Data driven discovery of systems of ordinary differential equations using nonconvex multitask learning

Clément Lejeune, Josiane Mothe, Adil Soubki, Olivier Teste

Data Driven Discovery of Systems of Ordinary Differential Equations Using Nonconvex Multitask Learning

Clément Lejeune[1*†], Josiane Mothe[1], Adil Soubki[2] and Olivier Teste[1]

[1] Institut de Recherche en Informatique de Toulouse, Université de Toulouse France .
[2] Airbus Operations, Toulouse, France .

*Corresponding author(s). E-mail(s): clement.lejeune@irit.fr;
Contributing authors: josiane.mothe@irit.fr;
adil.soubki@airbus.com; olivier.teste@irit.fr;
†now: clement-c.lejeune@thalesgroup.com

Abstract

In physical sciences, dynamic systems are modeled using their parameters within governing equations that often form a system of ordinary differential equations (SODE). This system consists of multiple equations, each of which relates the time derivative of a single parameter to several parameters. A parameter can appear in multiple equations, and this parameter potentially links the equations to each other. Although in certain cases the SODE can be written by domain experts, it is often unknown. With advances in sensor technology, large quantities of data can be sampled from dynamic systems, thus enabling the data-driven discovery of closed-form SODEs. State-of-the-art approaches are based on sparse single-task learning, which means that each equation from the SODE is learned independently. Omitting the coupling features of equations leads to SODEs that weakly identify the dynamic system. Furthermore, the convexity of the sparse penalty included in the learning criterion gives an SODE that is biased with respect to the true SODE. To reduce such a bias, we propose a multitask learning (MTL) based penalty which can learn the closed-form SODE with unbiasedness. The purpose of each task is to discover a single equation. But discovering an SODE is nontrivial, as dynamic systems are often nonlinear

and the available data are noisy. Our proposal improves SODE identification by harnessing a nonconvex sparse matrix-structured penalty which takes into account the coupling feature as well as addresses the bias issue. Experimental results, based on noisy data simulated from known SODEs, confirm that, compared to single-task learning, MTL is more effective for recovering the closed-form SODE, and the proposed nonconvexity ensures that it can be estimated with unbiasedness. We also show the benefits of our approach on a real-world public dataset sampled from a laboratory-based ecological experiment.

Keywords: sparse estimator, unbiased sparsity, multitask learning, data-driven identification of dynamics, multivariate time series

1 Introduction

Governing equations are mathematical models, such as partial differential equations (e.g. Navier-Stokes equation for fluid dynamic modeling) or systems of ordinary differential equations (SODEs), which are widely used in science and engineering to model dynamic systems [1, 2]. A governing equation models the dependency relationships between several parameters like velocity or chemical concentration, known as state variables, in a dynamic system. The solution to a governing equation is a function depicting the temporal and/or spatial evolution of the state variables that correspond to physical quantities (e.g. current-voltage, motion). Traditionally, governing equations are derived from principles that have been formalized from general empirical observations consistent with certain hypotheses, for instance Newton's laws are based on the constant-mass hypothesis [3]. However, in practice it is hard to derive governing equations from the existing rules, because either the practitioners do not have sufficient knowledge or they do not have sufficient time.

In line with the development of sensor technology, data can be sampled from dynamic systems. This provides new opportunities to extract knowledge about the physical behavior underlying a dynamic system. Consequently, there has been a growing interest over recent years in developing data-driven methods for the discovery of governing equations [4–8].

Indeed, gaining access to the model that governs an unknown dynamic from data samples can improve our understanding of a physical system and is a challenging task of scientific interest. There is also the practical challenge of obtaining a surrogate model to simulate the system of interest in various conditions e.g. for prototype design in engineering [2].

In this paper, we focus on the case where state variables are sampled from a dynamic system throughout a scalar variable $t \in \mathbb{R}$, that without loss of generality refers to time. Thus, the sampled state variable forms a multivariate time series. If we are to discover the dynamic relationship between the state variables, we have to assume that the resulting data represent the solution of an unknown SODE as proposed by [4]. In an SODE, each equation models the

dependency relationship between several scalar-dependent state variables and their first-order time derivatives. Famous SODEs include the Lorenz attractor in turbulence atmospheric modeling and the Lotka-Voltera in population dynamics, see [9] for other examples.

Let us recall how an SODE is formalized. Let $f = \big[f_1, \ldots, f_p\big]^T : \mathbb{R}^p \to \mathbb{R}^p$ be a continuous map which defines the evolution of a state variable $\mathbf{x}(t) \in \mathbb{R}^p$ assumed first-order differentiable, then the SODE is expressed as $\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} := \dot{\mathbf{x}}(t) = f\big(\mathbf{x}(t)\big)$, or equivalently:

$$\begin{cases} \dot{x}_1(t) = f_1\big(x_1(t), \ldots, x_p(t)\big) \\ \ldots \\ \dot{x}_p(t) = f_p\big(x_1(t), \ldots, x_p(t)\big) \end{cases} \tag{1}$$

Learning an SODE from data samples can be seen as predicting the time derivative of the state variables from a combination (often nonlinear) of the state variables. Hence, discovering an SODE signifies learning f in closed form from samples of $\dot{\mathbf{x}}$ and $\mathbf{x}$. An SODE comprises multiple equations (as many as state variables) that relate the variables equations to others. In other words, the equations are coupled (See Fig 1, where the occurrence of x_1^3 and x_2^3 within both equations makes the SODE coupled). Our idea is that using multitask learning (MTL) to harness this coupling can improve the data-driven discovery of an SODE. That is why we hypothesize that in this context, MTL can be better than single-task learning.

In Fig 1 we illustrate the core of the discovery of nonlinear dynamics framework with $\mathbf{x}(t) \in \mathbb{R}^2$ and f polynomial in $\mathbf{x}(t)$: (top) based on data ($\dot{X}_n$ and $\boldsymbol{\Theta}_{X_n}$) sampled from a dynamic system. We solve the minimization problem involving a data fidelity term $\ell(\dot{X}_n, \boldsymbol{\Theta}_{X_n}\boldsymbol{\beta})$ and a sparse penalty term $R(\boldsymbol{\beta})$ (bottom center) leading to the identification of an SODE (bottom left). This optimization instantiates the problem with $R := \|\cdot\|_{1,1}$, which is convex. Then, the learned SODE is identified through the minimizer $\hat{\boldsymbol{\beta}}$ whose entries and sparsity heavily rely on the chosen penalty. Thus, when the penalty is convex (e.g. LASSO) the nonzero values are statistically biased w.r.t the true unknown values. This means that the SODE is not estimated accurately. There exist nonconvex sparse penalties (e.g. smoothly clipped sbsolute deviation [10]) to remedy the bias issue but they cannot consider the MTL feature. Existing MTL penalties (group-LASSO [11], sparse-group-lasso [12]) are convex and thus result in a biased SODE. We propose a specific nonconvex sparse penalty as a mean for learning the matrix coefficient that both reduces its bias and takes into account the multitask feature of the SODE. As a result, when there is a coupling within the true SODE, its closed-form SODE can be better recovered using our penalty.

Our contributions in this paper are as follows:

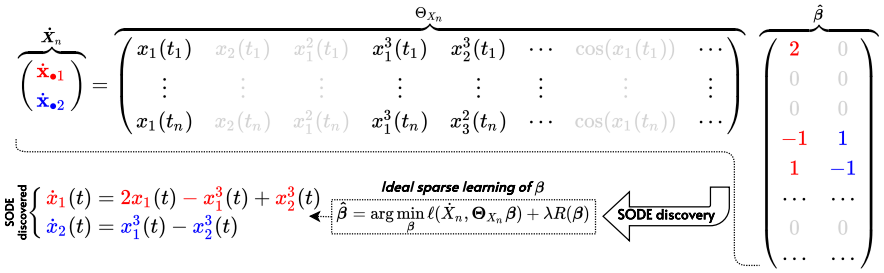


Fig. 1: (From top to center to bottom left) Generic workflow of the data-driven discovery of an SODE (here two-dimensional) with linear regression. Top: based on samples of estimated state variable time derivatives ($\dot{X}_n$'s columns), and resulting from nonlinear transformations of the state variable samples (columns of the dictionary $\boldsymbol{\Theta}_{X_n}$), a linear model is assumed for the two sets of samples. Center: the SODE is discovered by minimizing a learning criterion inducing sparsity in the minimizer. Bottom left: the SODE is identified using the learned coefficient matrix $\hat{\boldsymbol{\beta}}$.

- We recast the discovery of a closed-form SODE as a multitask problem. We also formalize the learning as an optimization problem involving a matrix-structured, sparse and nonconvex regularizer to account for task relatedness, sparsity and unbiasedness.
- We highlight the bias of the learned coefficients, induced by the convexity of state-of-the-art regularizers, on the resulting SODE.
- Using experiments on reference SODEs, we show that learning via our multitask penalty rather than a convex single-task penalty leads to a better recovery of the equations. We also demonstrate the benefit of our recasting on a real dataset involving two adversarial biological quantities, and provide an interpretation of the SODE that was discovered.

The paper is organized as follows: We start by defining our notation in Table 1. in Section 2 we introduce the data-driven discovery of an SODE as a sparse regression problem (single-task and multitask) and then highlight the drawbacks of the state-of-the-art regularizers. In Section 3, we introduce our contribution, a regularizer that is multitask-based and involves unbiasedness. Then, we present a generic algorithm from the literature which solves the regression problem and can be used for all the regularizers introduced in the paper. Through numerical experiments on synthetic and real datasets, in Section 4 we show the benefit of learning an SODE with such a regularizer. We conclude and propose some perspectives in Section 5.

