}(\boldsymbol{z},\boldsymbol{\theta}) = \begin{bmatrix} \sqrt{\alpha}\boldsymbol{F}_D(\boldsymbol{z}) \\ \sqrt{\beta}\boldsymbol{F}_C(\boldsymbol{z},\boldsymbol{\theta}) \end{bmatrix}, \tag{28}$$

observe that

$$\mathcal{J}(\boldsymbol{z},\boldsymbol{\theta}) = \frac{1}{2}\left\|\boldsymbol{F}(\boldsymbol{z},\boldsymbol{\theta})\right\|^2 + \frac{\lambda}{2}\mathrm{tr}(\boldsymbol{z}^\top \boldsymbol{K}\boldsymbol{z}) + \frac{\mu}{2}\|\boldsymbol{\theta}\|_F^2. \tag{29}$$

The addition of the quadratic regularization term $\frac{1}{2}\boldsymbol{z}^T\boldsymbol{K}\boldsymbol{z}$ is completely benign in the nonlinear least squares setting, and may be treated explicitly in Gauss-Newton Hessian approximations. We further perform a change of variables using the Cholesky factors of $\boldsymbol{K}$ in order to transform the quadratic regularization into a multiple of the identity, see Section A.3. Additionally, note that $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{\theta})$ is affine with respect to $\boldsymbol{\theta}$, which makes it natural to apply methods for sparse linear least squares for the feature selection procedure $\mathcal{S}$; see Section 3.3.

3.1 Optimization

We compute minimizers of (15) by alternating between *active-set iterations*, and *sparsifying iterations*.

1. **Active-set iterations** refine the estimates of $\boldsymbol{z},\boldsymbol{\theta}$ under the condition that $\mathrm{supp}(\boldsymbol{\theta}) \subset S^{(k)}$ for a fixed active set $S^{(k)}$:

$$(\boldsymbol{z}^{(k+1)},\boldsymbol{\theta}^{(k+1)}) = \underset{\boldsymbol{z},\boldsymbol{\theta}}{\arg\min}\, \mathcal{J}(\boldsymbol{x},\boldsymbol{\theta}), \text{ subject to } \mathrm{supp}(\boldsymbol{\theta}) \subset S^{(k)} \tag{30}$$

   These iterations drop the features not selected in step (2) and solve smooth nonlinear least-squares problems restricted to the identified support using LM. Taking previous parameter values as a warm start greatly improves efficiency. More algorithmic details are discussed in Section 3.2.
2. **Sparsifying iterations** freeze $\boldsymbol{z} = \boldsymbol{z}^{(k)}$ and compute

$$S^{(k+1)} = \mathcal{S}(\boldsymbol{x}(\cdot;\boldsymbol{z}^{(k)})) \tag{31}$$

   in order to estimate a new active set. Once the estimate $\boldsymbol{x}(\cdot,\boldsymbol{z})$ of the trajectory is fixed, a natural strategy is to target the sparse regression problem of finding sparse $\boldsymbol{\theta}$ which approximately minimizes the collocation error $\mathcal{L}_C^\tau(\boldsymbol{x}(\cdot;\boldsymbol{z}^{(k)}),\boldsymbol{\theta})$. This objective then has a *linear* least squares structure, and $S^{(k+1)}$ is taken to be the support of the identified solution. This approach can be seen as applying a traditional 2-step SINDy approach to the current trajectory estimate $\boldsymbol{x}(\tau_m;\boldsymbol{z}^{(k)})$, using the *exact* derivatives of the *estimated* trajectory.

We initialize with a dense support by setting $S^{(0)}$ to be all available indices of $\boldsymbol{\theta}$, and alternate these two steps until the support $S^{(k)}$ stabilizes, i.e. $S^{(k)} = S^{(k-1)}$, which corresponds to a fixed point. In most of our examples, we take $\mathcal{S}$ to be the result of sequentially thresholded ridge regression [11], which is also the choice taken in [21, 88] and makes our algorithm template match their ADO approach. Options for sparsifiers are discussed further in Section 3.3.

3.2 Smooth Nonlinear Least Squares

To solve the smooth subproblems arising in the active-set refinement eq. (30), we use an LM method, which is a standard and well studied algorithm for nonlinear least squares[2, 57, 62, 98]. For notational convenience, stack the variables $\boldsymbol{z}, \boldsymbol{\theta}$ and restrict to the current support:

$$\boldsymbol{\eta} = \begin{bmatrix} \mathrm{vec}(\boldsymbol{z}) \\ \mathrm{vec}(\boldsymbol{\theta}_{S^{(k)}}) \end{bmatrix}, \qquad \boldsymbol{G}(\boldsymbol{\eta}) = \boldsymbol{F}(\boldsymbol{z}, \boldsymbol{\theta}). \tag{32}$$

The subproblem in eq. (30) takes the form

$$\min_{\boldsymbol{\eta}} \mathcal{J}(\boldsymbol{\eta}) := \frac{1}{2}\big\|\boldsymbol{G}(\boldsymbol{\eta})\big\|^2 + \frac{1}{2}\boldsymbol{\eta}^\top \boldsymbol{Q}\boldsymbol{\eta}, \tag{33}$$

where we define the positive semi-definite matrix

$$\boldsymbol{Q} := \begin{bmatrix} \lambda\,(I_d \otimes \boldsymbol{K}) & 0 \\ 0 & \mu\, I_{|S^{(k)}|} \end{bmatrix}, \;\; \text{so that } \frac{1}{2}\,\boldsymbol{\eta}^\top \boldsymbol{Q}\,\boldsymbol{\eta} = \frac{\lambda}{2}\,\mathrm{tr}(\boldsymbol{z}^\top \boldsymbol{K}\boldsymbol{z}) + \frac{\mu}{2}\,\|\boldsymbol{\theta}\|_F^2.$$

Note that we have dropped rows/columns for inactive coefficients, i.e., we only keep $\boldsymbol{\theta}_{S^{(k)}}$.

Let $\boldsymbol{M}$ be a positive definite matrix defining a step metric. In our case, we set $\boldsymbol{M} = \boldsymbol{Q}$. LM linearizes inside of the least-squares term and adds a quadratic damping $\gamma^{(k)}$ to control step size and force convergence. The iteration is given by

$$\boldsymbol{\eta}^{(k+1)} = \arg\min_{\boldsymbol{\eta}} \; m^{(k)}(\boldsymbol{\eta}) + \frac{\gamma^{(k)}}{2}(\boldsymbol{\eta} - \boldsymbol{\eta}^{(k)})^\top \boldsymbol{M}(\boldsymbol{\eta} - \boldsymbol{\eta}^{(k)}), \tag{34}$$

$$m^{(k)}(\boldsymbol{\eta}) := \frac{1}{2}\left\|\nabla \boldsymbol{G}(\boldsymbol{\eta}^{(k)})(\boldsymbol{\eta} - \boldsymbol{\eta}^{(k)}) + \boldsymbol{G}(\boldsymbol{\eta}^{(k)})\right\|^2 + \frac{1}{2}\boldsymbol{\eta}^\top \boldsymbol{Q}\boldsymbol{\eta}. \tag{35}$$

We update $\gamma^{(k)}$ adaptively. Define the *gain ratio* (observed vs. predicted decrease)

$$\rho^{(k)} = \frac{\mathcal{J}(\boldsymbol{\eta}^{(k)}) - \mathcal{J}(\boldsymbol{\eta}^{(k+1)})}{\mathcal{J}(\boldsymbol{\eta}^{(k)}) - m^{(k)}(\boldsymbol{\eta}^{(k+1)})}. \tag{36}$$

If $\rho^{(k)} < 0.01$, we reject the step and increase $\gamma^{(k)}$; otherwise we accept and update $\gamma^{(k)}$ for the next iteration. One practical choice, analyzed in [2], is

$$\gamma^{(k+1)} = \begin{cases} \gamma^{(k)}\, c & \rho^{(k)} < 0.4, \\ \gamma^{(k)} & \rho^{(k)} \in [0.4, 0.9], \\ \gamma^{(k)}/c & \rho^{(k)} > 0.9, \end{cases} \tag{37}$$

for some $c > 1$; we use $c = 1.2$.

3.3 Sparsifying iterations

As mentioned in Section 3.1, we alternate between solving the fixed support, smooth nonlinear least squares problem (30) and updating the support of the coefficient set to obtain S^{k+1}, applying a sparse feature selection method in eq. (31). Hence, a crucial component of our algorithm is the choice of regularizer or feature selection procedure in eq. (31). The work that introduced SINDy [11] suggested ridge regularized sequentially thresholded least squares (STLSQ). This alternates between solving an ridge regularized least squares problem, and evicting features whose coefficients fall below a given threshold. Another approach is to add regularization which promotes sparsity. These include LASSO

[92] which regularizes with an ℓ_1 penalty, SR3[17, 102] which minimizes smoothed versions of nonsmooth penalties, and MIOSR [7], which directly solves ℓ_0 constrained problems using mixed integer programming. Ensembling the approaches above via bagging and feature subsampling can sometimes improve the performance of such algorithms [25], and provide some uncertainty quantification [29].

An alternative approach to feature selection is to define an appropriate sparsity promoting prior distribution and likelihood, and approximately sample from the posterior. These include spike-and-slab priors [30] and the regularized horseshoe prior [38].

In most of our experiments, we apply ridge regularized STLSQ as it empirically performed the best, and was easy to tune. In section 4.5.2 we applied SR3 [102] with an L1 penalty–we found this to be more consistent on varying trajectory lengths and noise levels without any retuning, but it can difficult to intuit good parameters a priori. In certain cases, including ensembling within our sparsification procedure greatly improved the robustness of our method, see section 4.5.1. At the cost of losing a straightforward variational problem, considering an algorithmic sparsification framework allows the specific choice of sparsifier to become an implementation detail that is easy for users to modify and provides substantial flexibility.

4 Numerical Results

We validate JSINDy on a range of examples that illustrate its capability in handling noisy, scarce and partial observations, higher order systems, and model misspecification.

For each example, we generate a true trajectory $\boldsymbol{x}(t)$ solving the ODE $\frac{d^p}{dt^p}\boldsymbol{x}(t) = f(D[\boldsymbol{x}], \boldsymbol{\theta})$, and take noisy measurements $\boldsymbol{y}_n = \boldsymbol{V_n}^\top \boldsymbol{x}(t_n) + \boldsymbol{\epsilon}_n$ by adding Gaussian noise $\boldsymbol{\epsilon}_n$. Fitting our model results in state estimates $\widehat{\boldsymbol{x}}(t)$, and system coefficients $\widehat{\boldsymbol{\theta}}$. We evaluate the state estimates, $\widehat{\boldsymbol{x}}(t)$ against the true trajectory $\boldsymbol{x}(t)$ using root mean square error (RMSE) averaged across coordinates, and normalized by the variance across time of each coordinate.

$$\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x})^2 := \frac{1}{T \cdot d} \sum_{m=1}^{d} \frac{\int_0^T (\widehat{x}_m(t) - x_m(t))^2 dt}{\int_0^T (\overline{x_m} - x_m(t))^2 dt}, \quad \overline{x_m} := \frac{1}{T}\int_0^T x_m(s) ds \tag{38}$$

Where appropriate, we evaluate the fitted coefficient matrix $\widehat{\boldsymbol{\theta}}$ using the normalized Frobenius norm compared to the true coefficients $\boldsymbol{\theta}$.

$$\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) := \frac{\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_F}{\|\boldsymbol{\theta}\|_F} \tag{39}$$

This is not relevant in Section 4.2 due to unidentifiability, and in Section 4.4, as there is no "true" coefficient matrix. All models, for each experiment, use a sum of a scaled constant kernel and Matèrn kernel with $\nu = 11/2$, which corresponds to five-times differentiable functions.

We test our algorithm on a variety of systems and observation patterns. We start with the Lorenz 63 system, which is commonly used for testing methods for identifying nonlinear dynamics, using a relatively standard sampling frequency and noise level. Next, we consider the regime of scarce sampling, with sampling frequencies far below what have been previously considered in the literature, working with the Lorenz and Lotke–Volterra dynamics. We then consider problems with partial observations, where we never observe the entire state at once (even with noise), with cases of alternating between observing the different coordinates, and observing only a subset. Finally, we consider the related problem of identifying higher order ODEs from only observations of the state variable, but not its derivatives, working with the Van der Pol system, a higher dimensional coupled version of the Duffing system, and the nonlinear pendulum under model miss-specification. The problem of

identifying higher order ODEs from point observations can can be seen as a version of a partially observed problem with additional structure between dimensions if reduced to first order.

In all of these examples, we find that the assumption that $\boldsymbol{x}(t)$ satisfies *some* autonomous ODE is a strong prior that greatly improves inference, even when the ODE must itself be identified. Identification of the ODE itself is further facilitated by the assumption of sparsity in the dynamics which significantly restricts the class of models under consideration. The benefits of a joint approach to learning as opposed to two-step estimation have been observed in many works [21, 27, 39, 41, 44, 88]. The problem of data scarcity was considered in [21, 39, 41]. The SIDDS algorithm of [39] found that such an approach can reach the Cramer-Rao lower bound and demonstrated the necessity of decoupling the discretization of the ODE from the grid of observations in order to reduce bias when the sampling period is large. Our work expands the flexibility to even scarcer data and partial observations.