2 Related work

Extracting information from noisy data resulting from physical experiments with standard machine learning algorithms can lead to irrelevant, or at worst,

Notation	Description
$\mathbb{R}^n$	Set of n-vectors with real number elements
$[a;\ b]$	Closed subset of $\mathbb{R}$
$\mathbb{R}^{n\times p}$	Set of $n \times p$ matrices with real number elements
$\mathbf{w} := [w_1, \ldots, w_n]^T$	An element of $\mathbb{R}^n$ (lowercase bold letter)
$\boldsymbol{W} := [\mathbf{w}_{\bullet 1}, \ldots, \mathbf{w}_{\bullet p}]$	An element of $\mathbb{R}^{n\times p}$ (uppercase letter)
$\mathbf{w}_{\bullet j}$	j-th column of matrix $\boldsymbol{W}$
$\mathbf{w}_{i\bullet}$	i-th row of matrix $\boldsymbol{W}$
$\mathbf{x}(t) := [x_1(t), \ldots, x_p(t)]^T$	p dimensional state variable $\mathbf{x}$ at time t
$\mathbf{x}_{\bullet k} := [x_k(t_1), \ldots, x_k(t_n)]^T$	k-th dimension of $\mathbf{x}$ sampled at $t_1 < \ldots < t_n$
$X_n := [\mathbf{x}_{\bullet 1}, \ldots, \mathbf{x}_{\bullet p}]$	Input data matrix of $(n \times p)$: n samples (row concatenation) of the sampled state variable $\mathbf{x}_{\bullet k}$
$\dot{\mathbf{x}}(t) := [\frac{\mathrm{d}x_1(t)}{\mathrm{d}t}, \ldots, \frac{\mathrm{d}x_p(t)}{\mathrm{d}t}]^T$	First-order time derivative of $\mathbf{x}$
$\dot{X}_n$	n samples (row concatenation) of $\dot{\mathbf{x}}$ at $t_1, \ldots, t_n$
$\Vert\mathbf{w}\Vert_q := (\sum_i \vert w_i\vert^q)^{1/q}$	ℓ_q norm $(q \geq 1)$ of $\mathbf{w}$
$\Vert\boldsymbol{W}\Vert_{q,r} := \left(\sum_{i=1}^n \Vert\mathbf{w}_{i\bullet}\Vert_q\right)^{1/r}$	$\ell_{q,r}$ matrix norm $(q, r \geq 1)$ of $\boldsymbol{W}$

Table 1: Notations.

incorrect conclusions. This is because the scientist does not consider the physical feature of the solution in the learning objective. Recently thare has been increasing research on physics-informed methods to incorporate a physical prior into the learning process [4, 13–16]. Generally, these methods formulate a learning objective by leveraging a model from the physical sciences. In [13] the authors proposed a method to predict a state variable with Gaussian processes in which the covariance function derives from a known partial differential equation (PDE). Another line of research is about dynamic mode decomposition in which the goal is to learn both a linear operator and the associated eigenspace from time series and/or spatial data [17]. Several dynamic mode decomposition variants have been proposed, but the most popular and recent variants rely on the Koopman operator, which provides information on the growth rates and the frequencies of the long-term dynamics [18–20]. In [21], the authors assumed that a closed-form PDE can be recovered as a sparse plus low-rank combination of nonlinear terms of precomputed dictionary. Their approach relies on the robust principal component analysis of Candès *et al.* [22]. Our work is similar to the one of [21] in the sense that we attempt to discover a closed-form model, but here an SODE is based on a dictionary computed from noisy data.

Learning an SODE from data samples can be traced back to the seminal work of [23]. Schmidt and Lipson proposed a combinatorial approach based on genetic programming to select the parsimonious model that best recovers the data from among a large set of candidate models. As mentioned in [4], genetic programming methods do not scale to large datasets and tend to overfit. To remedy this, in [4] Brunton *et al.* recast learning an SODE as a sparse regression problem, referred to as the sparse identification of nonlinear dynamics in the literature. Our proposal is formalized according to this framework, which we introduce in the next section.

2.1 Sparse single-task learning of an SODE

State-of-the-art methods for discovering an SODE, [4, 24], implicitly assume that each of $\dot{x}_1, \ldots, \dot{x}_p$ in Equation (1) are uncorrelated targets which can be predicted by a sparse combination of elements included in a given dictionary of candidate functions. An example of a dictionary $\boldsymbol{\Theta}_{X_n}$ and a two-dimensional SODE is given in Fig 1. The dictionary is built by the user with linear and nonlinear candidate functions from the samples X_n of $\mathbf{x}$. This dictionary reflects the prior knowledge of the observed phenomenon and is possibly overcomplete. In [4, 7, 24], f is assumed to be linear with respect to the dictionary elements. The linear assumption on f with respect to the dictionary elements makes it easy to learn and interpret. Next $f_1, \ldots, f_p$ are learned separately with a sparsity-promoting algorithm e.g. LASSO [25] or elastic-net [26].

2.1.1 Discovering an SODE using sparse linear regression

Starting from n noisy samples of a p-dimensional state variable stored as X_n and the associated time derivative samples $\dot{X}_n$ (often estimated from X_n, e.g. with finite-difference, spline-based methods), we first build a dictionary of m arbitrary candidate functions, for instance one chooses to include linear and quadratic monomials, and one cosine term, then, $\boldsymbol{\Theta}_{X_n} \overset{e.g.}{=} \left[\mathbf{x}_{\bullet 1}, \mathbf{x}_{\bullet 2}, \ldots, \mathbf{x}^2_{\bullet 1}, \mathbf{x}^2_{\bullet 2}, \ldots, \cos \mathbf{x}_{\bullet 1}, \ldots\right] \in \mathbb{R}^{n \times m}$. Then from the linear assumption on f, *i.e.* $\dot{X}_n = f\big(X_n\big) = \boldsymbol{\Theta}_{X_n}\boldsymbol{\beta}$ with $\boldsymbol{\beta} \in \mathbb{R}^{m \times p}$ wherein the q-th row refers to the coefficient vector associated with the candidate functions of the q-th SODE component, we can find a sparse estimate $\hat{\boldsymbol{\beta}}$ by minimizing a data fidelity plus a sparsity term:

$$\hat{\boldsymbol{\beta}} := \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{m \times p}} \left\{ \ell(\dot{X}_n, \boldsymbol{\Theta}_{X_n}\boldsymbol{\beta}) + \lambda R(\boldsymbol{\beta}) \right\} \tag{2}$$

where $\lambda > 0$ is the sparsity amount. We learning of $\boldsymbol{\beta}$ in Fig 1 with a two-dimensional SODE. We present Algorithm.1 to solve Equation (2) when the loss is quadratic and when R is convex or nonconvex with a specific property in Section 3.

In [4, 6, 24] Equation (2) is instantiated with $R := R_{\ell_{1,1}} = \|\cdot\|_{1,1}$ *i.e.* $\hat{\boldsymbol{\beta}}$ is a sparse estimate of the following problem:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{m \times p}} \left\{ \frac{1}{2} \|\dot{X}_n - \boldsymbol{\Theta}_{X_n}\boldsymbol{\beta}\|^2_{2,2} + \lambda \|\boldsymbol{\beta}\|_{1,1} \right\} \tag{3}$$

which is a special case of Equation (2) with $\ell(\cdot, \cdot)$ being the quadratic loss. Since $\boldsymbol{\beta} = \left[\boldsymbol{\beta}_{\bullet 1}, \cdots, \boldsymbol{\beta}_{\bullet p}\right]$, and $R_{\ell_{1,1}}$ acts independently on each entry of $\boldsymbol{\beta}$, solving Equation (3) reduces to p independent LASSO [25] subproblems where each estimates $\boldsymbol{\beta}_{\bullet k}$ from $\left(\dot{\mathbf{x}}_{\bullet k}, \mathbf{x}_{\bullet k}\right)$, for $k = 1 \ldots p$, with the ℓ_1 penalty. Hence, the SODE is discovered using a single-task learning approach.

2.2 Background and limitations of sparse convex single-task regularizers

In [6], the authors formalized the discovery of nonlinear dynamics using partial differential equations, thus their dictionary is built differently from our $\boldsymbol{\Theta}_{X_n}$. The learning criterion used by the author is formalized similarly as in Equation (3) for an SODE. Since Equation (3) is convex in $\boldsymbol{\beta}$, it can be solved by the Douglas-Rachford algorithm [27] which is a proximal-type algorithm [28]. Proximal operators are at the core of sparse learning, hence our proposal. In our context, we use these operators to solve Equation (2) (see Algorithm.1 in Section 3). We recall the general definition.

Definition 2.1. *(proximal operator [28]) The proximal operator associated with a closed, proper and convex function in a Hilbert space, $h : \mathcal{H} \to \mathbb{R}$, is defined for any $\mathbf{y} \in \mathcal{H}$, with $\lambda > 0$ as:*

$$\operatorname{prox}_{\lambda h}(\mathbf{y}) := \arg\min_{\mathbf{u}\in\mathcal{H}} \left\{ \frac{1}{2}\|\mathbf{y}-\mathbf{u}\|_{\mathcal{H}}^2 + \lambda h(\mathbf{u}) \right\}$$

Remark 2.2.1. *Here $\mathcal{H}$ reduces either to $\mathbb{R}^p$ or $\mathbb{R}^{n\times p}$. If h is strongly convex, the minimizer in $\mathcal{H}$ is unique and $\operatorname{prox}_{\lambda h}(\mathbf{u})$ is single-valued [28]. If h is not convex (but remains closed and proper), the proximal operator is still defined well. In this case, since the minimizer set may not reduce to a singleton, the proximal operator can be multivalued.*

When h is also separable, that is for any vector or matrix $\boldsymbol{W}$, $h(\boldsymbol{W}) = \sum_{i,j} h_{ij}(w_{ij})$ with $h_{ij} : \mathbb{R} \to \mathbb{R}$, computing $\operatorname{prox}_{\lambda h}(\boldsymbol{W})$ reduces to compute the proximal operator of h_{ij} for every i, j and then to concatenate $\left\{\operatorname{prox}_{\lambda h_{ij}}(w_{ij})\right\}_{i\leq n, j\leq p}$ according to the dimensions of $\boldsymbol{W}$. In the case of a separable (*i.e.* single-task) vector or matrix regularize, evaluating the proximal operator for a given element is equivalent to evaluating the proximal operator for each of the separable parts. As the $\ell_{1,1}$ norm is separable, with $h_{ij}(w_{ij}) = |w_{ij}|$, we have $\operatorname{prox}_{\lambda|\cdot|}(w_{ij}) = \operatorname{sign}(w_{ij}) \max(0, |w_{ij}| - \lambda)$, which is known as the soft-thresholding operator [25].

2.2.1 Single-task limitation of a separable regularizer

The proximal operator of $R_{\ell_{1,1}}$ serves as a shrinkage operator, assigning zero to coefficients in $\boldsymbol{\beta}$ that do not sufficiently decrease the data fidelity. However, $R_{\ell_{1,1}}$ is separable and hence does not account for any matrix structure. Indeed, we can permute any element within the coefficient matrix $\boldsymbol{\beta}$ for the purposes of learning, and the resulting $R_{\ell_{1,1}}$ remains unchanged. Therefore, for MTL, separability across tasks is not desirable. Consequently, the regularizer acts as if $\boldsymbol{\beta}$ were a vector in $\mathbb{R}^{mp}$. More generally, if a vector structure is

considered for $\boldsymbol{\beta}$ using a fully separable norm regularizer rather than a matrix-structured regularizer, then the task relatedness *i.e.* correlations between columns $[\boldsymbol{\beta}_{\bullet 1}, \cdots, \boldsymbol{\beta}_{\bullet p}]$ is omitted.

2.2.2 The bias induced by a convex regularizer

Definition 2.2. *Let $\hat{\beta}$ be an estimate of the ground-truth coefficient β, the bias is the expectation over the distribution of $\hat{\beta}$ of the estimation error thus defined as* $\text{bias}(\hat{\beta}) = \mathbb{E}(|\hat{\beta} - \beta|)$. [1]

Using triangle inequality, one can show that a regularizer inducing a norm in $\mathbb{R}^{m \times p}$ is convex and thus induces a bias toward zero in the learned coefficients [29, p. 73], which in our context degrades the SODE identification.

Example 2.2.1. *Let us consider the proximal operator of the $\ell_{1,1}$ norm. An estimate of a true coefficient β_{ik} is $\hat{\beta}^n_{ik} = \text{prox}_{\lambda|\cdot|}(\hat{w}^n_{ik}) = \text{sign}(\hat{w}^n_{ik}) \max(0, |\hat{w}^n_{ik}| - \lambda)$, where $\hat{w}^n_{ik}$ is an unregularized estimate of β_{ik} (i.e. preceding the proximal step, see Algorithm.1) from n samples. Hence as $n \to \infty$:*

- *for $|\beta_{ik}| \in [0;\ \lambda]$, $\text{bias}(\hat{\beta}^n_{ik}) \to \beta_{ik}$,*
- *for $|\beta_{ik}| \in]\lambda;\ \infty]$, $\text{bias}(\hat{\beta}^n_{ik}) \to \lambda$.*

Consequently, if the i-th candidate function is relevant to the k-th equation of the SODE, the estimate $\hat{\beta}^n_{ik}$ is necessarily biased. In Fig 2 (right), it is clear that the soft-thresholding operator never reaches the identity function and, therefore, always returns a bias estimate of β_{ik}. Correcting the bias with λ is equivalent to applying the hard-thresholding operator [30], which is the proximal operator of the ℓ_0 penalty (the nonzero indicator), and thus is unbiased, but makes the objective function of Equation (2) NP-hard since instantiating $R(\boldsymbol{\beta}_{ik}) := \mathcal{I}(\beta_{ik} \neq 0)$ makes the objective function discontinuous.

2.3 A nonconvex separable regularizer

Here we introduce the single-task version of a nonconvex regularizer: the smoothly clipped sbsolute deviation (SCAD) [10] which induces unbiasedness for large coefficients and serves as a building block for our proposed regularizer that we introduce in Section 3.

Definition 2.3. *Let $\lambda > 0$ and $\theta > 2$ be hyperparameters that operate as the level of sparsity and unbiasedness, respectively; the SCAD is defined for $w \in \mathbb{R}$ as:*

$$r^{SCAD}_{\lambda,\theta}(w) = \begin{cases} \lambda|w| & \text{if } |w| \leq \lambda \\ -\frac{\lambda^2 - 2\theta\lambda|w| + w^2}{2(\theta - 1)} & \text{if } \lambda < |w| \leq \theta\lambda \\ \frac{(\theta+1)\lambda^2}{2} & \text{if } |w| > \theta\lambda \end{cases} \tag{4}$$

[1] This is not the conventional definition of estimation bias, which is $\mathbb{E}(\hat{\beta}) - \beta$, but note that from Jensen inequality, our definition is an upper bound of the latter quantity.

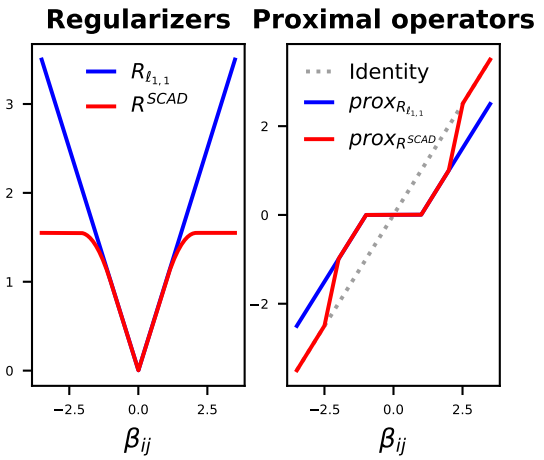


Fig. 2: Left: The (convex) LASSO ($R_{\ell_{1,1}}$, blue) and (nonconvex) SCAD ($R^{SCAD}_{\lambda,\theta}$, red) regularizers, with $\lambda = 1, \theta = 2.01$. Right: Their associated proximal operator as a function of a coefficient β_{ij}. In the right plot, we can see hat the soft-thresholding operator $\text{prox}_{\lambda R_{\ell_{1,1}}}(\beta_{ij})$ (blue) induces a bias since it thresholds β_{ij} toward zero and away from the identity (gray dotted lines), which represents the best unbiasedness for a true nonzero coefficient. In contrast, $\text{prox}_{r^{SCAD}_{\lambda,\theta}}(\beta_{ij})$ (red), continuously reaches the identity as $|\beta_{ij}|$ increases, meaning that SCAD induces less bias than the LASSO, as a true coefficient is high.

Remark 2.3.1. *$r^{SCAD}_{\lambda,\theta}$ is semiconcave (i.e. concave with respect to $|w|$).*

- *For $|w| \leq \lambda$, the SCAD penalty behaves as ℓ_1.*
- *For $\lambda < |w| \leq \theta\lambda$, the SCAD interpolates quadratically between ℓ_1 and ℓ_0 and the transition between them is controlled by θ.*
- *For $|w| > \theta\lambda$, the penalty level is constant, thus the SCAD behaves as ℓ_0.*

SCAD is illustrated in Fig 2 (left).

Remark 2.3.2. $\lim_{\theta\to\infty} r^{SCAD}_{\lambda,\theta}(w) = \lambda|w|$*, thus the SCAD induces ℓ_1 sparsity as the upper limit with respect to θ.*

Remark 2.3.3. $\lim_{\theta\to 2,|w|\leq 2\lambda} r^{SCAD}_{\lambda,\theta}(w) = \lambda|w|$ *and* $\lim_{\theta\to 2,|w|>2\lambda} r^{SCAD}_{\lambda,\theta}(w) = \lambda\mathcal{I}(w \neq 0)$*, thus the SCAD induces ℓ_1 sparsity for small $|w|$ and ℓ_0 sparsity (unbiased but discontinuous) for large $|w|$, as the lower limit with respect to θ.*

For any $w \in \mathbb{R}$, $\text{prox}_{r^{SCAD}_{\lambda,\theta}}(w)$ is known analytically [10] (see the Appendix.B) and, as plotted in Fig 2 (right), since it reaches the identity line, induces less bias than $\text{prox}_{\lambda|\cdot|}(w)$ for large coefficients, *i.e.* as $|w| > 2\lambda$.

The matrix-extended version of the SCAD (4) is $R^{SCAD}_{\lambda,\theta}(\boldsymbol{W}) = \sum_i \sum_j r^{SCAD}_{\lambda,\theta}(w_{ij})$, which is also separable. Hence, $\text{prox}_{R^{SCAD}_{\lambda,\theta}}(\boldsymbol{W})$ is the concatenation of $\left\{\text{prox}_{r^{SCAD}_{\lambda,\theta}}(w_{ij})\right\}_{i\leq n, j\leq p}$ along the dimensions of $\boldsymbol{W}$. However, as a separable regularizer cannot account for task relatedness, the only benefit of learning with $R^{SCAD}_{\lambda,\theta}$ over $R_{\ell_{1,1}}$ is unbiasedness. Indeed, the two summands in $R^{SCAD}_{\lambda,\theta}$ act similarly at the row and column levels of $\boldsymbol{\beta}$, and therefore, do not resolve the shortcomings mentioned in Section 2.1.

2.4 Building block for MTL in linear regressions

MTL consists of learning p functions $\left[f_1, \cdots, f_p\right]$ jointly by assuming that they share a common set of features [31]. For the k-th task, a dataset $\{\mathbf{y}_{ik};\ z_{ik}\}_{i\leq n_k}$ of n_k samples with m features is given. Therefore, the p regression coefficient vectors can be represented in a matrix $\boldsymbol{\beta} \in \mathbb{R}^{m\times p}$. The task similarity is reflected in the learning criterion by a regularizer R applied to this matrix. For p linear regressions, MTL is formulated as:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}:=[\boldsymbol{\beta}_{\bullet 1},\cdots,\boldsymbol{\beta}_{\bullet p}]} \Big\{ \sum_{k=1}^{p} \sum_{i=1}^{n_k} \frac{1}{2n_k} (z_{ik} - \mathbf{y}_{ik}^T \boldsymbol{\beta}_{\bullet k})^2 + \lambda R(\boldsymbol{\beta}) \Big\} \tag{5}$$

where R may account for task relatedness. Equation (5) is a special case of Equation (2) and can be solved with Algorithm.1. Note that when $n_k = n$ and $R = R_{\ell_{1,1}}$, Equation (5) reduces to Equation (3). Thus, by choosing a regularizer that is more appropriate than the $\ell_{1,1}$ norm for considering task relatedness, discovering an SODE can be formulated as MTL.

2.4.1 Considering task relatedness

To account for task relatedness, the regularizer has to be matrix-structured, for instance, solving Equation (5) with $R_{\ell_{2,1}}(\boldsymbol{\beta}) := \|\boldsymbol{\beta}\|_{2,1} = \sum_i^m \|\boldsymbol{\beta}_{i\bullet}\|_2$ (*i.e.* group-lasso [11]), makes $\boldsymbol{\beta}$ row-sparse *i.e.* some rows are identically nonzero and all the others are null [32], due to the nonseparability with respect to $\boldsymbol{\beta}_{i\bullet}$ and sparsity with respect to $\boldsymbol{\beta}_{\bullet j}$. To discover an SODE, $R_{\ell_{2,1}}$ forces all the equations to share the same candidate functions of $\boldsymbol{\Theta}_{X_n}$. The proximal operator associated with $R_{\ell_{2,1}}$ is given in the Appendix.B.

2.4.2 Considering task-specific elements

Although an SODE shares some candidate functions across its equations, it is sufficiently flexible to allow specific candidate functions for each equation. This can be achieved by taking a convex combination of two norms *i.e.* $R_{\ell_{2,1}+\ell_{1,1},\alpha}(\boldsymbol{\beta}) := \alpha\|\boldsymbol{\beta}\|_{2,1} + (1-\alpha)\|\boldsymbol{\beta}\|_{1,1}$ with $\alpha \in [0,1]$ [12]. For $\alpha > 0.5$, a greater level of importance is given to commonality in candidate functions across equations compared to specificity, and conversely, for $\alpha < 0.5$. In practice, α is chosen by cross validation. The proximal operator associated with

$R_{\ell_{2,1}+\ell_{1,1},\alpha}$ is given in Appendix.B. For the sake of clarity, we omit the index α in the notation and write $R_{\ell_{2,1}+\ell_{1,1}}$.

2.4.3 Limitations of sparse convex MTL regularizers

Despite being able to select relevant candidate functions, the convexity of $R_{\ell_{2,1}}$ and $R_{\ell_{2,1}+\ell_{1,1}}$ induces a bias, similarly to $R_{\ell_{1,1}}$, in $\hat{\boldsymbol{\beta}}$ through their associated proximal operator [12].