We largely use the same hyperparameters throughout. We set the trajectory regularization $\lambda = 10^{-3}$, the data weight $\alpha = 1$, the collocation penalty $\beta = 10^5$, and use 500 uniformly spaced collocation points with uniform weight. Except for Section 4.5 where we apply SR3, we use STLSQ as a sparsifier, with thresholds and ridge regularization parameters chosen for each problem. See Table 1 for full details of hyperparameters and problem parameters.

Additionally, we include two systematic experiments: one varying the time horizon and noise level on the Lorenz system, and another varying the noise level on a nonlinear damped oscillator system. These show promising empirical results compared to other competitive models.

Experiment	**Noise abs**	**Noise rel (%)**	**Order**	$\boldsymbol{\Delta t}$	$\boldsymbol{t_1}$	**Obs.**	**Obs. Pattern**	**Thr**	**Ridge**	$\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x})$	$\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta})$
Lorenz	2.0	24.60	1	0.05	10	200 (×3)	Full	0.5	0.01	2.35×10^{-2}	3.97×10^{-2}
Scarce Lorenz	0.0	0.00	1	0.5	10	21 (×3)	Full	0.5	0.01	1.61×10^{-3}	2.36×10^{-3}
Lotka–Volterra	0.2	19.68	1	3	50	17 (×2)	Full	0.1	0.1	1.13×10^{-2}	3.32×10^{-2}
Lorenz Streams	0.32	6.92	1	0.025	10	404	One at a time	0.25	0.01	5.69×10^{-3}	1.96×10^{-2}
Lorenz Unobserved	0.32	3.98	1	0.025	10	404	3rd coord. unobs.	0.1	0.01	6.84×10^{-3}	N/A
Van der Pol	0.1	7.39	2	2	100	50	Pos.-only	0.1	0.01	4.54×10^{-2}	1.86×10^{-1}
Coupled Duffing	0.2	16.85	2	0.2	30	150 (×5)	Pos.-only	0.1	0.01	3.55×10^{-2}	3.63×10^{-2}
Nonlin. pend. (linear)	0.1	4.17	2	1	100	100	Pos.-only	0.1	0.01	1.25×10^{-1}	N/A
Nonlin. pend. (cubic)	0.1	4.17	2	1	100	100	Pos.-only	0.1	0.01	2.22×10^{-2}	N/A
Damped nonlin. osc.	0.32	38.22	1	0.05	10	200 (×2)	Full	0.08	100.00	6.60×10^{-2}	2.47×10^{-1}

Table 1: Summary of experiments. The $n(\times d)$ indicates that the observations were made at n time points of the d dimensional state. **Thr.** and **Ridge** are the threshold and ridge regularization strength used in the STLSQ sparsifier.

Lorenz System. As a first example which is commonly used to test methods for identifying nonlinear dynamics, we considered the Lorenz system:

$$\begin{aligned} \frac{dx_1}{dt} &= -10x_1 + 10x_2, \\ \frac{dx_2}{dt} &= 28x_1 - x_2 - x_1x_3, \\ \frac{dx_3}{dt} &= -\frac{8}{3}x_3 + x_1x_2. \end{aligned} \tag{40}$$

We use the initial condition $\boldsymbol{x}(0) = (-8, 8, 27)$ and took full observations with a sampling period $\Delta t = 0.05$, and Gaussian noise with variance $\sigma^2 = 4$.

Our approach obtained the equations

$$\begin{aligned} \frac{dx_1}{dt} &= -9.607x_1 + 9.689x_2 \\ \frac{dx_2}{dt} &= 29.122x_1 - 1.239x_2 - 1.033x_1x_3 \\ \frac{dx_3}{dt} &= -2.624x_3 + 0.999x_1x_2. \end{aligned} \tag{41}$$

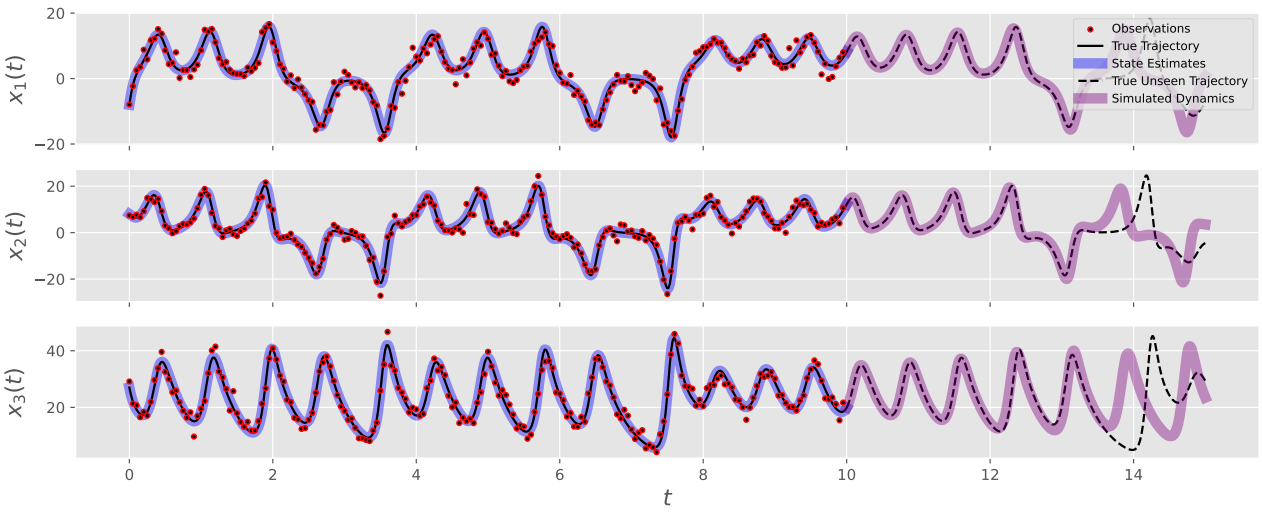

Figure 1: Lorenz 63 system with observations sampled at $\Delta t = 0.05$ until $t = 10$ with added noise set to $\sigma^2 = 4$. Simulated dynamics from eq. (41) are compared against true dynamics from $t = 10$ until $t = 15$.

with $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 2.353 \cdot 10^{-2}$ and $\text{RMSE}_{\text{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 3.975 \cdot 10^{-2}$. Results can be seen in Figure 1. This is essentially a baseline example. For this problem, the sampling rate was at the lower end of the standard range that is considered in the literature. Our approach both identified the correct active features, and approximated the coefficient values to reasonable accuracy. In the rest of our numerical examples, we largely consider examples where traditional two-step methods are not applicable, or will obviously break down.

4.1 Scarce observations

We now consider cases of very scarcely sampled data with the Lorenz 63 and the Lotka-Volterra systems.

4.1.1 Lorenz 63 Scarce

We modified the previous example, instead sampling 21 observations of the entire state over the interval $[0, 10]$ with a sampling period of $\Delta t = 0.5$, and no noise. Results can be seen in Figure 2 with the following learned system below:

$$\begin{aligned} \frac{dx_1}{dt} &= -10.008\, x_1 + 10.006\, x_2 \\ \frac{dx_2}{dt} &= 28.073\, x_1 - 1.012\, x_2 - 1.003\, x_1 x_3 \\ \frac{dx_3}{dt} &= -2.663\, x_3 + 1.001\, x_1 x_2 \end{aligned} \tag{42}$$

with $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 1.608 \cdot 10^{-3}$ and $\text{RMSE}_{\text{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 2.356 \cdot 10^{-3}$. These results are very accurate, owing to the lack of noise. Indeed, a parameter counting argument suggests that once the support is identified, this problem is essnetially a noiseless and well-posed though ill-conditioned inversion problem, where we should expect good algorithms to attain very high accuracy.

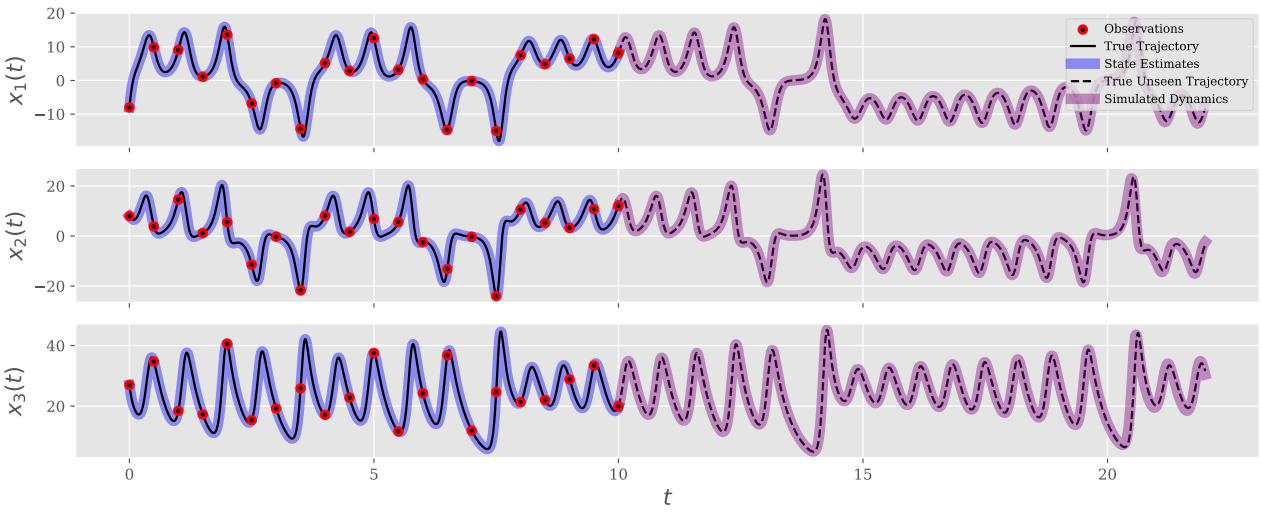

Figure 2: Lorenz 63 system with observations sampled at $\Delta t = 0.5$ until $t = 10$ with no added noise. Simulated dynamics of (42) are plotted against true, unseen, trajectory from $t = 10$ to $t = 25$.

4.1.2 Lotka Volterra

The Lotka-Volterra system is a nonlinear oscillator with periodic solutions, describing population dynamics of a predator-prey ecology, and is a common benchmark for system identification[7, 25, 44, 87]. This system governs the populations of prey x_1 and predators y_1:

$$\begin{aligned} \frac{dx_1}{dt} &= p_{11}x_1 - p_{12}x_1x_2, \\ \frac{dx_2}{dt} &= -p_{21}x_2 + p_{22}x_1x_2, \end{aligned} \tag{43}$$

with parameters $p_{11} = p_{12} = p_{21} = 1.$, and $p_{22} = 0.5$. We again considered scarce sampling, with approximately 2 observations per cycle, taken at a sampling period of $\Delta t = 3$ in the interval $t \in [0, 50]$. Gaussian noise with variance $\sigma^2 = 0.04$ was added to the observations. We set the feature library to include monomials up to second order:

$$\Phi = \left\{1, x_1, x_2, x_1^2, x_1x_2, x_2^2\right\}.$$

Our approach obtained the system

$$\begin{aligned} \frac{dx_1}{dt} &= 1.009x_1 - 1.056x_1x_2 \\ \frac{dx_2}{dt} &= -0.980x_2 + 0.500x_1x_2. \end{aligned} \tag{44}$$

with $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 1.131 \cdot 10^{-1}$ and $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 3.323 \cdot 10^{-2}$. We also compare the phase portrait of the learned model to the true dynamics, simulating multiple trajectories from various initial conditions in Figure 4.

4.2 Partial Observations

We now consider problems with only partial observations. Rather than observing the entire state $\boldsymbol{x}(t_n)$, we observe a sequence of linear measurements of the state, $\boldsymbol{y}_n = \boldsymbol{V}_n^\top \boldsymbol{x}(t_n)$. We test our algorithm on the Lorenz dynamics with two challenging observation patterns. In these examples, we took each $\boldsymbol{V}_n$ to be a standard basis vector, observing one coordinate at a time. In both of these examples, we took the same initial conditions as above, set the sampling period $\Delta t = 0.025$, and set the noise $\sigma^2 = 0.1$.

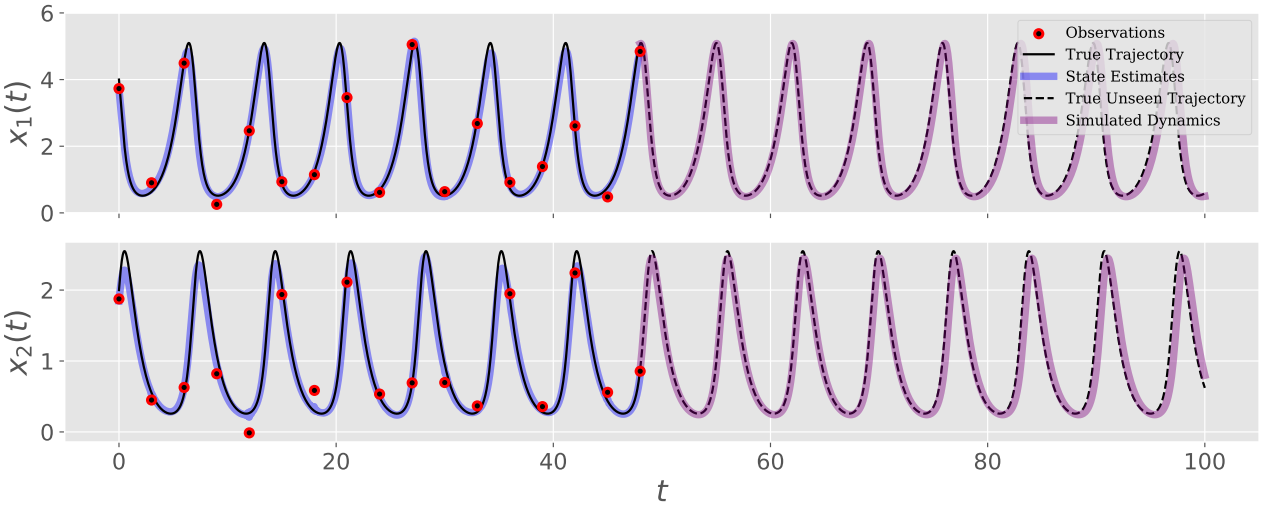

Figure 3: Results for the Lotke-Volterra system of section 4.1.2. Samples are taken at the rate $\Delta t = 3$ up to $t = 48$, including additive noise with $\sigma = 0.2$. Simulated dynamics are shown up to $t = 100$, comparing the using the results from (44) with the true dynamics.