In Section 2.1 and Section 2.4, task relatedness and nonconvexity are shown as being two important weaknesses that cannot be addressed by the $\ell_{1,1}$ regularizer. Our contribution harnesses both the nonconvexity and the task relatedness within a single regularizer to improve the discovery of an SODE.

3 A nonconvex matrix-structured regularizer

In this section, we introduce our proposal, a regularizer, which jointly accounts for task relatedness, sparsity and unbiasedness. Then we introduce a generative, iterative, thresholding learning algorithm [33] that can be instantiated with all the regularizers presented in this paper.

3.1 A nonconvex nonseparable regularizer

We propose to "un-separate" $R^{SCAD}_{\lambda,\theta}$ to unharness both from the nonseparability and the nonconvexity of the SCAD. The key technique is to replace the second summation in $R^{SCAD}_{\lambda,\theta}$ by $r^{SCAD}_{\lambda,\theta}(\|\mathbf{w}_{i\bullet}\|_1)$:

$$R^{SCAD-\ell_1}_{\lambda,\theta}(\boldsymbol{W}) := \sum_{i=1}^{m} r^{SCAD}_{\lambda,\theta}\big(\|\mathbf{w}_{i\bullet}\|_1\big) \tag{6}$$

In this way, as the regularizer acts on the coefficient vector of the i-th candidate function using the SCAD of the ℓ_1 norm, $R^{SCAD-\ell_1}_{\lambda,\theta}$ forces $\mathbf{w}_{i\bullet}$ to be sparse, unbiased and correlated across the p tasks. Another way of understanding the rationale behind our proposal is to see the hierarchical sparsity. The first level (the "highest") is the task-level: we enforce the set of nonzero coefficients to be the same across each task. Such a sparsity level is somehow a group sparsity and enforces the learning to be multitask. The second level (the "lowest") is the component level: for a given task, we allow some (small) coefficients to be both nonzero and specific to each task. It is important to note the analytical similarity with $R_{\ell_{2,1}} = \sum_{i=1}^{m}\|\mathbf{w}_{i\bullet}\|_2$ (Section 2.4), which does not allow each equation of the SODE to have a specific candidate function. Contrary to $R_{\ell_{2,1}}$, our proposal jointly enforces unbiasedness and sparsity in each $\mathbf{w}_{i\bullet}$, and thereby enables the equations of the SODE to have specific candidate functions.

Despite $R^{SCAD-\ell 1}$ being nonconvex, it is closed and proper by construction. Thus the image of the proximal operator of our regularizer is nonempty and we can compute it analytically as follows: for any $\boldsymbol{W} \in \mathbb{R}^{m\times p}$, $\lambda > 0$,

$\theta > 2$ and $(i, q) \in \{1, \ldots, m\} \times \{1, \ldots, p\}$, let $\tilde{w}_{iq}$ be the element of the i-th row and q-th column of the matrix $\widetilde{W}$ returned by the proximal operator,

$$\tilde{w}_{iq} := \begin{cases} \operatorname{sign}(w_{iq}) \max(0, |w_{iq}| - \lambda) & \text{if } |w_{iq}| \leq B_i^1, \\ \dfrac{\theta - 1}{\theta - 2} \operatorname{sign}(w_{iq}) \max(0, |w_{iq}| - \dfrac{\theta\lambda}{\theta - 1}) & \text{if } B_i^1 < |w_{iq}| \leq B_i^2, \\ w_{iq} \quad \text{if } |w_{iq}| > B_i^2 \end{cases} \tag{7}$$

with $B_i^1 := 2\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1$, $B_i^2 := \theta\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1$ and $\mathbf{w}_{i\bullet,-q}$ denotes all the elements except the q-th of the vector $\mathbf{w}_{i\bullet}$. Knowing this proximal operator is a computational benefit in solving Equation (2). We give a proof detailing the computational steps to obtain the above closed form in the Appendix.A.

Remark 3.1.1. *In the first part,* $(\leq B_i^1)$, $\operatorname{prox}_{R_{\lambda,\theta}^{SCAD-\ell_1}}(\cdot)$ *equals* $\operatorname{prox}_{R_{\ell_{1,1}}}(\cdot)$ *(soft-thresholding operator). In the second part,* $\operatorname{prox}_{R_{\lambda,\theta}^{SCAD-\ell_1}}(\cdot)$ *acts as a (group) soft-thresholding operator but with a rate* $\frac{\theta-1}{\theta-2}$ *which is greater than* 1*, thus outputs unbiased coefficient vectors. In the third part,* $\operatorname{prox}_{R_{\lambda,\theta}^{SCAD-\ell_1}}(\cdot)$ *acts as a (group) hard-thresholding operator, returning* $\tilde{\mathbf{w}}_{i\bullet}$ *identically.*

Remark 3.1.2. *Since* B_i^1 *and* B_i^2 *are only row dependent, each row of* $\operatorname{prox}_{R_{\lambda,\theta}^{SCAD-\ell_1}}(\boldsymbol{W})$ *can be computed in parallel.*

3.2 MTL with a nonconvex regularizer

Algorithm 1 GISTA to solve Equation (2)

Input: data samples $\dot{X}_n$ and X_n, , $R(\cdot)$, $\lambda > 0$, $\theta > 2$ (for SCAD$-\ell_1$ and SCAD), $1 > \alpha > 0$ (for $R_{\ell_{2,1}+\ell_1}$), initial guess $\boldsymbol{\beta}_0$, step size γ

build $\boldsymbol{\Theta}_{X_n}$ from X_n
$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta}_0$, $\gamma \leftarrow \gamma_{min}$, $\boldsymbol{G} \leftarrow \boldsymbol{\Theta}_{X_n}^T \boldsymbol{\Theta}_{X_n}$, $\boldsymbol{B} \leftarrow \boldsymbol{\Theta}_{X_n}^T \dot{X}_n$
$\boldsymbol{W} \leftarrow \boldsymbol{\beta} - \gamma_{min}(\boldsymbol{G}\boldsymbol{\beta} - \boldsymbol{B})$
while $\boldsymbol{\beta}$ has not converged **do**
    while line search criterion is unsatisfied **do**
        $\boldsymbol{\beta} \leftarrow \operatorname{prox}_{\gamma\lambda R}(\boldsymbol{\beta})$
        $\gamma \leftarrow 0.8\gamma$
    end while
    $\boldsymbol{W} \leftarrow \boldsymbol{\beta} - \gamma(\boldsymbol{G}\boldsymbol{\beta} - \boldsymbol{B})$
    $\boldsymbol{\beta} \leftarrow \operatorname{prox}_{\gamma\lambda R}(\boldsymbol{W})$
end while

Output: $\boldsymbol{\beta}$

We instantiate the general iterative shrinkage thresholding algorithm (GISTA) [33] in Algorithm 1 to perform learning. The GISTA can be instantiated with regularizers either convex, or nonconvex and expressed as a difference of two convex (DC) functions, see [34–36] for examples of such regularizers. It generalizes FISTA [37] whose convergence is guaranteed when minimizing quadratic plus convex regularizers functions. Despite the nonconvexity of DC functions, convergence guarantees to a stationary point in finite time of the GISTA are established in [33]. $R^{SCAD-\ell_1}$ and R^{SCAD} enjoy the DC property, thus we can use the GISTA to learn with any regularizers presented in this paper. Convergence is not ensured for other nonconvex penalties, such as the group bridge penalty $R_{\ell_{q,r}}(0 < q, r < 1)$, since their associated proximal operator does not give rise to a closed-form expression. Hence, to compute the proximal operator the bridge penalty in Algorithm 1, an inner nonconvex optimization procedure is necessary and is likely to affect the convergence of GISTA.

GISTA consists of two nested loops. The outer loop (lines 4–12) consists of a loss gradient descent step (line 9) followed by a proximal step (line 10) that set to zero the coefficients that insignificantly decrease the loss term. The inner loop (lines 5–8) consists of the backtracking line search to compute the gradient step size γ, that given the current descent direction, ensures a sufficient decrease [29]. The value 0.8 (line 7) is typical and corresponds to the 'slow-rate' parameter [29]. To limit the cost of one iteration, we precomputed $\boldsymbol{\Theta}_{X_n}^T \boldsymbol{\Theta}_{X_n}$ and $\boldsymbol{\Theta}_{X_n}^T \dot{X}_n$ (line 2). A detailed complexity analysis of proximal gradient algorithms, such as GISTA, is given in [38].

4 Numerical experiments

4.1 Experimental setting

4.1.1 Synthetic datasets generated from known SODEs

We evaluated our approach on three SODEs (*i.e.* ground-truth) known from the literature [4, 7, 39]: the damped oscillator with cubic dynamic (DOC) used to model the nonlinear pendulum in mechanics, the Lotka-Volterra (LV) system used as prey-predator interaction model and the Lorenz attractor (LAT) used to model atmospheric turbulence. Each one of these SODEs has common functions as well as specific other functions across their equations.

We base our experiments on those in [7].

Simulation of the state variables

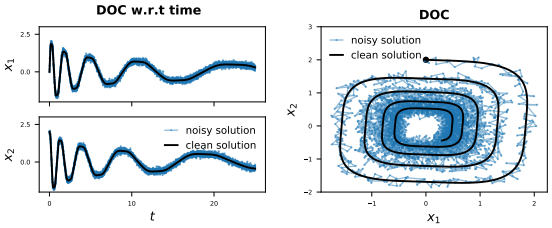
We generated the state variables, $x_1, \ldots, x_p$, along time (*i.e.* multivariate time series) by numerically solving the true SODEs with an implicit backward differentiation formula method of order five, implemented in the SCIPY library [40]. For each SODE, the resulting time series has $n = 5 \times 10^3$ time steps.

In a real life scenario the data are noisy so we added white noise to the clean time series. We experimented with Gaussian noise, by varying the variance

(a) DOC ($p = 2$ tasks)

$\mathbf{x}(t = 0) = [0, 2]$, for $t \in [0;\ T_{DOC} = 25]$:

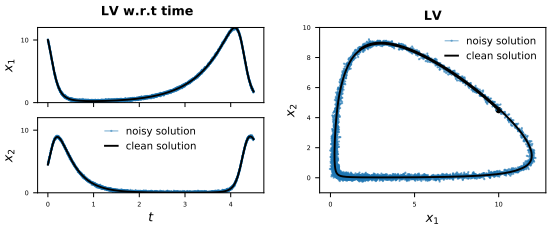
$$\begin{cases} \dot{x}_1(t) = -0.1x_1^3(t) + 2x_2^3(t) \\ \dot{x}_2(t) = -2x_2^3(t) - 0.1x_1^3(t) \end{cases}$$



(b) LV ($p = 2$ tasks)

$\mathbf{x}(t = 0) = [10, 4.5]$, for $t \in [0;\ T_{LV} = 5]$:

$$\begin{cases} \dot{x}_1(t) = 1.5x_1(t) - x_1(t)x_2(t) \\ \dot{x}_2(t) = -3x_2(t) + x_1(t)x_2(t) \end{cases}$$



(c) LAT ($p = 3$ tasks)

$\mathbf{x}(t = 0) = [-5, 1, 20]$, $\sigma = 10$, $\rho = 28$, $\beta = \frac{8}{3}$, for $t \in [0;\ T_{LAT} = 10]$:

$$\begin{cases} \dot{x}_1(t) = -\sigma x_1(t) + \sigma x_2(t) \\ \dot{x}_2(t) = \rho x_1(t) - x_2(t) - x_1(t)x_3(t) \\ \dot{x}_3(t) = -\beta x_3(t) + x_1(t)x_2(t) \end{cases}$$

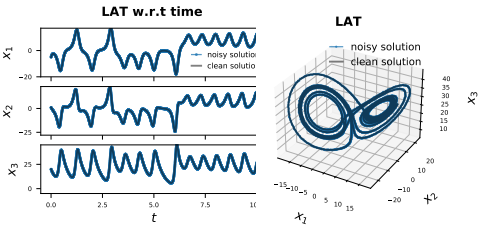


Fig. 3: Left: the ground-truth SODEs used in our experiments. Center: the numerical solution (black), *i.e.* the state variables $x_1, \ldots, x_p$ plotted w.r.t time. The training data (blue) corresponds to a noisy version (in these plots the noise is Gaussian) of the numerical solution. Right: same data as in the center plotted as one state variable w.r.t the other(s) to visualize their correlation.

level in terms of logarithmic signal-to-noise ratio (log-SNR). The log-SNR is defined as:

$$\text{log-SNR} := 10 \log_{10} \left(\frac{\sum_{j=1}^{p} \sigma^2_{ground-truth,k}}{\sigma^2_{noise}} \right) \tag{8}$$

where $\sigma^2_{ground-truth,k}$ is the variance in the k-th state variable (without noise) and σ^2_{noise} is the variance in the noise. This enables a comparison of the results across synthetic datasets with an equal amount of noise w.r.t the variance in the ground-truth. The lower the log-SNR, the higher the noise variance. In our experiments, for each SODE we fixed the log-SNR to $\{20, 30, 40, 50\}$ and then deducted σ^2_{noise}. Samples of the clean and noisy time series (for a log-SNR equal to 30) are plotted in Fig 3 for each SODE. Since the noise may contain outliers in real life, we also experimented with a Student-t noise with five degrees of freedom (maximum kurtosis) *i.e.* the tail distribution is the heaviest.

Building the dictionary

The dictionary of candidate functions was built sufficiently large, involving both monomial and interaction terms $(x_1x_2, x_1^2x_2)$ that can span polynomials up to degree three. It was computed from the noisy time series, hence the regression covariates were also corrupted.

Estimation of the derivative

We experimented with three methods to estimate the time derivative of the state variables from the noisy time series: the finite-difference (FD) method, the spectral (SP) method and the B-spline (BSP) method.

The FD method is the most basic derivative estimation method as it consists approximating the derivative pointwisely as follows $\dot{y}(t_j) = \frac{y(t_{j+1})-y(t_{j-1})}{2}$. It is computationally inexpensive but sensitive to noise [9, ch. 5].

The SP method relies on the following property: let $\mathcal{F}_y(f)$ be the Fourier transform (FT) of the differentiable function y at frequency $f \in \mathbb{R}$, then $\mathcal{F}_{\frac{\mathrm{d}y}{\mathrm{d}t}}(f) = 2\pi i f \mathcal{F}_y(f)$. In other words, the derivative can be estimated by taking the inverse FT of the original time series multiplied by a linear factor. Thus, by using this property, it benefits from the low computational cost of the inverse Fast FT algorithm [41] for estimating the derivative from the raw data. Since noise measurement resides in high frequencies, the FT spectrum is low-pass filtered before computing its inverse transform. We experimented three filter sizes.

The BSP method, widely used in functional data analysis [42], consists of two steps. First, approximate the time series with a weighted linear combination of BSPs *i.e.* piecewise polynomial functions, usually of degree three or four. The weights are computed by minimizing standard least-square criteria as the data fidelity term. To balance between smoothness and data fitting error, the minimization is stopped when the error reaches a fixed amount s (the higher, the smoother the approximation). We experimented with six values of $s \in \{5, 10, 50, 100, 500, 1000, 5000\}$. Second, knowing the analytical form of each BSP, the derivative is computed from the approximated function as a linear combination of the BSPs' derivative. See [42, ch. 4,5] and [43] for a

detailed explanation of penalized curve smoothing and an application to outlier detection in functional data analysis, respectively.

4.1.2 Real data

As an example of a real-world application, we applied our approach to discover the dynamics of a laboratory-based ecological phenomenon whose SODE is unknown [44]. In this system, algae of genus Chlorella is grown in a large glass test tube (chemostat) to which a nutrient-rich medium is continuously added, and from which the contents are removed (including the algae) at a constant rate. The growth of the algae population is limited by nutrition in the ecology and by predation by the rotifer, Brachionus, a genus of microscopic animals. The rotifers reproduce according to how much algae they consume, and die either from natural causes or when they are removed from the tank. Hence, there might be a predator-prey dynamic between these two quantities [9]. The goal is to discover the SODE that governs the dynamics underlying the growth rate of the algae and the rotifers.

The dataset (abbreviated Chemostat data) consists of a bivariate time series with 108 daily time steps. The first state variable, x_1, is the Chlorella concentration, and the second state variable, x_2, is the Brachionus concentration. As recommended in the analysis of [9], we preprocessed the time series by approximating them with 103 cubic BSP (using the R packages FDA [45]). Then we reconstructed the time series by evaluating the approximation function on a regular grid of 5×10^3 time steps in the interval $[7;\ 114]$. The resulting reconstruction can be considered nonnoisy, so we estimate the derivative with the FD method. We built the dictionary with monomial functions up to degree five, with first, second and third order interactions $x_1x_2, x_1^2x_2, x_1^3x_2$.

4.2 Setting for the GISTA

We followed the settings of the original GISTA paper [33]. We initialized the algorithm with $\boldsymbol{\beta}_0 = \mathbf{0}$ and set the step size $\gamma = 2$. We stopped the outer loop (lines 4-11 Algorithm.1) when the iteration number exceeded 10^3. We stopped the inner loop when the objective function of Equation (2) was below its quadratic approximation or the number of inner iterations exceeds 10. The hyperparameters λ, θ (for $R^{SCAD-\ell_1}_{\lambda,\theta}$ and $R^{SCAD}_{\lambda,\theta}$) and α (for $R_{\ell_{2,1}+\ell_1}$) were selected with a time based five fold cross validation: each training/testing fold consists in successive samples of the state-variables (for a given fold, the last training sample "immediatly preceeds" the first testing sample). The training folds are of increasing size (25%, 40%, 55%, 70% and 85% of the whole dataset) and, the testing folds do not overlap and have a fixed size (15% of the whole dataset). Moreover, this strategy enables to asses the effect of the number of training time steps on the SODE recovery.

4.3 Comparison with baselines

We compare our proposal $R_{\lambda,\theta}^{SCAD-\ell_1}$ to four baselines $R_{\ell_{1,1}}$, $R_{\ell_{2,1}}$, $R_{\ell_{2,1}+\ell_{1,1}}$ and $R_{\lambda,\theta}^{SCAD}$. To fairly compare the effects of the baseline regularizers on the learned SODEs, all were instantiated them within GISTA.

We compared the learned SODEs with three metrics: $\epsilon_\beta = \frac{\|\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^*\|_{2,2}^2}{\|\boldsymbol{\beta}^*\|_{2,2}^2}$, $\epsilon_T = \frac{\sum_{i=1}^n \|\hat{\mathbf{x}}(t_i)-\mathbf{x}^*(t_i)\|_2^2}{\sum_{i=1}^n \|\mathbf{x}^*(t_i)\|_2^2}$ [2] and $\epsilon_{MIS} = \frac{\sum_{i,j} \mathcal{I}(\hat{\beta}_{ij}\neq\beta_{ij}^*)}{\sum_{i,j} \mathcal{I}(\beta_{ij}^*\neq 0)}$.

ϵ_β measures the relative bias w.r.t the ground-truth coefficient matrix $\boldsymbol{\beta}^*$. ϵ_T measures the relative squared error, along t, of the numerical solution for the learned SODE w.r.t to the numerical solution of the ground-truth SODE. ϵ_{MIS} measures the rate of misidentified candidate functions. The lower ϵ_β, ϵ_T and ϵ_{MIS} are, the better the recovery of the SODE. Since the SODE is unknown for the Chemostat, note that only ϵ_T, wherein $\mathbf{x}^*$ refers to the learning data (preprocessed), can be computed.

4.4 Implementation

The GISTA was implemented in the Python library LIGHTNING [46] which is a SCIKIT-LEARN compatible interface for linear regression and classification. The experiments were run in parallel on a cluster equiped with 24 Intel Xeon-Gold-6136 3GHz processors, each one holding 192Go RAM. The whole process, *i.e.* building the dictionary, derivative estimation, hyperparametrs selection and earned SODE simulation, is implemented within the Python API PYSINDY [47][3].

4.5 Results

4.5.1 Synthetic datasets

For each of these SODEs, we repeated the experiment five times. In Fig 4, 5 and 6 we report the average and standard deviations of ϵ_β, ϵ_T and ϵ_{MIS}. We see that for all the experiments the bias is reduced with R^{SCAD} and our proposal $R^{SCAD-\ell_1}$ (see purple curves and vertical bars for ϵ_β). (given in the supplementary material as it results in many figures)

Effect of the derivative estimation methods

As can be seen in Fig 4, 5 and 6, the variations in the three error metrics are high when the derivative is estimated with the FD method. That (unsurprisingly) tells us that this method is highly sensitive to noise. The SP method gives the worst results, which we explain with the non-stationarity of the time series resulting from the SODE. Indeed, only a part of the signal is seen during the time based cross validation, thus the Fourier spectrum of the training part might be a poor estimate of the whole ground-truth. The filter size selection

[2] $\hat{\mathbf{x}}(t_i)$ is the solution to the learned SODE at t_i and $\mathbf{x}^*(t_i)$ is the solution to the ground-truth SODE at t_i.

[3] The code is accessible from https://github.com/Clej/Unbiased-SODE-discovery.

is noise sensitive, and consequently, in the Fourier sepctrum, some ground-truth frequencies which are close to the noise frequencies might be abnormally removed, resulting in a poor derivative estimate. Therefore, the SP method is unreliable for derivative estimation used with (nonstationary) time based cross validation. When using BSP instead, the errors are quite stable against noise. This makes it a good candidate for inferring the derivative when the practitioner does not have prior knowledge about the noise measurement level.