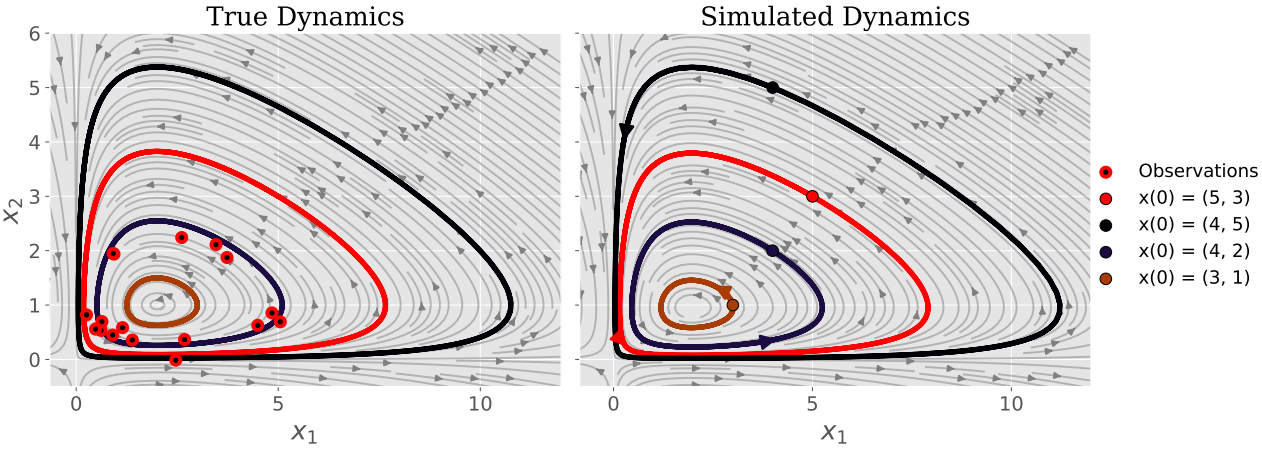

Figure 4: Left: True Lotka-Volterra plotted from multiple initial conditions on phase portrait using (43). Red trajectory and measurements are those from Figure 3. Right: Using learned model (44) to simulate same trajectories provided same initial conditions.

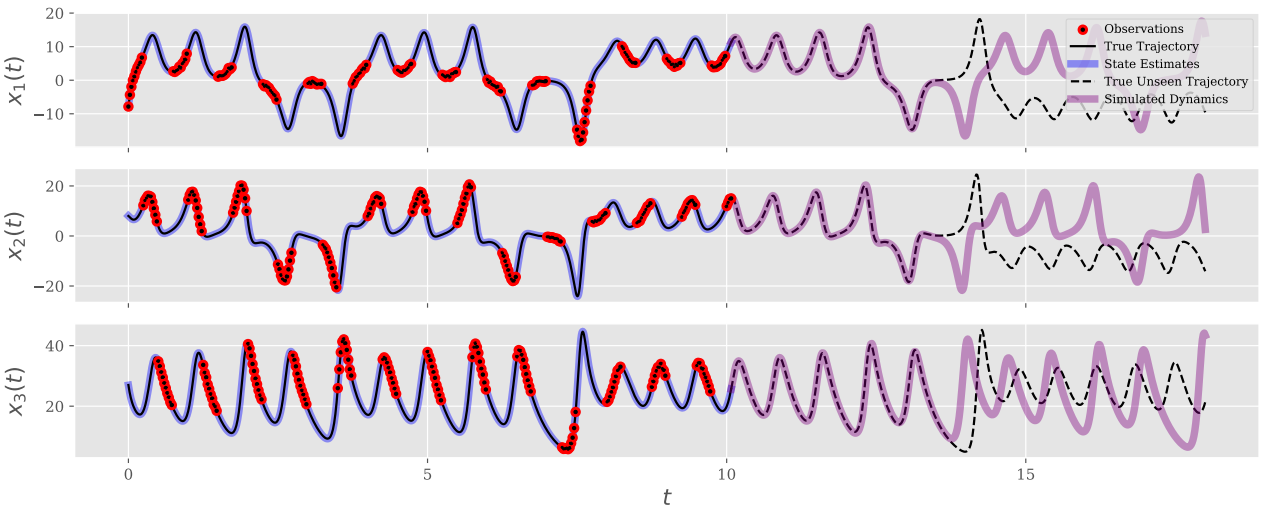

Figure 5: Results for Lorenz 63 system with streams of partial observations. Observations alternative between each coordinate every 10 samples at sampling rate $\Delta t = 0.025$ up to $t = 10$ with added noise set to $\sigma^2 = 0.1$. Learned dynamics are simulated from $t = 10$ until $t = 20$.

4.2.1 Alternating coordinates

In the first case, we alternated between periods of observing each of the three coordinates one at a time for 10 time steps each. For each coordinate, this leads to relatively long gaps (of duration $t = 0.5$) where no observations of the state are made. However, our approach allows the dynamics to effectively fill in this missing information, correlating the data available between the different coordinates. Notably, if the dynamics were *known ahead of time*, then applying a high order nonlinear Kalman smoothing method could be similarly effective, but this is not applicable here, since there is no a priori knowledge of the governing equations [86]. Our model learned the system in (45) with $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 5.692 \cdot 10^{-3}$ and $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 1.961 \cdot 10^{-2}$. Results can be seen in Figure 5.

$$\begin{aligned} \frac{dx_1}{dt} &= -9.987\, x_1 + 9.951\, x_2, \\ \frac{dx_2}{dt} &= 28.104\, x_1 - 1.030\, x_2 - 1.003\, x_1 x_3, \\ \frac{dx_3}{dt} &= -0.606\, \cdot 1 - 2.642\, x_3 + 1.005\, x_1 x_2. \end{aligned} \tag{45}$$

4.2.2 Hidden coordinates

We now consider the problem of identifying dynamics with a hidden variable that is never observed. In this case, the observations alternate between the first two coordinates on each time step, and the third is never observed. A similar problem have been considered in [36] and [77]. [36] approach this using the lens of computational graph completion [69] and kernel learning, though the data was without noise, and the model was considered in discrete time. [77]. In this case, we cannot expect to recover the values of the hidden third coordinate due to strong identifiability issues. For example, rescaling the third coordinate by an arbitrary scalar, and modifying the first two equations appropriately gives rise to an infinite family of equations of the same sparsity level which reproduce exactly the original dynamics, restricted to the first two coordinates.

Nevertheless, we can learn an extra latent coordinate; when combined with the first two observed coordinates, these together satisfy a modified ODE that reproduces the dynamics on the observed

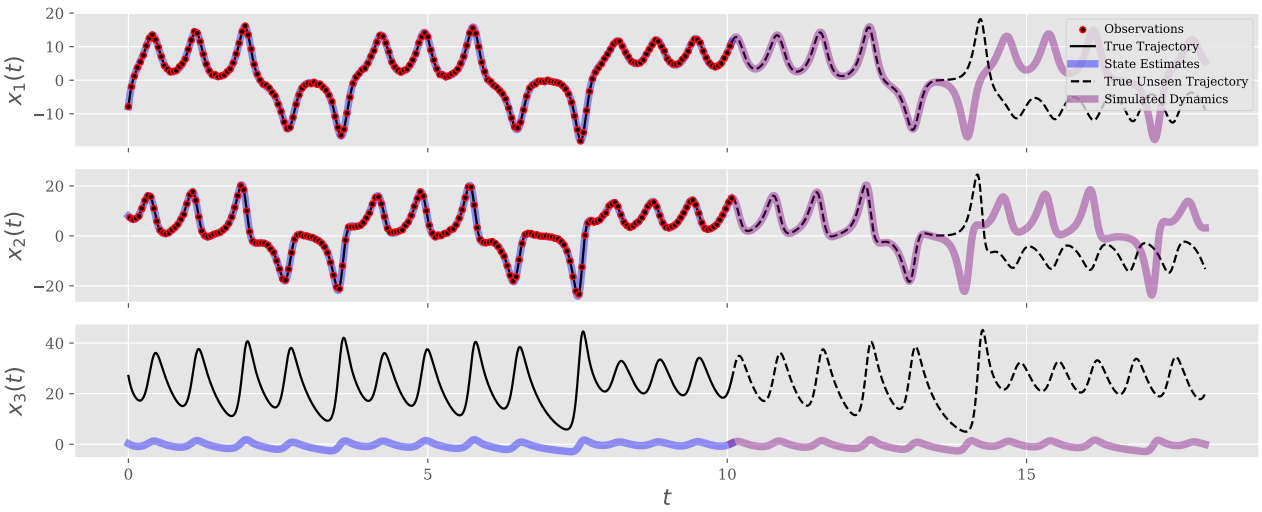

Figure 6: Results for Lorenz 63 system with an unobserved coordinate. Observations alternate between first two coordinates at sampling rate $\Delta t = 0.025$ up to $t = 10$, with no knowledge of the third, except that the system is known to be three dimensional. Our algorithm constructs a third, latent coordinate and corresponding dynamics which, when evolved together with the first two coordinates, reproduces the the observed dynamics. Further, we we can effectively extrapolate the time series forward by using our learned dynamics and imputing the value of our artificial coordinate as a new initial condition, generating good predictions for the first two coordinates.

coordinates. This is demonstrated by taking our state estimate at $t = 10$, the last observed datapoint, and simulating the learned ODE forward in time to predict the evolution of the first two states. Doing so produces accurate trajectories for four additional cycles, including a lobe switch, before diverging from the true trajectory due to chaos. Note that modeling only the first two states with an ODE is insufficient–the flow would be self-intersecting and cannot satisfy any autonomous ODE without a third coordinate (no chaos in two dimensions). The results can be seen in Figure 6 where the following syste is learned in eq. (46). Because the true hidden state is unidentifiable, we compute the state estimation error restricted to the first two coordinates and obtain $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 6.840 \cdot 10^{-3}$. Similarly, since we cannot expect to recover the third coordinate, we omit the coefficient error.

$$\begin{aligned}
\frac{dx_1}{dt} &= 0.722 \cdot 1 - 9.969\, x_1 + 9.936\, x_2 + 0.430\, x_3, \\
\frac{dx_2}{dt} &= -0.456 \cdot 1 - 0.453\, x_1 - 0.979\, x_2 - 0.316\, x_3 + 0.408\, x_1 x_2 - 8.832\, x_1 x_3 + 0.161\, x_3^2, \\
\frac{dx_3}{dt} &= -8.626 \cdot 1 + 0.057\, x_2 - 2.683\, x_3 + 0.133\, x_1 x_2 - 0.421\, x_1 x_3.
\end{aligned} \tag{46}$$

Traditional approaches to handling partially observed systems and modeling time series often enrich the dynamics by inserting time delay coordinates [6, 10, 12, 31, 53, 71, 90]. However, that comes with its own challenges, such as selecting the number and spacing of delay coordinates, difficulties in continuous time, especially when observations are not evenly spaced, and introducing bias due to possibly noisy inputs. Our approach has its own drawback of requiring a separate state estimation problem to make predictions, and producing a somewhat uninterpretable third coordinate. We remark that it is possible to include time delay embeddings in our approach as well, essentially turning our collocation method for differential equations into a collocation approach for a *delay* differential equations–we leave this to future work.

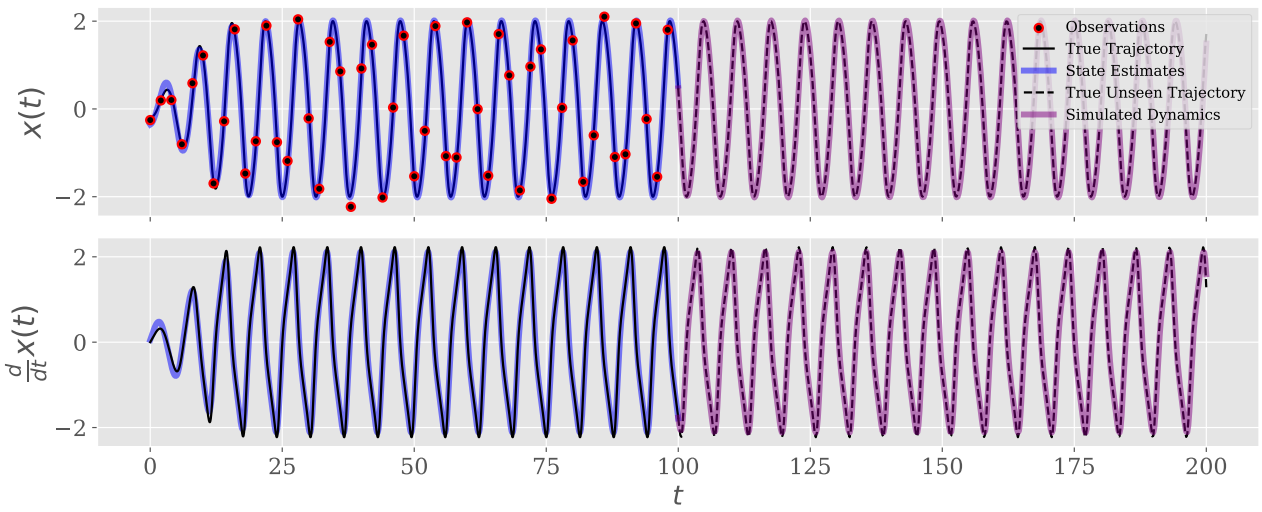

Figure 7: Results for the Van der Pol oscillator. Sampling is done only on system state, $\boldsymbol{x}$, with sampling rate $\Delta t = 2$ up to $t = 100$ with simulated dynamics from $t = 100$ to $t = 200$ using the learned equations from eq. (48).

4.3 Higher order systems

Our approach is also capable of learning higher order differential equations given only state measurements. While devices exist that can directly measure first and second order derivatives (e.g. Doppler radar, gyroscopes, accelerometers), measurements of the system state are the most common, and second order laws are ubiquitous in physics. We consider the Van der Pol oscillator, a coupled version of the Duffing system, and the nonlinear pendulum under model misspecification.

4.3.1 Van der Pol Oscillator

The Van der Pol oscillator is governed by the second-order differential equation

$$\frac{d^2x}{dt^2} = \mu(1 - x^2)\frac{dx}{dt} - x. \tag{47}$$

We set $\mu = 0.5$ and note that for $\mu > 0$, Van der Pol systems demonstrate an attractive limit cycle with fast and slow branches. We trained on equally sampled measurements with $\Delta t = 2$ for $t \in [0, 100]$ with additive Gaussian noise with variance $\sigma^2 = 0.1$. We took the feature library to contain cubic polynomials of x and $\frac{d}{dt}x$. Our approach obtained the equation

$$\frac{d^2x}{dt^2} = -0.982x + 0.343\frac{dx}{dt} - 0.337\frac{dx}{dt}x^2 \tag{48}$$

with coefficient error $\text{RMSE}_{\text{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 1.858 \cdot 10^{-1}$, and state estimation error $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 4.540 \cdot 10^{-2}$. Our algorithm identifies nearly the correct equation with exactly the correct support.

4.3.2 Coupled Duffing Oscillators

As an example of a larger second order ODE, we consider a ring of d Duffing oscillators with nearest-neighbor coupling. For each coordinate $i = 1, \ldots, d$, $x_i(t)$ satisfies the differential equation

$$\frac{d^2}{dt^2}x_i(t) = -\alpha_i\, x_i(t) - \beta_i\, x_i(t)^3 + \mu\left(x_{i-1}(t) - 2x_i(t) + x_{i+1}(t)\right), \tag{49}$$

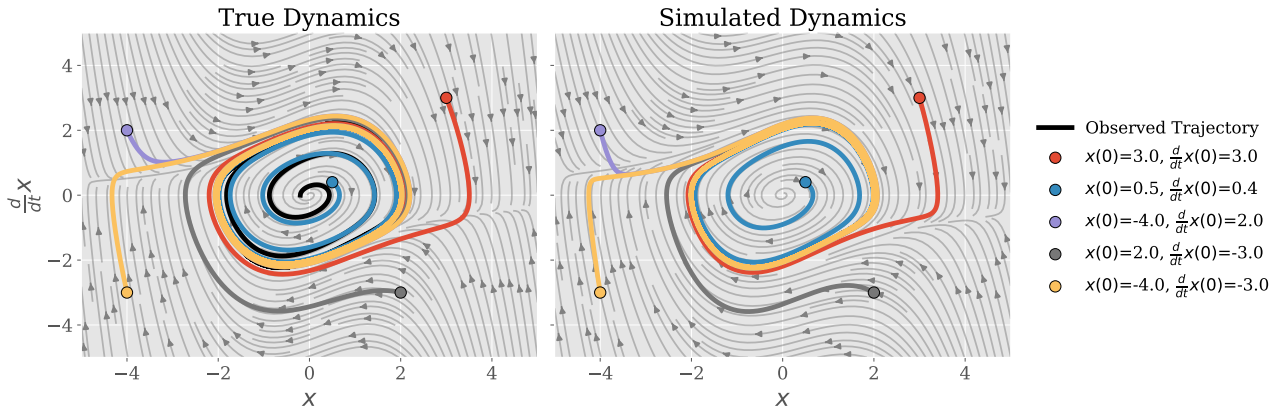

Figure 8: Left: True Von der Pol oscillator trajectories plotted from multiple initial conditions on phrase portrait using eq. (47). Right: Using learned model eq. (48) to simulate same trajectories provided same initial conditions.

with periodic indexing $x_{d+1} := x_1$. These dynamics are Hamiltonian. Set $\boldsymbol{v} := \frac{d}{dt}\boldsymbol{x}$, and

$$\mathcal{H}(\boldsymbol{x}, \boldsymbol{v}) = \sum_{i=1}^{d} \frac{1}{2} v_i^2 + \sum_{i=1}^{d} \left(\frac{\alpha_i}{2} x_i^2 + \frac{\beta_i}{4} x_i^4 \right) + \frac{\mu}{2} \sum_{i=1}^{d} (x_{i+1} - x_i)^2 , \tag{50}$$

the dynamics satisfy

$$\frac{d}{dt}\boldsymbol{x} = \nabla_{\boldsymbol{v}} \mathcal{H}(\boldsymbol{x}, \boldsymbol{v}), \quad \frac{d}{dt}\boldsymbol{v} = -\nabla_{\boldsymbol{x}} \mathcal{H}(\boldsymbol{x}, \boldsymbol{v}) \tag{51}$$

We remark that the nonlinearity has a significant qualitative effect in the dynamics. If we consider a linear version where we drop the cubic restoring force, the dynamics would be governed by the matrix $M := \mathrm{Diag}((\alpha_i)_{i=1}^n) + \mu K$, where K is the graph Laplacian on a cycle. Taking $\mu < 0$ causes the coupling term to encourage exponential blow up, and with μ negative enough relative to α_i, M will have both positive and negative eigenvalues. This implies unbounded level sets in the Hamiltonian, and divergent trajectories in the linear version. However, the positive cubic restoring terms ($\beta_i > 0$) are quartic in the Hamiltonian, which guarantees compact level sets, bounded trajectories [91].

We considered an example in $d = 5$ dimensions with parameters $\mu = -2$, $\alpha_i \overset{\text{i.i.d.}}{\sim} \mathrm{Unif}[2, 4]$, $\beta_i = 2$. We initialized the system with all coordinates at the same position and velocity, $x_i(0) = 0$ and $\frac{d}{dt}x_i(0) = 0$ for $i = 1, \ldots, d$. We took equally spaced measurements of the position values $q_i(t)$ with a sampling period $\Delta t = 0.2$ between $t = 0$ and $t = 30$ as the input data, and include additive Gaussian noise of variance $\sigma^2 = 0.04$ which corresponds to a signal to noise ratio of 39.2.

Trajectory regularization, data weight, collocation penalties and points remain as in the previous example $\lambda = 10^{-3}, \alpha = 1, \beta = 10^5, m = 1000$. We took the sparsifier to be STLSQ with threshold 0.1, and ridge regularization parameter 0.01. The feature library includes monomials up to cubic order of q_i and $\dot{q}_i$, but excludes interaction terms,

$$\Phi = \left\{ 1, x_1, \frac{d}{dt}x_1, ..., x_5, \frac{d}{dt}x_5, q_0^2, \dot{q}_0^2, \ldots, q_4^3, \dot{q}_4{}^3 \right\}. \tag{52}$$

This library contains 31 functions (down from 286 if we were to include all 10-variable polynomials up to cubic order). Our algorithm correctly identifies the correct support in all five variables which amounts to finding the 20 active variables out of 155 with a maximum coefficient error of 0.12.

We show the learned equations on the left, and the true equations on the right.

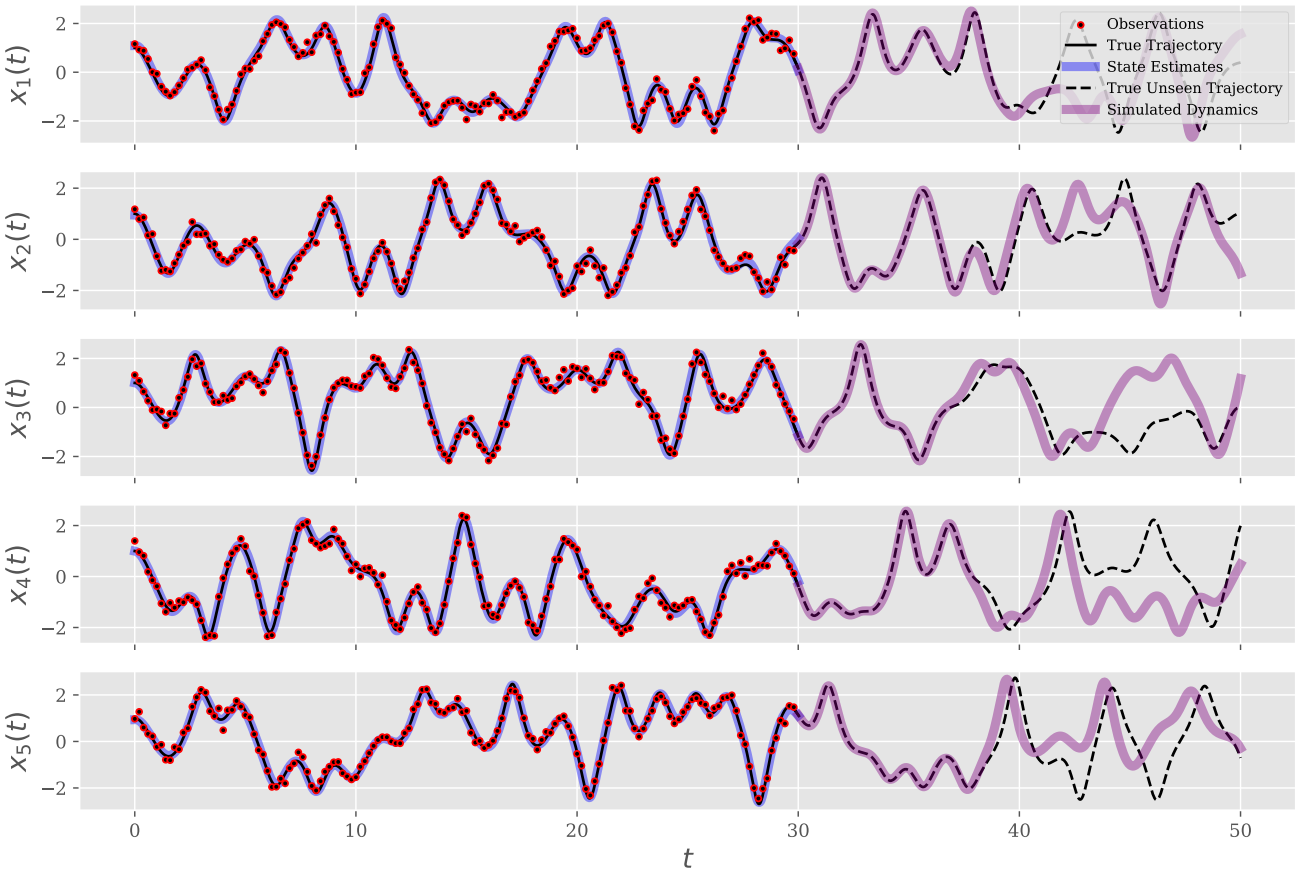

Figure 9: Position observations sampled at rate $\Delta t = 0.2$ up to $t = 30$ with added noise set to $\sigma^2 = 0.04$ equating to a relative noise of 39.2%. Learned and true dynamics from eq. (53) are simulated from $t = 30$ to $t = 50$.

$$
\begin{aligned}
\frac{d^2x_1}{dt^2} &= 1.272\,x_1 - 2.074\,x_2 - 2.083\,x_5 - 1.890\,x_1^3, & \frac{d^2x_1}{dt^2} &= 1.392\,x_1 - 2\,x_2 - 2\,x_5 - 2\,x_1^3 \\
\frac{d^2x_2}{dt^2} &= -1.951\,x_1 + 0.695\,x_2 - 2.010\,x_3 - 1.977\,x_2^3, & \frac{d^2x_2}{dt^2} &= -2\,x_1 + 0.711\,x_2 - 2\,x_3 - 2\,x_2^3 \\
\frac{d^2x_3}{dt^2} &= -2.016\,x_2 + 1.862\,x_3 - 1.956\,x_4 - 2.045\,x_3^3, & \frac{d^2x_3}{dt^2} &= -2\,x_2 + 1.81\,x_3 - 2\,x_4 - 2\,x_3^3 \\
\frac{d^2x_4}{dt^2} &= -2.023\,x_3 + 1.679\,x_4 - 2.072\,x_5 - 2.019\,x_4^3, & \frac{d^2x_4}{dt^2} &= -2\,x_3 + 1.521\,x_4 - 2\,x_5 - 2\,x_4^3 \\
\frac{d^2x_5}{dt^2} &= -1.921\,x_1 - 1.961\,x_4 + 1.925\,x_5 - 2.054\,x_5^3, & \frac{d^2x_5}{dt^2} &= -2\,x_1 - 2\,x_4 + 1.988\,x_5 - 2\,x_5^3.
\end{aligned}
\tag{53}
$$

We show estimates of the derivatives of one of the state variables and the error in the forward simulations of these learned dynamics starting from the last observation at $t = 30$ in Figure 9 with $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 3.554 \cdot 10^{-2}$ and $\text{RMSE}_{\text{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 3.627 \cdot 10^{-2}$. Our model produces very accurate derivative estimates despite only having access to noisy observations of the original state variables.