Effect of the noise level and type

We observe that for low log-SNR values, learning with $R^{SCAD-\ell_1}$ in conjunction with the FD method is similar to learning with the baseline regularizers. The same observation holds when using the BSP method. However, as the log-SNR increases (*i.e.* the noise level decreases), the error decays are the highest with $R^{SCAD-\ell_1}$ whereas the decay is lower (or remains constant) with the convex regularizers. Also, note that with the BSP method, learning with convex regularizers increases ϵ_{MIS} w.r.t the log-SNR whereas with the nonconvex R^{SCAD} and $R^{SCAD-\ell_1}$, it remains constant. Overall the Gaussian noise experiments suggest that with both the FD and BSP methods, and for moderate log-SNR levels ≥ 30, (that means when dealing with high-quality samples of the state variables), the SODEs are best recovered with our proposal. The experiments with the Student-t noise partially confirm our observations made about the effect of the derivative estimation method. Since for the convex regularizers and R^{SCAD}, the errors are quite variable across the three SODEs, the FD method is not robust against noise (here heavy-tailed). Indeed, the errors are low with the LV SODE (see the first row of Fig 5) and high with the DOC and LAT SODEs. Note however that, $R^{SCAD-\ell_1}$ is more robust than the baselines against the Student-t noise. The three errors are almost the highest with the SP method for the three SODEs, hence we confirmed that it is not suited to derivative estimation for SODE discovery. When the SODE is learned with the nonconvex regularizers, the BSP method gives much better results than both the FD and SP methods. Overall, the Student-t noise experiments suggest that the SODEs are better recovered with nonconvex regularizers.

Effect of the multitask consideration

We now compare the results of the multitask based regularizers ($R_{\ell_{2,1}}$, $R_{\ell_{2,1}+\ell_{1,1}}$, $R^{SCAD-\ell_1}$) w.r.t the single-task-based ones ($R_{\ell_{1,1}}$, R^{SCAD}). First, we analyze the results of DOC experiments, in which the SODE has the same terms across across its equations (see (a) Fig 3). Hence, among the baselines, since $R_{\ell_{2,1}}$ enforces the equations to share the same terms, it should better recover the ground-truth SODE than the single-task-based ones. This is confirmed by our results since the single-task-based regularizers (blue and red in Fig 4) are outperformed by $R_{\ell_{2,1}}$ and $R^{SCAD-\ell_1}$ (orange and purples curves). We note however that the decay of ϵ_T with $R_{\ell_{2,1}}$ is similar to $R^{SCAD-\ell_1}$, and the latter gives lower errors. Similarly, we analyze the results of the LV experiments, in which the SODE (see (b) Fig 3) has one common and one

specific term across its two equations. In this context, we expect $R_{\ell_{2,1}+\ell_{1,1}}$ to recover the ground-truth SODE well. This is partly confirmed by our results, we see that $R_{\ell_{2,1}+\ell_{1,1}}$ performs similarly to $R_{\ell_{1,1}}$, but both are outperformed by $R^{SCAD-\ell_1}$ (blue and green in Fig 5). The decays of ϵ_T are also similar and $R^{SCAD-\ell_1}$ is slightly better. Finally and equivalently, we analyze the results of the LAT experiments in which the SODE has approximately the same number of shared and specific terms (see (c) Fig 3). For this kind of SODE, we again expect $R_{\ell_{2,1}+\ell_{1,1}}$ to perform best. This is clearly not the case as both R^{SCAD} and $R^{SCAD-\ell_1}$ behave similarly (red and purple in Fig 6), the former being slightly better. As for the LV SODE, $R_{\ell_{2,1}+\ell_{1,1}}$ and $R^{SCAD-\ell_1}$ show similar results and $R_{\ell_{2,1}}$ gives the worst results. Overall, we see that task relatedness is an important feature in SODE discovery and that our proposal improves it. Although $R_{\ell_{2,1}+\ell_{1,1}}$ was designed to trade off between task relatedness and task specificity, the balance is driven by a hyperparameter whose selection is sensitive. Contrarily, by construction, our proposal is shown by our experiments to be adaptive to the balance between task relatedness and task specificity.

Effect of the number of training time steps

We remind that in time based cross validation, each training/testing fold consists in successive samples of the observed time series, the training folds are of increasing size and the size of the testing folds is fixed. Hence, the number of training time steps increases w.r.t the fold index. The testing folds do not overlap but the training ones do. Consequently, as mentioned in Section 4.2, time based cross validation enables to assess the relationship between the recovery errors and the number of training time steps. We assess this effect on the three synthetics SODEs with a log-SNR value of 30. For that, we train the models (one for each regularizer) on each (training) fold and then we compute the three errors metrics on the associated testing folds. We report the results on Fig 7, 8, 9. From these figures, we confirm that as the fold number index increases, the errors decrease or remain stationary, with an exception for the spectral derivative estimator. We note that the highest error decays occur with $R^{SCAD-\ell_1}$ and R^{SCAD}. The convex baselines are similar.

Statistical assesment of the experimental results

Regularizer	$\epsilon_\beta(0.40)$	$\epsilon_{MIS}(0.40)$	$\epsilon_T(0.40)$
$R_{\ell_{1,1}}$	3.66	3.33	3.26
$R_{\ell_{2,1}}$	3.35	4.04	3.02
$R_{\ell_{2,1}+\ell_{1,1}}$	3.46	3.31	3.51
R^{SCAD}	2.33	2.41	2.75
$R^{SCAD-\ell_1}$	**2.17**	**2.10**	**2.54**

Table 2: Mean rank of each regularizer w.r.t the three error metrics. Second row, in parenthesis, is the critical distance, see [48] for details. Bold refers to the best rank.

We assess the relevance of our conclusion with a two step statistical hypothesis test. We apply the method of Demsar [48]. The first step aims to confirm that not all the regularizers are similar, *i.e.* at least one is different w.r.t the other ones. This step is accomplished with the non-parametric Friedman test. If the latter concludes that at leat one regularizer is different from the others, then in the second step all the regularizers are compared pairwisely through their mean rank: if the difference between two average ranks is greater than what is called a "critical distance" statistic, the two regularizers are said to be different. The second step is done with the Nemenyi post-hoc test. See [48] for more details. For any hypothesis tests, we set the p-value to 5%. We repeat the whole procedure for each one of the three error metrics on all the results (derivative estimator, log-SNR value, etc). The null hypothesis of the Friedman test is rejected with probability of error 10^{-40} for ϵ_β, 10^{-86} for ϵ_{MIS} and 10^{-14} for ϵ_T, thus there is a difference between the regularizers. We report the mean ranks in Table 2. It confirms our empirical comparisons made in the previous paragraphs.

4.5.2 Chemostat dataset

We show the numerical solution, as well as the analytic form, of the SODEs learned for each regularizer in Fig 10(a) and Fig 10(b) respectively. As for the synthetic datasets, the results show that learning with $R^{SCAD-\ell_1}$ improves the recovery performance (smallest ϵ_T in Table 3) compared to R^{SCAD} and the convex baselines.

Regularizer	ϵ_T
$R_{\ell_{1,1}}$	35.0
$R_{\ell_{2,1}}$	30.0
$R_{\ell_{2,1}+\ell_{1,1}}$	30.0
R^{SCAD}	20.0
$R^{SCAD-\ell_1}$	**4.6**

Table 3: ϵ_T for the Chemostat SODE learned with the four baseline regularizers and our proposal $R^{SCAD-\ell_1}$.

We interpret the dynamics of the underlying biological phenomenon by examining the closed-form SODE discovered with $R^{SCAD-\ell_1}$ (Fig 10(b) last column). Based on the analysis carried out in [9] and noting the strong similarity with the LV model (*i.e.* only linear and same first-order interaction terms in the two equations), we can interpret the discovered SODE as a predator-prey model as follows: within both equations, we observe the presence of a first-order interaction term x_1x_2 that models the rate (decreasing or increasing depending on the coefficient sign) at which the Chlorella, x_1, and the Brachionus, x_2, meet in the chemostat. Note that among the five SODEs displayed, only $R^{SCAD-\ell_1}$ enabled the discovery of this interaction term in both equations, which emphasizes the importance of considering the SODE coupling using MTL. The linear

terms in the two equations can be interpreted as the exponential reproduction of each "species". This exponential dynamic reproduction is limited by the species interaction and is only true for the experimental time frame.

5 Conclusion & future work

We recast the discovery of a closed-form SODE as an MTL problem. We proposed a penalty that (i) accommodates the coupling within an SODE and (ii) provides unbiased coefficients. The learning was conducted by instantiating GISTA [33]. Numerical experiments on synthetic and real datasets confirmed that harnessing both MTL and nonconvexity outperforms learning with state-of-the-art MTL-based convex regularizers.

Scientific data analysts also face the case where an experiment has been repeated under various experimental designs and they have to deal with multiple datasets. In future work, we will address the joint discovery of SODEs from multiple multivariate time series. We also plan to extend our work to partial differential equations to discover dynamics from spatiotemporal data, and thereby understand both time and spatial dynamics.

Declarations

- Funding: French National Association for Research and Technology (ANRT) and Airbus Operations (PhD grant no. 2017-1391).
- We confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.
- Ethics approval: We have read and agree with the author ethical responsibilities. The submitted work has not been published elsewhere in any form or language. We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.
- All authors consent to participate in the submitted work.
- All authors consent to the publication of the submitted work.
- Availability of data and materials: synthetic data can be reproduced by the following guidelines in the code given. Real data are publicly available and can be retrieved from their respective citation. Material: not applicable.
- Code availability: available through a dedicated Github repository if the paper is accepted.
- Authors' contributions: **Clément Lejeune**: Conceptualization, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing - original draft, Writing - review and editing. **Josiane Mothe**: Conceptualization, Funding acquisition, Investigation, Project administration, Supervision, Validation, Visualization, Writing - original

draft, Writing - review and editing. **Adil Soubki**: Conceptualization, Funding acquisition, Project administration, Supervision. **Olivier Teste**: Conceptualization, Funding acquisition, Project administration, Supervision, Writing - review and editing.