The regularized least squares collocation applied to a space of five times differentiable functions allows for derivative estimates up to fourth order to remain accurate, despite the dynamics being only second order. This level of performance is comparable to data simulation approaches where we know the model *beforehand* and apply high order nonlinear Kalman smoothing. The key point is that our approach simultaneously discovers the model and estimates the states.

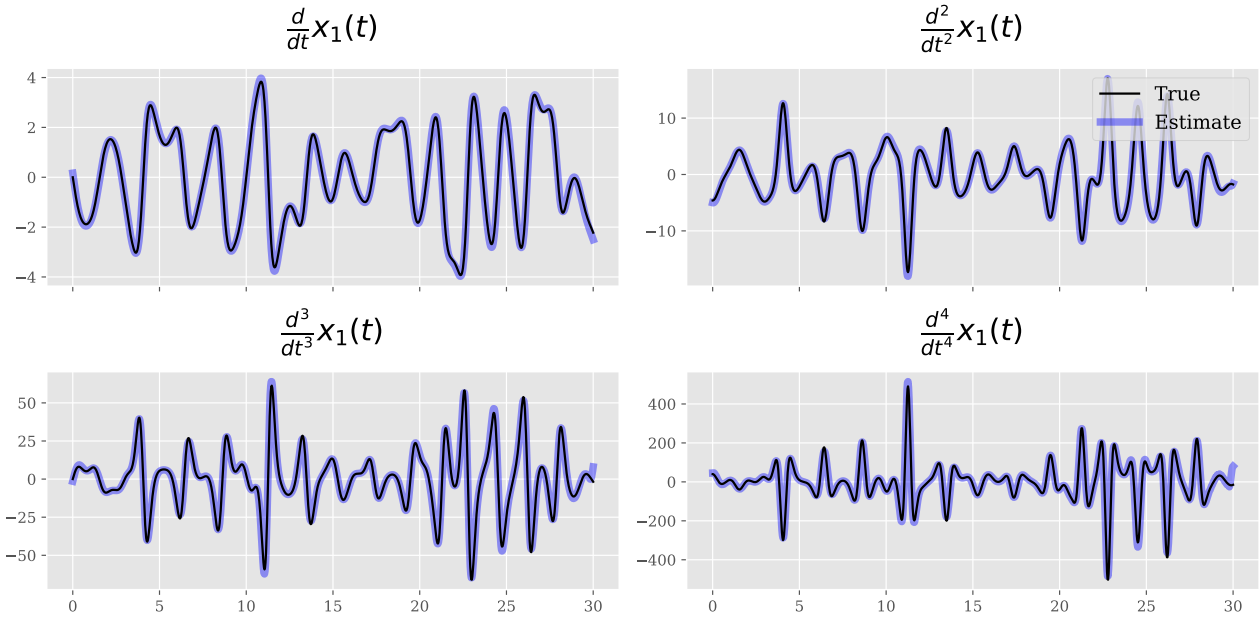

Figure 10: Results for estimating time derivatives of the coupled Duffing system. The relative mean square errors of the estimates of the first, second, third, and fourth derivatives are $2.32 \cdot 10^{-3}, 3.68 \cdot 10^{-3}, 6.57 \cdot 10^{-3}$ and $1.1 \cdot 10^{-2}$ respectively. In this case, our approach resolves the fine structure in estimating the state.

4.4 A misspecified model

We considered case where the dynamics are misspecified by fitting polynomials to nonlinear pendulum dynamics. Our least squares relaxation of the dynamics allows some of the unmodeled dynamics to be accommodated as an error term, as in eq. (61). Robustness to misspecification is very useful; in many experimental or designed processes we may have little to no prior knowledge of the true functional form of the underlying dynamics. In these cases, we hope to maintain highly accurate state estimation (despite the misspecified dynamics), while producing a reasonable model for simulating the future.

We consider the nonlinear pendulum,

$$\frac{d^2x}{dt^2} = -\sin(x), \tag{54}$$

and fit two models, one that is linear and another that is cubic. Simulated trajectories were created with initial displacement of $x(0) = 3$ and no initial velocity $\frac{d}{dt}x(0) = 0$. Observations were made with a sampling period of $\Delta t = 1$ over the interval $t \in [0, 100]$, with Gaussian noise ($\sigma^2 = 0.1$).

The relatively large initial condition $x(0) = 3$ means that the nonlinearity has a significant effect; replacing $\sin(x)$ with a cubic Taylor series centered around zero results in finite time blowup with

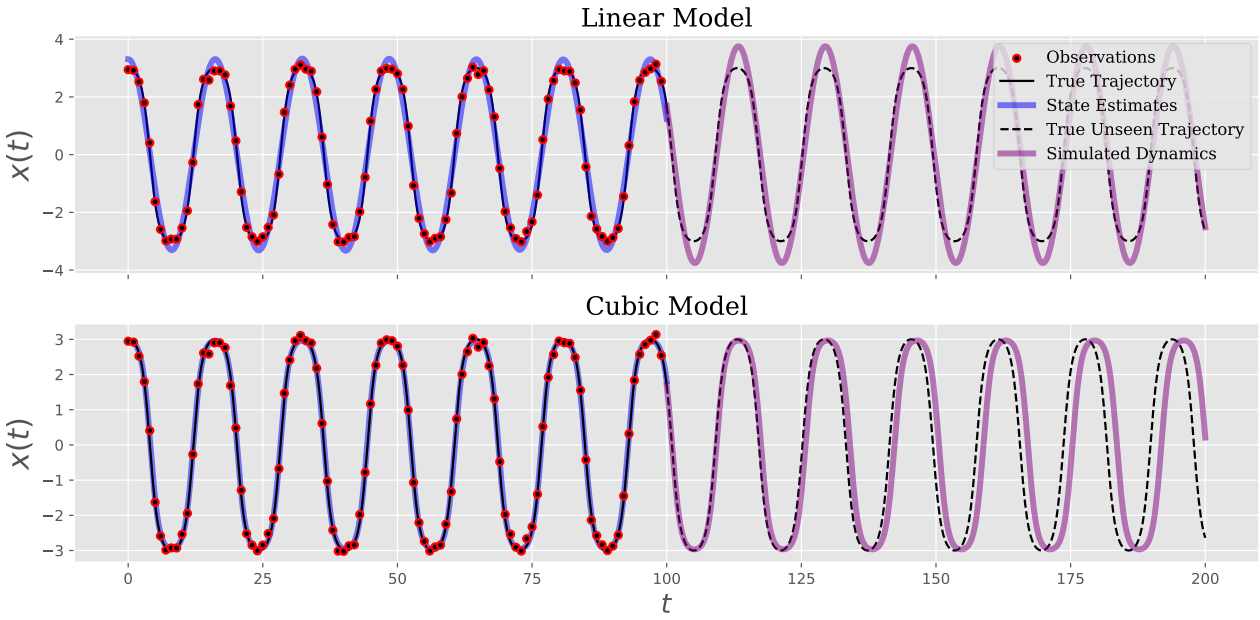

Figure 11: Results for misspecified linear and cubic models for nonlinear pendulum. Observations sampled at rate $\Delta t = 1$ up to $t = 100$ with added nosie set to $\sigma^2 = 0.01$. Learned dynamics for both linear and cubic model, eq. (55) and eq. (56), are simulated from $t = 100$ to $t = 200$ where the linear model successfully captures periodicity of the true trajectory, but fails to capture the correct amplitude, whereas this is flipped for the cubic model.

these initial conditions. We obtained the following models:

$$\text{Linear:} \quad \frac{d^2x}{dt^2} = -0.151\,x \tag{55}$$

$$\text{Cubic:} \quad \frac{d^2x}{dt^2} = -0.796\,x + 0.086\,x^3. \tag{56}$$

Both of these perform very well in state estimation, despite only forming a somewhat crude approximation to the dynamics as evaluated through forward simulation (see Figure 11). The filtering error of the the linear model was $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 1.250 \cdot 10^{-1}$ and for the the cubic model, was $\text{RMSE}_{\text{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 2.221 \cdot 10^{-2}$. The linear model matched the period of the true nonlinear dynamics eq. (54), but did not have enough degrees of freedom to match the amplitude and misrepresented the shapes of the peaks. The cubic model performed quite well, especially in state estimation and matching the shapes of the oscillations, but has a slightly incorrect period.

4.5 Benchmarking

We now consider two systematic experiments testing the robustness of our method to noise, and comparing primarily to ODR-BINDy[27].

4.5.1 Damped Nonlinear Oscillator

We considered increasing noise levels with fixed time length and observation frequency for a nonlinear damped oscillator that was studied in [27]. The dynamics are given by

$$\begin{aligned} \frac{d}{dt}x_1 &= -0.1x_1^3 + 2x_2^3 \\ \frac{d}{dt}x_2 &= 2x_1^3 - 0.1x_2^3. \end{aligned} \tag{57}$$

We took the initial conditions $\boldsymbol{x}(0) = (2, 0)$, and took samples of both variables with sampling period $\Delta t = 0.05$ for $t \in [0, 10]$. We first considered an example with noise level $\sigma = 0.32$. We set trajectory regularization $\lambda = 10^{-5}$, data weight $\alpha = 10$, collocation penalty $\beta = 10^5$, and used STLSQ with threshold to 0.08 and ridge parameter to 100. This is a surprisingly challenging example, especially with this level of noise–our model does not identify the correct support, but still performs quite well in estimating the state. We obtained $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x}) = 6.60 \times 10^{-2}$, $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta}) = 2.47 \times 10^{-1}$, and the equations

$$\begin{aligned} \frac{d}{dt}x_1 &= 0.078\, x_2 - 0.560\, x_1 x_2^2 + 1.859\, x_2^3 \\ \frac{d}{dt}x_2 &= -0.008\, x_1 - 0.212\, x_1^2 - 1.950\, x_1^3 - 0.284\, x_1 x_2^2. \end{aligned} \tag{58}$$

The oscillatory terms were recovered with reasonable accuracy, but the small damping was hard to capture correctly. From Figure 12, we can see that the state estimates and reproduced dynamics are still close, which suggests some ill-posedness in the sparse recovery problem.

Next, we tested on a sequence of noise levels $\sigma = 0.02, 0.04, 0.08, 0.16, 0.32, 0.64$ and averaged the values of $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x})$ and $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta})$ over 32 random seeds. In this case, we found that including ensembling [25] in the inner sparsifier greatly improved the consistency of the results for higher noise levels. We compared five models, weak SINDy[63], weak SINDy with ensembling, JSINDy, JSINDy with ensembling, and ODR-BINDy [25, 27]. We used PySINDy for ensembling and weak SINDy [46], and the GitHub repository [26] associated to [27] for ODR-BINDy. We also tested with other two-step variants of SINDy, but they all performed worse than weak SINDy, so we omit them. As ODR-BINDy also produces state estimates, we compare the state estimation error between JSINDy, JSINDy with ensembling and ODR-BINDy.

We note that our ensembling only occurs inside of the sparsifier $\mathcal{S}$, and therefore does not incur much additional computational cost, which is dominated by the nonlinear least squares problems rather than the sparsifying iterations. In cases where the model is already successful, ensembling can introduce additional extraneous features, but for challenging problems, it seems to improve the robustness. Perhaps surprisingly, the error of weak ensemble SINDy was quite high for low levels of noise, but did not degrade as noise was increased for the scale that we considered.

In terms of the coefficient error, and without ensembling the sparsifier, our model performs noticeably worse than ODR-BINDy; maximizing the Bayesian evidence seems to a good model selection criterion. Upon introducing ensembling, our model selection stabilizes and the coefficent errors become comparable. In terms of state estimation, JSINDy both with and without ensembling performs well across varying levels of noise, regardless of the coefficient accuracy, owing to the stabilizing effect of the RKHS regularization and least squares relaxation.

4.5.2 Lorenz 63 example

We also tested the performance of our method on varying levels of noise and durations of observations with fixed observation frequency on the Lorenz 63 dynamics (40) with results in Figure 14. We applied

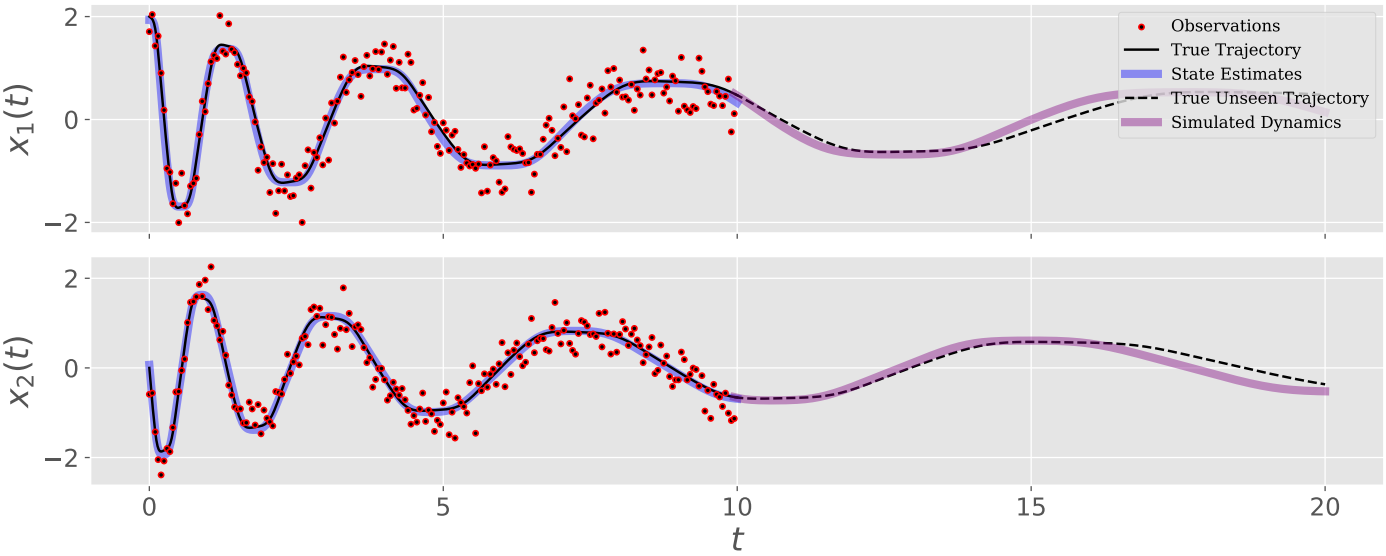

Figure 12: Results for the nonlinear damped oscillator with full state observations, $\Delta t = 0.05$ in the interval $[0, 10]$, and $\sigma = 0.32$ additive noise. While the learned equations in eq. (58) do not match the true dynamics in terms of the coefficient values, JSINDy effectively infers the states, and produces a reasonable extrapolation in forward simulation.

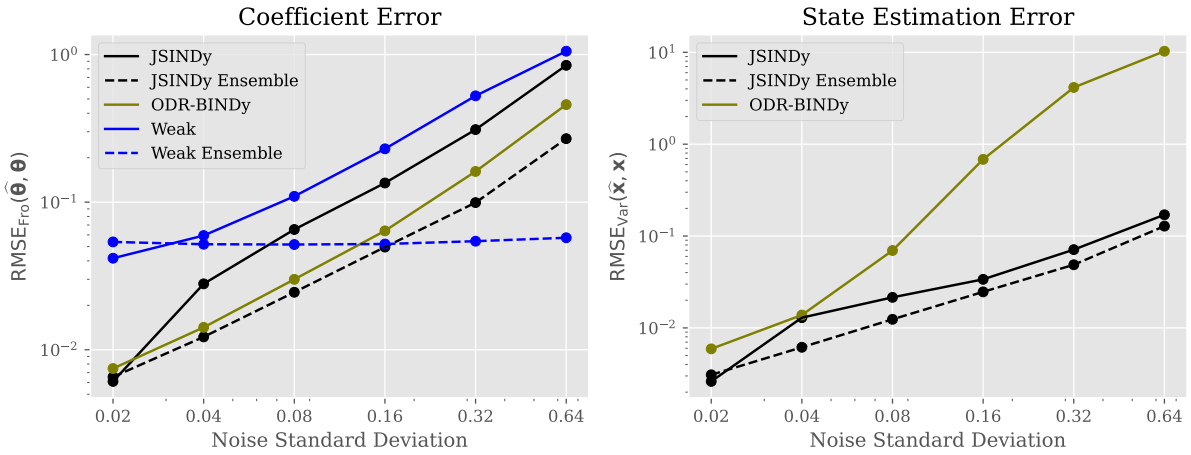

Figure 13: Results for the damped nonlinear oscillator across varying levels of noise. Left: coefficient error $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta})$, Right: state estimation error $\mathrm{RMSE}_{\mathrm{Var}}(\widehat{\boldsymbol{x}}, \boldsymbol{x})$. In this case, ensembling significantly improves the coefficient error for both weak SINDy and JSINDy. For JSINDy, including ensembling inside of the sparsifier brings the coefficient error from noticably worse than ODR-BINDy to slightly better. Overall, JSINDy performs very well in state estimation across varying levels of noise.

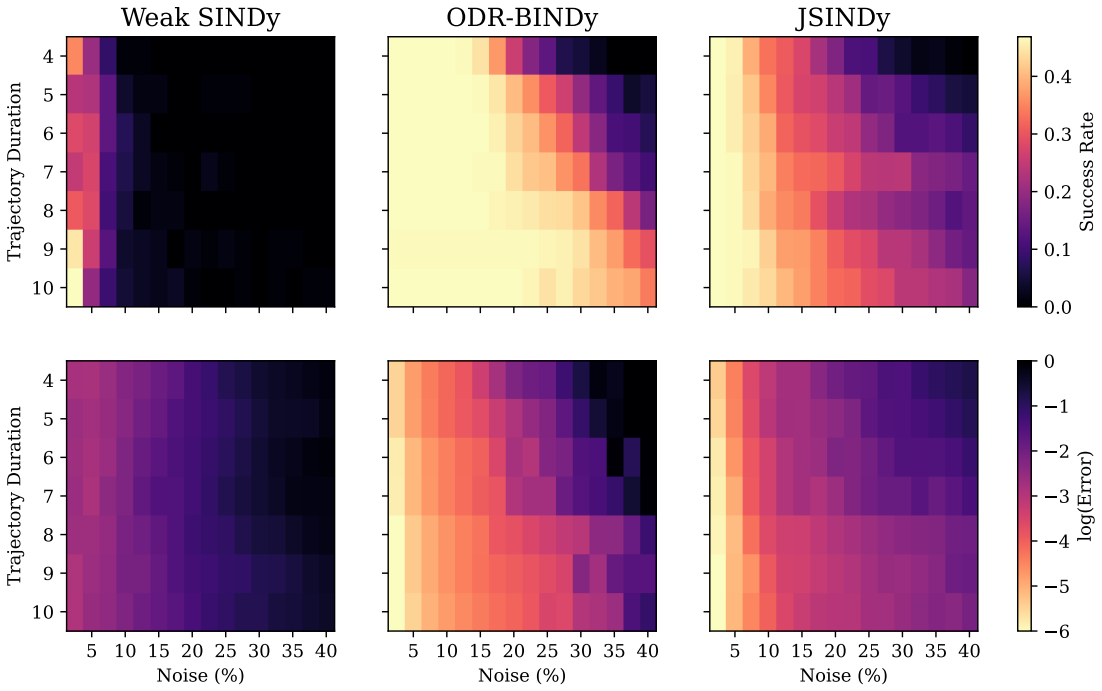

Figure 14: Experiment across different trajectory lengths and noise levels, averaged over 128 independent trials and $\Delta t = 0.01$. (3). **Top:** Proportion of trials where the exact support was exactly recovered **Bottom:** Relative coefficient error $\mathrm{RMSE}_{\mathrm{Fro}}(\widehat{\boldsymbol{\theta}}, \boldsymbol{\theta})$ of identified system.

JSINDy with data weight $\alpha = 1$, collocation weight $\beta = 10^5$, regularization $\lambda = 10^{-5}$, and used SR3 as a sparsifier with an L1 penalty with weight 5, and relaxation coefficient $\nu = 0.05$ [102]. We compare to ensemble weak SINDy and ODR-BINDy. The setup is identical to that of [27, Sec. 3.2], though we did not exclude the constant function from the library for weak SINDy. ODR-BINDy performed best in variable selection, likely owing to their approach of Bayesian evidence maximization. While also better than JSINDy in coefficient error, the difference is somewhat less pronounced. A much larger difference can be seen between JSINDy and ensemble weak SINDy, as well as the other models shown in [27, Sec. 3.2].

5 Discussion and Conclusion

A key technique used in this work is the least squares relaxation of the ODE constraint:

$$\frac{d^p}{dt^p}\boldsymbol{x} = f(D[\boldsymbol{x}](t); \boldsymbol{\theta}) \tag{59}$$

into the penalty term

$$\mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2}\int_0^T \left| \frac{d^p}{dt^p}\boldsymbol{x} - f(D[\boldsymbol{x}](t); \boldsymbol{\theta}) \right|^2 dt. \tag{60}$$

In this section, we provide some insight into the mathematical underpinnings of this approach, discuss the theoretical advantages, and highlight the role of regularization, while providing additional context and reference to trace many of these ideas in the literature.

The least squares relaxation has been applied in the PDE inverse problem setting [54], where it was found to mitigate the effects of local minima [55]. It was further analyzed in [56], which

presented a representer theorem for relaxations of parametric linear PDEs and cast the inversion problem as minimizing a special RKHS norm. The least squares approach is also common in the ML literature, most notably through physics informed neural networks [73], as well as approaches to learning differential equations and parameter estimation [21, 27, 41, 78, 88, 89]. In our case, the additional error associated to the ODE from applying a least-squares approach can be beneficial to the overall learning process, rather than a limitation. We prefer the relaxation approach to alternatives, such as reduced or shooting methods that view the ODE as a constraint and enforce feasibility from the start, which can be seen in neural ODEs, inverse problems, and optimal control [19, 33, 48, 58], or formulations that enforce the hard constraint within an all-at-once formulation such as [39], which optimizes by solving the discretized KKT conditions.

The RKHS regularized least squares approach that we take has two major advantages over reduced approaches. First, state estimates for any length of time are well defined for a much broader class of ODEs, which can be seen from the existence of bounded solutions in Theorem 2.2. Further, the state estimates are allowed to be self-correcting in that the additional data misfit terms may inform the state estimates. In contrast, in reduced approaches, certain parameter values may lead to finite time divergence of the ODE. Second, this approach improves the regularity of the objective with respect to the system parameters $\boldsymbol{\theta}$. For chaotic dynamics, we can generically expect that small perturbations to parameters of the system will lead to exponentially large changes over long periods of time relative to the Lyapunov exponent. Several works have observed that this leads to extremely ill-conditioned and non-convex objectives in reduced formulations, especially in chaotic regimes where the mismatch between simulated and observed trajectories can grow exponentially with time [15, 16, 76]. As a result, gradient-based optimization often stalls, converges to spurious local minima, or fails entirely. These issues can be mitigated significantly through multiple-shooting and multipoint-penalty methods [15, 16, 22], which break a long trajectory into shorter segments, while simulating within each segment. The full space approaches considered in this work can, in a sense, be seen as the limit of many segments in a multiple-shooting method, but we leave a more systematic comparison to future work.

The least squares relaxation avoids simulating forward trajectories altogether and instead distributes the residual over time, substantially smoothing the optimization landscape and improving robustness. Further, it makes the objective a *linear* least squares regression problem in the coefficients $\boldsymbol{\theta}$ for any fixed state estimate $\boldsymbol{x}$. This natural perspective also appears in [56] in the context of recovering a coefficient field in a PDE-constrained inverse problem. [27] points out the connection to orthogonal distance regression in all-at-once approaches, casting optimization with respect to the state variables as adjusting the inputs to correct an error-in-variables bias. Indeed, in the linear case, such an approach appears as early as 1950 in Householder's method for exponential fitting [40], where the error-in-variables issue in Prony's method is corrected by jointly fitting state estimates and discrete time dynamics.

Least squares relaxation of the ODE constraint can also be seen as minimizing a backwards error of the ODE. Introducing an auxiliary forcing function $\boldsymbol{r}(t)$, we may equivalently write the least squares penalty as

$$\mathcal{L}_C(\boldsymbol{x}, \boldsymbol{\theta}) = \begin{cases} \min\limits_{\boldsymbol{r} \in L^2([0,T], \mathbb{R}^d)} & \int_0^T |\boldsymbol{r}(t)|^2 \, dt \\ \text{subject to } \dfrac{d^p}{dt^p}\boldsymbol{x}(t) = f(D[\boldsymbol{x}](t); \boldsymbol{\theta}) + \boldsymbol{r}(t) \end{cases}, \tag{61}$$

where the ODE is understood in the weak sense.

When this formulation of $\mathcal{L}_C$ is embedded into the optimization problem eq. (5) with an appropriate lifting to optimize with respect to $\boldsymbol{r}$, this can be interpreted as finding a control or forcing term $\boldsymbol{r}(t)$ with small L^2 norm which is optimized to steer $\boldsymbol{x}$ to match the data and have small RKHS norm. This view is also well motivated when the model is misspecified, where $\boldsymbol{r}$ can act as an error term in

the dynamics.

While one can control the error in terms of the size of the residual $\boldsymbol{r}$ and a Lipschitz assumption on f via classical error analysis for ODEs using Grönwall's inequality, this will generally result in overly pessimistic error estimates which grow exponentially in time as it assumes worst case behavior of $\boldsymbol{r}$. Rather, $\boldsymbol{r}$ largely pushes the trajectory to match the data, so its effects should be small when the model is well specified, and large in misspecified cases, where it is necessary. By assimilating the observation data over a long trajectory, the error can be compared to that of Kalman filtering/smoothing (see e.g. [1]) under very low system noise and a reasonably accurate system model.