Appendix A

Proof of the results in Equation 7 We detail the steps to obtain the closed form of the proximal operator of $R_{SCAD-\ell_1}$, defined for any $\boldsymbol{W} \in \mathbb{R}^{m\times p}$ as:

$$R^{SCAD-\ell_1}_{\lambda,\theta}(\boldsymbol{W}) = \sum_{i=1}^{m} r^{SCAD}_{\lambda,\theta}\left(\|\mathbf{w}_{i\bullet}\|_1\right) = \begin{cases} \lambda\|\mathbf{w}_{i\bullet}\|_1 & \text{if } \|\mathbf{w}_{i\bullet}\|_1 \leq \lambda \\ -\frac{\lambda^2 - 2\theta\lambda\|\mathbf{w}_{i\bullet}\|_1 + \|\mathbf{w}_{i\bullet}\|_1^2}{2(\theta-1)} & \text{if } \lambda < \|\mathbf{w}_{i\bullet}\|_1 \leq \theta\lambda \\ \frac{(\theta+1)\lambda^2}{2} & \text{if } \|\mathbf{w}_{i\bullet}\|_1 > \theta\lambda \end{cases} \tag{A1}$$

We first note that $R^{SCAD-\ell_1}$ is row-separable. We compute $\text{prox}_{R^{SCAD}_{\lambda,\theta}}(\cdot)$ according to the bounds λ and $\theta\lambda$. We denote the subdifferential of a function g at $\mathbf{u} \in \mathbb{R}^p$ as $\partial g(\mathbf{u}) = \{\mathbf{y}, g(\mathbf{z}) \geq g(\mathbf{u}) + \mathbf{y}^\top(\mathbf{z} - \mathbf{x}),$ where $\mathbf{z}$ is in the domain of $g\}$. By abuse of notation, the sign function is used equivalently for vectors and scalars.
For the first bound, the optimization problem is:

$$\mathbf{z}^* \arg\min_{\mathbf{z}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 + \lambda\|\mathbf{z}\|_1$$

From the first-order optimality condition, we have the necessary conidition:

$$\mathbf{0} \in \nabla\|\mathbf{x} - \mathbf{z}\|_2^2 + \lambda\partial\|\mathbf{z}\|_1$$

Which is separable in p scalar problems. For $z_j \neq 0$, we have $\partial\|\mathbf{z}\|_1 = \operatorname{sign}\mathbf{z}$, thus we have:

$$\begin{aligned} & 0 = z_j - x_j + \lambda \operatorname{sign} z_j \\ \iff\ & z_j = x_j - \lambda \operatorname{sign} z_j \end{aligned}$$

Thus, on the first hand we have $|x_j| > \lambda$ and $\operatorname{sign} z_j = \operatorname{sign} x_j$ and we also have:

$$\begin{aligned} \|\mathbf{z}\|_1 \leq \lambda \iff & \|\mathbf{x} - \lambda \operatorname{sign}\mathbf{x}\|_1 \leq \lambda \\ \iff & \sum_j |x_j - \lambda \operatorname{sign} x_j| = |x_q - \lambda \operatorname{sign} x_q| + \sum_{j\neq q} + |x_j - \lambda \operatorname{sign} x_j| \leq \lambda \\ \iff & |x_q| - \lambda \leq |x_q - \lambda \operatorname{sign} x_q| \leq \lambda - \sum_{j\neq q} |x_j - \lambda \operatorname{sign} x_j| \\ \iff & |x_q| \leq 2\lambda - \sum_{j\neq q} |x_j - \lambda \operatorname{sign} x_j| \end{aligned}$$

Thus for $z_j \neq 0$, the q-th component of the minimizer is $z_q^* = x_q - \lambda \operatorname{sign} x_q$ if $\lambda < |x_q| \leq 2\lambda - \sum_{j\neq q}|x_j - \lambda \operatorname{sign} x_j|$. For $z_j = 0$ we have:

$$0 \in -x_j + [-\lambda, \lambda] \iff |x_j| \leq \lambda$$

And then, putting it all together, for the first bound, the q-th component of the minimizer is:

$$z_q^* = \operatorname{sign} x_q \max(|x_q| - \lambda, 0) \text{ if } |x_q| \leq 2\lambda - \sum_{j\neq q} |x_j - \lambda \operatorname{sign} x_j| \tag{A2}$$

For the second bound, we follow a very similar path and so we write it shorter. The optimization problem is:

$$\mathbf{z}^* = \arg\min_{\mathbf{z}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 - \frac{\lambda^2 - 2\theta\lambda\|\mathbf{z}\|_1 + \|\mathbf{z}\|_1^2}{2(\theta-1)}$$

Then, writting the optimality condition, we obtain for $z_j \neq 0$:

$$z_j = \frac{\theta-1}{\theta-2}x_j - \frac{\lambda\theta}{\theta-2}\operatorname{sign} z_j$$

And so we have $|z_j| > \frac{\lambda\theta}{\theta-1}$ and $\operatorname{sign} z_j = \operatorname{sign} x_j$, which, by again using the boundedness of $\mathbf{z}$, $\lambda < \|\mathbf{z}\|_1 \leq \lambda\theta$ and separating the sum, lead to:

$$z_q^* = \frac{\theta-1}{\theta-2}\operatorname{sign} x_q \max(|x_q| - \frac{\lambda\theta}{\theta-2}, 0) \text{ if } 2\lambda - A_q < |x_q| \leq \lambda\theta - A_q \tag{A3}$$

with $A_q = \sum_{j\neq q}|x_j - \lambda \operatorname{sign} x_j|$. Finally, for the third bound, the result is trivial since the second term in the optimization problem is a constant, thus the minimizer is necessarily $z_q^* = x_q$. Now by gathering (A2)(A3) and the last term, we obtain the q-th component w_q of the output of the proximal operator evaluated on, $\mathbf{w}_{i\bullet}$, the i-th row of $\boldsymbol{W}$:

$$\begin{cases} \operatorname{sign}(w_q)\max(0, |w_q| - \lambda) & \text{if } |w_q| \leq 2\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1, \\ \dfrac{\theta-1}{\theta-2}\operatorname{sign}(w_q)\max(0, |w_q| - \dfrac{\theta}{\theta-1}\lambda) & \\ \quad \text{if } 2\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1 < |w_q| \leq \theta\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1, & \\ w_q \quad \text{if } |w_q| > \theta\lambda - \|\mathbf{w}_{i\bullet,-q}\|_1, & \end{cases}$$

□

Appendix B

For reproducibility, we give the formulaes of the proximal operators associated with the four baseline regularizers used in our experiments. $\boldsymbol{W} \in \mathbb{R}^{n\times p}$, $\mathbf{w}_{i\bullet}$ denotes the i-th row (vector) of $\boldsymbol{W}$, $\lambda > 0$ and $\theta > 2$.

B.1 LASSO

$R_{\ell_{1,1}}(\boldsymbol{W}) = \sum_i \sum_j |w_{ij}|$

$$\operatorname{prox}_{\lambda R_{\ell_{1,1}}}(\boldsymbol{W})_{ij} = \operatorname{sign}(w_{ij})\max(0, |w_{ij}| - \lambda)$$

for any entry w_{ij} of $\boldsymbol{W}$ [25].

B.2 Group-LASSO

$R_{\ell_{2,1}}(\boldsymbol{W}) = \sum_i \|\mathbf{w}_{i\bullet}\|_2$

$$\operatorname{prox}_{\lambda R_{\ell_{2,1}}}(\boldsymbol{W})_{i\bullet} = \mathbf{w}_{i\bullet}\Big(1 - \frac{\lambda}{\max(\lambda, \|\mathbf{w}_{i\bullet}\|_2)}\Big)$$

for any row-vector $\mathbf{w}_{i\bullet}$ of $\boldsymbol{W}$ [11].

B.3 Sparse-Group-LASSO

$R_{\ell_{2,1}+\ell_{1,1}}(\boldsymbol{W}) = \alpha \sum_i \|\mathbf{w}_{i\bullet}\|_2 + (1-\alpha)\sum_i \sum_j |w_{ij}|$

$$\text{prox}_{\lambda R_{\ell_{2,1}+\ell_{1,1}}}(\boldsymbol{W})_{i\bullet} = \text{prox}_{(1-\alpha)\lambda R_{\ell_{1,1}}}(\text{prox}_{\alpha\lambda R_{\ell_{2,1}}}(\boldsymbol{W})_{i\bullet})_{i\bullet}$$

for $\alpha \in [0,1]$ and a row-vector $\mathbf{w}_{i\bullet}$ of $\boldsymbol{W}$. It corresponds to applying the "outer" proximal operator of $R_{\ell_{1,1}}$ entrywisely on the vector output by the "inner" proximal operator of $R_{\ell_{2,1}}$ [12].

B.4 SCAD

$R^{SCAD}_{\lambda,\theta}$

$\text{prox}_{R^{SCAD}_{\lambda,\theta}}(\boldsymbol{W})_{ij}$

$$= \begin{cases} \text{sign}(w_{ij})\max(0, |w_{ij}| - \lambda) & \text{if } |w_{ij}| \le 2\lambda \\ \text{sign}(w_{ij})\frac{\theta-1}{\theta-2}\max(0, |w_{ij}| - \frac{\lambda\theta}{\theta-2}) & \text{if } 2\lambda < |w_{ij}| \le \lambda\theta \\ w_{ij} & \text{if } |w_{ij}| > \lambda\theta \end{cases}$$

for any entry w_{ij} of $\boldsymbol{W}$ [10].

References

[1] Brunton, S.L., Noack, B.R., Koumoutsakos, P.: Machine Learning for Fluid Mechanics. Annual Review of Fluid Mechanics **52**, 477–508 (2020) https://arxiv.org/abs/1905.11075. https://doi.org/10.1146/annurev-fluid-010719-060214

[2] Li, S., Kaiser, E., Laima, S., Li, H., Brunton, S.L., Kutz, J.N.: Discovering time-varying aerodynamics of a prototype bridge by sparse identification of nonlinear dynamical systems. Physical Review E **100**(2), 22220 (2019). https://doi.org/10.1103/PhysRevE.100.022220

[3] Greiner, W.: Classical Mechanics: Point Particles and Relativity. Springer, ??? (2006)

[4] Brunton, S.L., Proctor, J.L., Kutz, J.N.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences (2016)

[5] Long, Z., Lu, Y., Ma, X., Dong, B.: PDE-Net : Learning PDEs from Data. In: International Conference on Machine Learning (2018)

[6] Schaeffer, H.: Learning partial differential equations via data discovery and sparse optimization. Proceeding of the Royal Society A **573** (2017)

[7] Schaeffer, H., McCalla, S.G.: Sparse model selection via integral terms. Physical Review E **96**(2), 023302 (2017)

[8] Zhang, L., Schaeffer, H.: On the Convergence of the SINDy Algorithm. Multiscale Modeling and Simulation **17**(3), 948–972 (2019)

[9] Ramsay, J.O., Hooker, G.: Dynamic data analysis (2017)

[10] Fan, J., Li, R.: Variable Selection via Nonconcave Penalized. Journal of the American Statistical Association **96**(456), 1348–1360 (2001)

[11] Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society. Series B: Statistical Methodology **68**(1), 49–67 (2006)

[12] Simon, N., Friedman, J., Hastie, T., Tibshirani, R.: A sparse-group lasso. Computational and Graphical Statistics, 1–13 (2013)

[13] Raissi, M., Perdikaris, P., Karniadakis, G.E.: Numerical Gaussian Processes for Time-dependent and Non-linear Partial Differential Equations. SIAM Journal of Scientific Computing **40**, 1–50 (2017) https://arxiv.org/abs/arXiv:1703.10230v1

[14] Raissi, M., Karniadakis, G.E.: Hidden physics models : Machine learning of nonlinear partial differential equations. Journal of Computational Physics **357**, 125–141 (2018)

[15] Bhat, H.S., Rawat, S.: Learning Stochastic Dynamical Systems via Bridge Sampling. In: European Conference on Machine Learning (2019)