In our case, the regularity of $\boldsymbol{r}$ follows from the smoothness of $\boldsymbol{x}$ lent by the RKHS regularization and continuity of f, noting that $\boldsymbol{r}(t) = \frac{d^p}{dt^p}\boldsymbol{x}(t) - f(D[\boldsymbol{x}](t); \boldsymbol{\theta})$, the difference of two continuous functions. However, without regularization of $\boldsymbol{x}$, there is no reason to believe that $\boldsymbol{r}$ is continuous, so $\boldsymbol{x}$ will only be piecewise p-times differentiable; for first order ODEs, one should expect kinks in the state estimates without higher order regularization. The closely related approach of ODR-BINDy [27] interprets the least squares approach as a stochastic relaxation, modeling error in the ODE as independent white noise. However, it can be difficult to rigorously interpret the noise stochastically taking the function space limit of inferring $\boldsymbol{x}(\cdot)$; in the random setting and without regularization, the covariance of the residuals becomes increasingly important as the mesh is refined and $\boldsymbol{r}(t)$ is considered as a random process. In the case that the residuals are uncorrelated (e.g. $\boldsymbol{r}(t), \boldsymbol{r}(t')$ are independent for $t \neq t'$), then either the pointwise variances have to blow up, or the noise completely washes out under the integral. The probabilistic model can be seen as applying maximum a posteriori estimation to the stochastic differential equation

$$d\boldsymbol{x} = f(\boldsymbol{x})dt + \epsilon d\boldsymbol{B}_t. \tag{62}$$

In this model, $\frac{d}{dt}\boldsymbol{x}$ will not be defined pointwise, possessing only the regularity of white noise. Of course, these considerations do not manifest computationally; the error terms are end up correlated, there is no "real" Brownian motion happening, and most reasonable discretizations will implicitly regularize the trajectory estimate. However, this does raise the question of determining a computationally effective and complete stochastic model in the vein of probabilistic numerics [37], as well as the idea of penalizing the residual in a stronger norm than L^2.

Conclusion

We proposed and demonstrated the effectiveness of JSINDy in both system identification and filtering by performing both jointly. We addressed a number of challenging problems in continuous time system within a single framework based on collocation in RKHSs, including scarse sampling, far below what has been considered in previous works, partially observed states, and higher order dynamics. Building on other recent advancements in all-at-once system identification methods [21, 27, 39, 77, 88], this work provides more strong evidence for the superiority of all-at-once methods for system identification, and demonstrates significant flexibility. In short, the assumption that the data satisfy some dynamics is a very good prior to use when it is true. Overall, the RKHS formulation provides a flexible and effective tool for modeling while providing a principled approach to regularization, resulting in stable optimization, and allowing for seamlessly incorporating prior assumptions, data, and dynamics.

Currently, the primary drawback of JSINDy is that dense kernel matrices arise from the global approximation, leading to cubic complexity in the trajectory length. This can, however, be mitigated, either through a state-space representation of Matèrn Gaussian processes as in [80], which essentially amounts to a finite difference approximation to the associated Sobolev norm, and would result in banded Jacobian matrices.

Further improvements are available on the side of optimization. The objective function in eq. (8) is amenable to partial minimization [2, 34, 68] with respect to the coefficients $\boldsymbol{\theta}$ in solving the fixed-

support nonlinear least squares problems. Our approach of alternating with an algorithmic sparsifier another choice which could use more investigation. In particular, incorporating Bayesian evidence as a model selection criteria could be highly effective, while raising interesting questions for efficiently solving discrete optimization problems that arise [27, 28, 51].

Acknowledgments

AH and BH acknowledge support from the National Science Foundation under awards 2208535 (Machine Learning for Bayesian Inverse Problems) and 2337678 (CAREER: Gaussian Processes for Scientific Machine Learning: Theoretical Analysis and Computational Algorithms). AH acknowledges support from a Carl E. Pearson Fellowship. JNK was supported in part by the NSF AI Institute for Dynamical Systems (dynamicsai.org), grant 2112085. The numerical experiments involving ODRBINDy were completed on Hyak, UW's high performance computing cluster, which is funded by the UW student technology fee.

References

[1] A. Aravkin, J. V. Burke, L. Ljung, A. Lozano, and G. Pillonetto. "Generalized Kalman smoothing: Modeling and algorithms." In: *Automatica* 86 (2017), pp. 63–86.

[2] A. Y. Aravkin, R. Baraldi, and D. Orban. "A Levenberg–Marquardt method for nonsmooth regularized least squares." In: *SIAM J. Sci. Comput.* 46 (4 2024), A2557–A2581.

[3] A. Y. Aravkin, D. Drusvyatskiy, and T. van Leeuwen. "Efficient Quadratic Penalization Through the Partial Minimization Technique." In: *IEEE Transactions on Automatic Control* 63.7 (2018), pp. 2131–2138.

[4] A. Argyriou and F. Dinuzzo. "A unifying view of representer theorems." In: *ICML* 32 (2 2014). Ed. by E. P. Xing and T. Jebara, pp. 748–756.

[5] K. J. Åström and P. Eykhoff. "System identification—a survey." In: *Automatica* 7.2 (1971), pp. 123–162.

[6] J. Bakarji, K. Champion, J. Nathan Kutz, and S. L. Brunton. "Discovering governing equations from partial measurements with deep delay autoencoders." In: *Proc. Math. Phys. Eng. Sci.* 479 (2276 2023).

[7] D. Bertsimas and W. Gurnee. "Learning sparse nonlinear dynamics via mixed-integer optimization." In: *Nonlinear Dynamics* 111.7 (2023), pp. 6585–6604.

[8] J. Bongard and H. Lipson. "Automated reverse engineering of nonlinear dynamical systems." In: *Proceedings of the National Academy of Sciences of the United States of America* 104 (2007), pp. 9943–8.

[9] J. Bradbury, R. Frostig, P. Hawkins, et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.6.0. 2018.

[10] D. S. Broomhead and R. Jones. "Time-series analysis." In: *Proc. R. Soc. Lond.* 423 (1864 1989), pp. 103–121.

[11] S. L. Brunton, J. L. Proctor, and J. Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems." In: *Proceedings of the National Academy of Sciences* 113.15 (2016).

[12] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz. "Chaos as an intermittently forced linear system." In: *Nature Communications* 8.1 (2017). Publisher: Nature Publishing Group, p. 19.

[13] E. J. Candès, J. Romberg, and T. Tao. "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information." In: *IEEE Transactions on Information Theory* 52.2 (2006), pp. 489–509.

[14] E. J. Candès and M. B. Wakin. "An introduction to compressive sampling." In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 21–30.

[15] F. Carbonell, Y. Iturria-Medina, and J. C. Jimenez. "Multiple shooting-Local Linearization method for the identification of dynamical systems." In: *arXiv preprint arXiv:1507.00975* (2015).

[16] D. Chakraborty, S. W. Chung, T. Arcomano, and R. Maulik. "Divide And Conquer: Learning Chaotic Dynamical Systems With Multistep Penalty Neural Ordinary Differential Equations." In: *arXiv preprint arXiv:2407.00568* (2024).

[17] K. Champion, P. Zheng, A. Y. Aravkin, S. L. Brunton, and J. N. Kutz. "A unified sparse optimization framework to learn parsimonious physics-informed models from data." In: *IEEE Access* 8 (2020), pp. 169259–169271.

[18] R. Chartrand. "Numerical differentiation of noisy, nonsmooth data." In: *ISRN Appl. Math.* 2011 (1 2011), pp. 1–11.

[19] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. "Neural Ordinary Differential Equations." In: *Advances in Neural Information Processing Systems (NeurIPS 31)*. 2018, pp. 6572–6583.

[20] Y. Chen, B. Hosseini, H. Owhadi, and A. M. Stuart. "Solving and learning nonlinear PDEs with Gaussian processes." In: *Journal of Computational Physics* 447 (2021), p. 110668.

[21] Z. Chen, Y. Liu, and H. Sun. "Physics-informed learning of governing equations from scarce data." In: *Nature Communications* 12 (2021).

[22] S. W. Chung and J. B. Freund. "An optimization method for chaotic turbulent flow." In: *J. Comput. Phys.* 457 (111077 2022), p. 111077.

[23] D. L. Donoho. "Compressed sensing." In: *IEEE Transactions on Information Theory* 52.4 (2006), pp. 1289–1306.

[24] L. C. Evans. *Partial Differential Equations*. 2nd. Vol. 19. Graduate Studies in Mathematics. Providence, RI: American Mathematical Society, 2010.

[25] U. Fasel, J. N. Kutz, B. W. Brunton, and S. L. Brunton. "Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 478.2260 (2022). Publisher: Royal Society, p. 20210904.

[26] L. Fung. *ODR-BINDy*. Version 1.0.0. 2025.

[27] L. Fung. "Overcoming error-in-variable problem in data-driven model discovery by orthogonal distance regression." In: *arXiv [stat.ME]* (2025).

[28] L. Fung, U. Fasel, and M. Juniper. "Rapid Bayesian identification of sparse nonlinear dynamics from scarce and noisy data." In: *Proc. Math. Phys. Eng. Sci.* 481 (2307 2025).

[29] L. M. Gao, U. Fasel, S. L. Brunton, and J. N. Kutz. *Convergence of uncertainty estimates in Ensemble and Bayesian sparse model discovery*. 2023. arXiv: `2301.12649 [cs.LG]`.

[30] L. M. Gao and J. N. Kutz. *Bayesian autoencoders for data-driven discovery of coordinates, governing equations and fundamental constants*. 2022.

[31] E. George, C. E. Chan, G. Dimand, R. M. Chakmak, C. Falcon, D. Eckhardt, and R. S. Martin. "Decomposing Signals from Dynamical Systems Using Shadow Manifold Interpolation." In: *SIAM Journal on Applied Dynamical Systems* 20.4 (2021), pp. 2236–2260.

[32] M. GEVERS. "A personal view of the development of system identification: A 30-year journey through an exciting field." In: *IEEE Control Systems Magazine* 26.6 (2006), pp. 93–105.

[33] D. Givoli. "A tutorial on the adjoint method for inverse problems." In: *Comput. Methods Appl. Mech. Eng.* 380 (113810 2021), p. 113810.

[34] G. H. Golub and V. Pereyra. "The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate." In: *SIAM J. Numer. Anal.* 10 (2 1973), pp. 413–432.

[35] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II Stiff and Differential-Algebraic Problems*. Second Revised Edition. Berlin: Springer, 2002.

[36] B. Hamzi, H. Owhadi, and Y. Kevrekidis. "Learning dynamical systems from data: A simple cross-validation perspective, part IV: Case with partial observations." In: *Physica D: Nonlinear Phenomena* 454 (2023), p. 133853.

[37] P. Hennig, M. A. Osborne, and H. P. Kersting. *Probabilistic numerics: Computation as machine learning*. Cambridge, England: Cambridge University Press, 2022. 410 pp.

[38] S. M. Hirsh, D. A. Barajas-Solano, and J. N. Kutz. "Sparsifying priors for Bayesian uncertainty quantification in model discovery." In: *R. Soc. Open Sci.* 9 (2 2022), p. 211823.

[39] J. M. Hokanson, G. Iaccarino, and A. Doostan. "Simultaneous identification and denoising of dynamical systems." In: *SIAM J. Sci. Comput.* 45 (4 2023), A1413–A1437.

[40] A. S. Householder. *On Prony's method of fitting exponential decay curves and multiple-hit survival curves*. Tech. rep. 1950.

[41] Y. Jalalian, J. F. O. Ramirez, A. Hsu, B. Hosseini, and H. Owhadi. *Data-Efficient Kernel Methods for Learning Differential Equations and Their Solution Operators: Algorithms and Error Analysis*. 2025. arXiv: `2503.01036 [stat.ML]`.

[42] A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. New York: Academic Press, 1970.

[43] *JSINDy*. `https://github.com/AHsu98/jsindy`. 2025.

[44] K. Kaheman, S. L. Brunton, and J. Nathan Kutz. "Automatic differentiation to simultaneously identify nonlinear dynamics and extract noise probability distributions from data." In: *Machine Learning: Science and Technology* 3.1 (2022), p. 015031.

[45] M. Kanagawa, P. Hennig, D. Sejdinovic, and B. K. Sriperumbudur. *Gaussian Processes and Kernel Methods: A Review on Connections and Equivalences*. 2018. arXiv: `1807.02582 [stat.ML]`.

[46] A. A. Kaptanoglu, B. M. de Silva, U. Fasel, et al. "PySINDy: A comprehensive Python package for robust sparse system identification." In: *Journal of Open Source Software* 7.69 (2022), p. 3994.

[47] K. J. Keesman. *System identification: an introduction*. Springer Science & Business Media, 2011.

[48] P. Kidger. "On Neural Differential Equations." PhD thesis. University of Oxford, 2021.

[49] P. Kidger. *sympy2jax: Turn SymPy expressions into trainable JAX expressions*. Version 0.0.7. Apache-2.0 license. 2025.

[50] P. Kidger and C. Garcia. "Equinox: neural networks in JAX via callable PyTrees and filtered transformations." In: *Differentiable Programming workshop at Neural Information Processing Systems 2021* (2021).

[51] A. A. Klishin, J. Bakarji, J. N. Kutz, and K. Manohar. "Statistical mechanics of dynamical system identification." In: *arXiv [cond-mat.stat-mech]* (2025).

[52] K. Lahouel. "Learning nonparametric ordinary differential equations from noisy data." In: *Journal of Computational Physics* 479 (2024), p. 111019.

[53] S. Lee, F. Dietrich, G. E. Karniadakis, and I. G. Kevrekidis. "Linking Gaussian process regression with data-driven manifold embeddings for nonlinear data fusion." In: *Interface Focus* 9 (3 2019), p. 20180083.

[54] T. van Leeuwen and F. J. Herrmann. "A penalty method for PDE-constrained optimization in inverse problems." In: *arXiv [math.OC]* (2015).

[55] T. van Leeuwen and F. J. Herrmann. "Mitigating Local Minima in Full-Waveform Inversion by Expanding the Search Space." In: *Geophysical Journal International* 195.1 (2013), pp. 661–667.

[56] T. van Leeuwen and Y. Yang. "An analysis of constraint-relaxation in PDE-based inverse problems." In: *Inverse Problems* 41 (2 2025), p. 025009.

[57] K. Levenberg. "A method for the solution of certain non-linear problems in least squares." In: *Quarterly of applied mathematics* 2.2 (1944), pp. 164–168.

[58] J.-L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations*. Springer-Verlag, 1971.

[59] L. Ljung. *System identification (2nd ed.): theory for the user*. USA: Prentice Hall PTR, 1999.

[60] L. Ljung. "Perspectives on system identification." In: *Annual Reviews in Control* 34.1 (2010), pp. 1–12.

[61] D. Long, N. Mrvaljević, S. Zhe, and B. Hosseini. "A kernel framework for learning differential equations and their solution operators." In: *Physica D: Nonlinear Phenomena* 460 (2024), p. 134095.

[62] D. W. Marquardt. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters." In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441.

[63] D. A. Messenger and D. M. Bortz. "Weak SINDy for partial differential equations." In: *Journal of Computational Physics* 443 (2021).

[64] A. Meurer, C. P. Smith, M. Paprocki, et al. "SymPy: symbolic computing in Python." In: *PeerJ Computer Science* 3 (2017), e103.

[65] R. K. Niven, A. Mohammad-Djafari, L. Cordier, M. W. Abel, and M. Quade. "Bayesian identification of dynamical systems." In: *39th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MaxEnt 2019)*. Vol. 33. 2019.

[66] J. S. North, C. K. Wikle, and E. M. Schliep. "A Bayesian approach for data-driven dynamic equation discovery." In: *Journal of Agricultural, Biological and Environmental Statistics* 27.4 (2022), pp. 728–747.

[67] J. S. North, C. K. Wikle, and E. M. Schliep. "A Bayesian Approach for Spatio-Temporal Data-Driven Dynamic Equation Discovery." In: *Bayesian Analysis* 1.1 (2023), pp. 1–30.

[68] D. P. O'Leary and B. W. Rust. "Variable projection for nonlinear least squares problems." In: *Comput. Optim. Appl.* 54 (3 2013), pp. 579–593.

[69] H. Owhadi. "Computational graph completion." In: *Res. Math. Sci.* 9 (2 2022), pp. 1–33.

[70] H. Owhadi and C. Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization*. 1st ed. Cambridge University Press, 2019.

[71] S. Pan and K. Duraisamy. "On the structure of time-delay embedding in linear models of non-linear dynamical systems." In: *Chaos* 30 (7 2020), p. 073135.

[72] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman. *Universal differential equations for scientific machine learning*. 2020.

[73] M. Raissi, P. Perdikaris, and G. E. Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations." In: *J. Comput. Phys.* 378 (2019), pp. 686–707.

[74] M. Raissi. "Deep hidden physics models: Deep learning of nonlinear partial differential equations." In: *Journal of Computational Physics* 357 (2018), pp. 125–141.

[75] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[76] A. H. Ribeiro, K. Tiels, J. Umenberger, T. B. Schön, and L. A. Aguirre. "On the smoothness of nonlinear system identification." In: *Automatica* 121 (2020), p. 109158.

[77] H. Ribera, S. Shirman, A. V. Nguyen, and N. M. Mangan. "Model selection of chaotic systems from data with hidden variables using sparse data assimilation." In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 32.6 (2022), p. 063101.

[78] S. H. Rudy, S. L. Brunton, and J. N. Kutz. "Smoothing and parameter estimation by soft-adherence to governing equations." In: *Journal of Computational Physics* 398 (2019), p. 108860.

[79] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz. *Data-driven discovery of partial differential equations*. 2016. arXiv: `1609.06401 [nlin.PS]`.

[80] Y. Saatçi. "Scalable inference for structured Gaussian process models." PhD thesis. University of Cambridge, 2012.

[81] H. Schaeffer. "Learning partial differential equations via data discovery and sparse optimization." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2197 (2017), p. 20160446.

[82] M. Schmidt and H. Lipson. "Distilling Free-Form Natural Laws from Experimental Data." In: *Science (New York, N.Y.)* 324 (2009), pp. 81–5.

[83] B. Schölkopf, R. Herbrich, and A. J. Smola. "A Generalized Representer Theorem." In: *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 416–426.

[84] G. L. Smith, S. F. Schmidt, and L. A. McGee. *Application of Statistical Filter Theory to the Optimal Estimation of Position and Velocity on Board a Circumlunar Vehicle*. NASA-TR-R-135. NTRS Author Affiliations: NASA Ames Research Center NTRS Document ID: 20190002215 NTRS Research Center: Ames Research Center (ARC). 1962.

[85] G. Söderlind. "Automatic control and adaptive time-stepping." In: *Numerical Algorithms* 31 (2002), pp. 281–310.

[86] J. Stevens-Haas, S. E. Webster, and A. Aravkin. "Theoretical advances in current estimation and navigation from a glider-based acoustic Doppler current profiler (ADCP)." In: *arXiv [math.OC]* (2021).

[87] J. M. Stevens-Haas, Y. Bhangale, J. Nathan Kutz, and A. Aravkin. "Learning Nonlinear Dynamics Using Kalman Smoothing." In: *IEEE Access* 12 (2024), pp. 138564–138574.

[88] F. Sun, Y. Liu, and H. Sun. "Physics-informed Spline Learning for Nonlinear Dynamics Discovery." In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. by Z.-H. Zhou. Main Track. International Joint Conferences on Artificial Intelligence Organization, 2021, pp. 2054–2061.

[89] L. Sun, D. Z. Huang, H. Sun, and J.-X. Wang. *Bayesian Spline Learning for Equation Discovery of Nonlinear Dynamics with Quantified Uncertainty*. 2022. arXiv: `2210.08095 [cs.LG]`.

[90] F. Takens. "Detecting strange attractors in turbulence." In: *Dynamical Systems and Turbulence, Warwick 1980*. Ed. by D. Rand and L.-S. Young. Vol. 898. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, pp. 366–381.

[91] G. Teschl. *Ordinary differential equations and dynamical systems*. Vol. 140. American Mathematical Soc., 2012.

[92] R. Tibshirani. "Regression Shrinkage and Selection via the Lasso." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288.

[93] C. Tsitouras. "Runge–Kutta pairs of order 5 (4) satisfying only the first column simplifying assumption." In: *Computers & Mathematics with Applications* 62.2 (2011), pp. 770–775.

[94] F. van Breugel, J. Nathan Kutz, and B. W. Brunton. "Numerical differentiation of noisy data: A unifying multi-objective optimization framework." In: *IEEE Access* (2020).

[95] J. M. Varah. "A spline least squares method for numerical parameter estimation in differential equations." In: *SIAM J. Sci. Stat. Comput.* 3 (1 1982), pp. 28–46.

[96] W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, and C. Grebogi. "Predicting Catastrophes in Nonlinear Dynamical Systems by Compressive Sensing." In: *Phys. Rev. Lett.* 106 (15 2011), p. 154101.

[97] H. Wendland. *Scattered data approximation*. Cambridge monographs on applied and computational mathematics. Cambridge, England: Cambridge University Press, 2010. 348 pp.

[98] S. Wright, J. Nocedal, et al. "Numerical optimization." In: *Springer Science* 35.67-68 (1999), p. 7.

[99] Z. Xu, D. Long, Y. Xu, G. Yang, S. Zhe, and H. Owhadi. "Toward Efficient Kernel-Based Solvers for Nonlinear PDEs." In: *arXiv [cs.LG]* (2025).

[100] Y. Yang, M. Aziz Bhouri, and P. Perdikaris. "Bayesian differential programming for robust systems identification under uncertainty." In: *Proceedings of the Royal Society A* 476.2243 (2020), p. 20200290.

[101] S. Zhang and G. Lin. "Robust data-driven discovery of governing physical laws with error bars." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 474.2217 (2018), p. 20180305.

[102] P. Zheng, T. Askham, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin. "A Unified Framework for Sparse Relaxed Regularized Regression: SR3." In: *IEEE Access* 7 (2019), pp. 1404–1423.

[103] D.-X. Zhou. "Derivative reproducing properties for kernel methods in learning theory." In: *J. Comput. Appl. Math.* 220 (1-2 2008), pp. 456–463.

A Theory

We provide some more details of the RKHS theory that underlies our approach and the least squares relaxation approach we take for the ODE.

A.1 RKHS and representer theorems

We first give a version of the representer theorem in the Hilbert space setting which is appropriate for Theorem 2.1. These theorems are well known and appear in various forms throughout the literature;we state it for completeness. The variant that we present can be found in [4, 70, 83], and similar results appear in [20, 45, 103].

Theorem A.1. *Let $\mathcal{H}$ be a separable Hilbert space. Let $F : \mathcal{H} \mapsto \mathbb{R}$ be given by*

$$F(u) = f(\langle u, w_1\rangle, ..., \langle u, w_m\rangle) + J(\|u\|) \tag{63}$$

where $J : \mathbb{R}^+ \mapsto \mathbb{R} \cup \{\infty\}$ is strictly increasing, w_i are bounded linear functionals, and $f : \mathbb{R}^m \mapsto \mathbb{R} \cup \{\infty\}$. Then if F admits a minimizer u^* with $F(u^*) < \infty$, then

$$u^* = \sum_{i=1}^{m} c_i w_i. \tag{64}$$

A.2 Matérn kernels and Sobolev spaces

Here, we review some prerequisite RKHS and Sobolev space theory, giving more details on the kernel functions we use, and discussing the smoothness of functions in RKHS $\mathcal{X}$ associated to the kernels we define in eq. (9). Much of this will pull directly from [45, 97] where we direct the interested reader for additional details. First, recall from eq. (9) that we take the sum of a constant kernel with a Matérn kernel

$$k(t, t') = c_0 + c_1 h_\nu\left(\frac{|t - t'|}{\ell}\right), \quad h_\nu(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu} r\right)^\nu K_\nu\left(\sqrt{2\nu}\, r\right). \tag{65}$$

From [45, Ex. 2.6], the RKHS $\mathcal{H}_k$ associated to this kernel is norm equivalent to the Sobolev space $H^{\nu+\frac{1}{2}}([0, T])$, functions which are $\nu + \frac{1}{2}$ times differentiable in the mean-square sense. Compactness of the domain means that adding a constant to the kernel function does not affect the equivalence. By the Sobolev embedding [24, Thm. 5.6.6, p. 270], this implies that $\mathcal{H}_k$ is also compactly embedded in the Hölder space $C^p([0, T])$ for $p < \nu$.

When ν is a half integer (i.e. $\nu = n + \frac{1}{2}$), h_ν reduces to the product of a polynomial and exponential in r.

$$h_\nu(r) = \exp\left(-\sqrt{2\nu} r\right) \frac{p!}{(2p)!} \sum_{i=0}^{p} \frac{(\nu - \frac{1}{2} + i)!}{i!(\nu - \frac{1}{2} - i)!}\left(2\sqrt{2\nu} r\right)^{\nu - \frac{1}{2} - i}. \tag{66}$$

Stably computing high order derivatives of this h_ν can still be challenging due to the absolute value inside of h_ν, and directly applying autodiff to this representation can cause catastrophic cancellation. However, for $j < 2\nu$, $\frac{d^j}{dt^j} h_\nu(\frac{|t-t'|}{\ell})$, can also be written as the product of a polynomial and exponential function; we compute this representation of the derivatives in SymPy[64], and then convert it Jax using Sympy2Jax [49].

A.3 Cholesky factorization and choice of basis

We perform a linear change of variables in order to simplify bookkeeping and mitigate some of the effects of ill-conditioning and numerical roundoff on convergence by reparametrizing with the Cholesky factors of $\boldsymbol{K}$. That is, write

$$\boldsymbol{K} = \boldsymbol{C}\boldsymbol{C}^\top \tag{67}$$

for the Cholesky factorization of $\boldsymbol{K}$, and set

$$\tilde{\boldsymbol{z}} = (\tilde{\boldsymbol{z}_1}, ..., \tilde{\boldsymbol{z}_d}), \quad \tilde{\boldsymbol{z}}_m = \boldsymbol{C}^\top \boldsymbol{z}_m. \tag{68}$$

Under this parametrization, we have that

$$\boldsymbol{x}(\cdot; (\boldsymbol{C}^{-T}\tilde{\boldsymbol{z}}_1, ..., \boldsymbol{C}^{-T}\tilde{\boldsymbol{z}}_d)) = \boldsymbol{x}(\cdot; \boldsymbol{z}). \tag{69}$$

It is clear that these are interchangable *mathematically*. Computationally, they relegate some of the bookkeeping with tracking the kernel matrix to an initial Cholesky factorization, which, in our implementation, is absorbed into the parametrization, though for simplicity here, we continue the exposition using the original variables. The fact that this mitigates some of the effects of ill-conditioning was somewhat mysterious to us, as it still entails computing Cholesky factorizations of ill-conditioned matrices.

B Computational tools

We perform all of our computations using Jax [9], and solve ODEs using the Tsit5 [93] solver with a PID controller for adaptive timestepping [35, 85] as recommended by, and implemented in Diffrax [48, 50]. Comparisons with SINDy use the PySINDy package [46]. Our experiments and current implementation are available in the GitHub repository [43]; an optimized implementation of JSINDy will soon be added to PySINDy. Our computations with JAX used an RTX-5000 Ada Generation GPU in a Lambda machine.