[16] Champion, K., Lusch, B., Kutz, J.N., Brunton, S.L.: Data-driven discovery of coordinates and governing equations. PNAS **116**(45), 22445–22451 (2019)

[17] Rowley, C.W., Mezi, I., Bagheri, S., Schlatter, P., Henningson, D.S.: Spectral analysis of nonlinear flows. Journal of Fluid Mechanics **641**, 115–127 (2009). https://doi.org/10.1017/S0022112009992059

[18] Williams, M.O., Kevrekidis, I.G., Rowley, C.W.: A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. Journal of Nonlinear Science **25**(6), 1307–1346 (2015) https://arxiv.org/abs/1408.4408. https://doi.org/10.1007/s00332-015-9258-5

[19] Yeung, E., Soumya, K., Hodas, N.O.: Learning Deep Neural Network Representations for Koopman Operators of Nonlinear Dynamical Systems. In: American Control Conference, pp. 4832–4839 (2019)

[20] Kawahara, Y.: Dynamic Mode Decomposition with Reproducing Kernels for Koopman Spectral Analysis. In: NIPS, pp. 1–9 (2016)

[21] Li, J., Sun, G., Zhao, G., Lehman, L.-w.H.: Robust Low-Rank Discovery

of Data-Driven Partial Differential Equations. In: AAAI (2020). https://doi.org/10.1126/sciadv.1602614

[22] Candès, E.J., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? Journal of the ACM **58**(3) (2011). https://doi.org/10.1145/1970392.1970395

[23] Schmidt, M.D., Lipson, H.: Distilling free-form natural laws from experimental data. Science **324**, 81–85 (2009)

[24] Rudy, S., Alla, A., Brunton, S.L., Kutz, J.N.: Data-driven identification of parametric partial differential equations. SIAM Journal on Applied Dynamical Systems **18**(2), 643–660 (2019)

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

[26] Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society. Series B **67**(2), 302–320 (2005)

[27] Combettes, P.-L., Pesquet, J.-C.: Proximal splitting methods in signal processing (2011)

[28] Parikh, N., Boyd, S.: Proximal Algorithms. Foundations and Trends in Optimization (2013)

[29] Boyd, S., Vandenberghe, L.: Convex optimization (2004)

[30] Donoho, D.L., Johnstone, J.M.: Ideal spatial adaptation by wavelet shrinkage. Biometrika **81**(3), 425–455 (1994)

[31] Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. Machine Learning **73**(3), 243–272 (2008)

[32] Obozinski, G., Taskar, B., Jordan, M.I.: Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing **20**(2), 231–252 (2010)

[33] Gong, P., Zhang, C., Lu, Z., Huang, J.Z., Ye, J.: A General Iterative Shrinkage and Thresholding Algorithm for Non-convex Regularized Optimization Problems. In: International Conference on Machine Learning (2013)

[34] Gasso, G., Rakotomamonjy, A., Canu, S.: Recovering sparse signals with a certain family of non-convex penalties and DC programming. IEEE Transactions on Signal Processing **57**(12), 4686–4698 (2009)

[35] Rakotomamonjy, A., Flamary, R., Gasso, G.: DC Proximal Newton for Non-Convex Optimization Problems. IEEE Transactions on Neural Networks and Learning Systems **27**(3), 636–647 (2016)

[36] Le Thi, H.A., Phan, D.N., Pham Dinh, T.: DCA based approaches for bi-level variable selection and application for estimate multiple sparse covariance matrices. Neurocomputing **466**, 162–177 (2021). https://doi.org/10.1016/j.neucom.2021.09.039

[37] Beck, A., Teboulle, M.: A Fast Iterative Shrinkage-Thresholding Algorithm. SIAM Journal of Imaging Sciences **2**(1), 183–202 (2009)

[38] Beck, A.: First-order methods in optimization (2017)

[39] Mangan, N.M., Kutz, J.N., Brunton, S.L., Proctor, J.L.: Model selection for dynamical systems via sparse regression and information criteria. Proceeding of the Royal Society A (2017)

[40] Oliphant, T., Peterson, P., Jones, E.: Python for scientific computing. Computing in Science & Engineering **9**(90) (2001–)

[41] Cooley, J.W., Tukey, J.W.: An Algorithm for the Machine Calculation of Complex Fourier Series. Mathematics of Computation **19**(90), 297–301 (1965)

[42] Ramsay, J.O., Silverman, B.W.: Functional Data Analysis. Springer, ??? (2006)

[43] Lejeune, C., Mothe, J., Soubki, A., Teste, O.: Shape-based outlier detection in multivariate functional data. Knowledge-Based Systems **198**, 105960 (2020)

[44] Becks, L., Ellner, S.P., Jones, L.E., Hairston Jr, N.G.: Reduction of adaptive genetic diversity radically alters eco-evolutionary community dynamics. Ecology letters **13**(8), 989–997 (2010)

[45] Ramsay, J.O., Graves, S., Hooker, G.: Fda: Functional Data Analysis. (2020). R package version 5.1.9. https://CRAN.R-project.org/package=fda

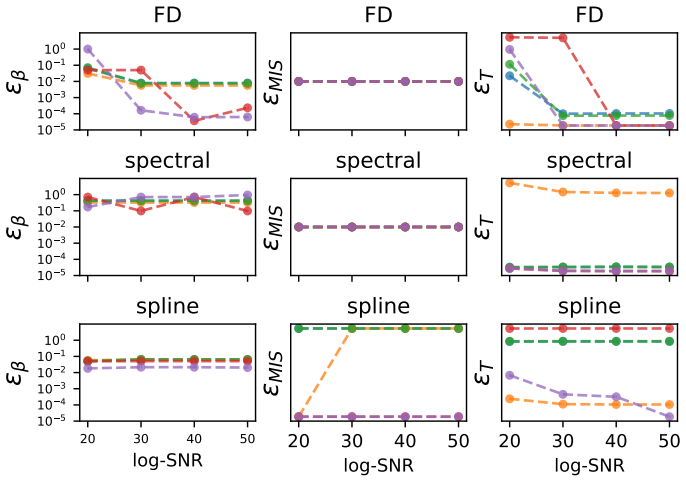
[46] Blondel, M., Pedregosa, F.: Lightning: large-scale linear classification, regression and ranking in python (2016). https://doi.org/10.5281/zenodo.200504

[47] Kaptanoglu, A.A., de Silva, B.M., Fasel, U., Kaheman, K., Goldschmidt, A.J., Callaham, J., Delahunt, C.B., Nicolaou, Z.G., Champion, K., Loiseau, J.-C., Kutz, J.N., Brunton, S.L.: Pysindy: A comprehensive

python package for robust sparse system identification. Journal of Open Source Software **7**(69), 3994 (2022)

[48] Demsar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research **7**, 1–30 (2006)

(a) DOC SODE, Gaussian noises



(b) DOC SODE, Student-t noise

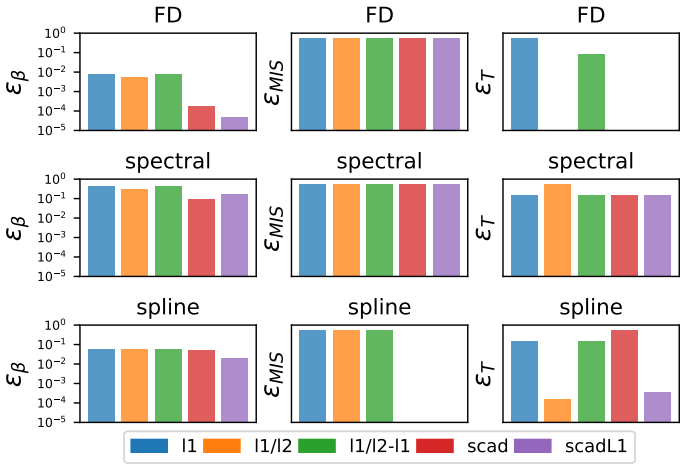
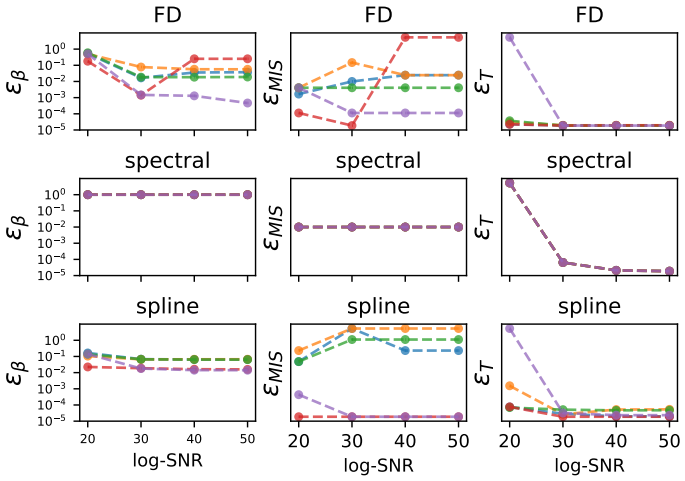


Fig. 4: Results (average over five trials in percentage, standard deviation was computed and lower than 10^{-4} hence invisible in the plots) of the learning errors of the DOC SODE with four baseline regularizers and our proposal $R^{SCAD-\ell_1}$. Each subfigure title ('FD', 'spectral' or 'spline') indicates the name of the derivative estimation method used. (a) Training data are contaminated by a Gaussian noise whose variance level is varied according to the log-SNR between 20 (high noise level) and 50 (low noise level), or by (b) a Student-t noise that results in a log-SNR ≈ 40. ϵ_β measures unbiasedness of the SODE coefficients. ϵ_{MIS} measures the misidentification error of the learned SODE. ϵ_T measures the error of the numerical solution of the learned SODE w.r.t the clean ground-truth.

(a) LV SODE, Gaussian noises



(b) LV SODE, Student-t noise

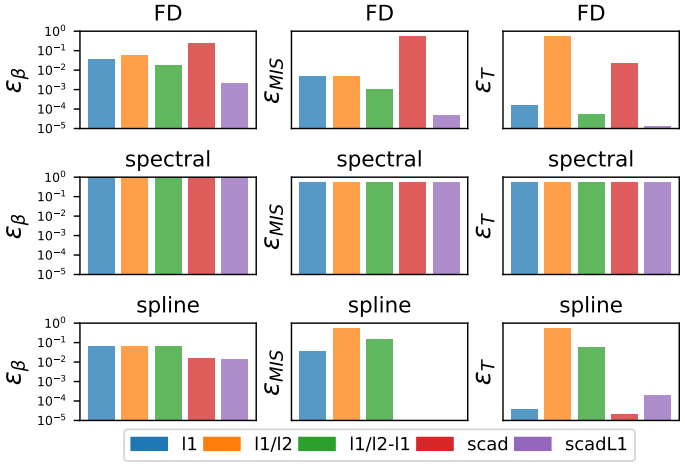
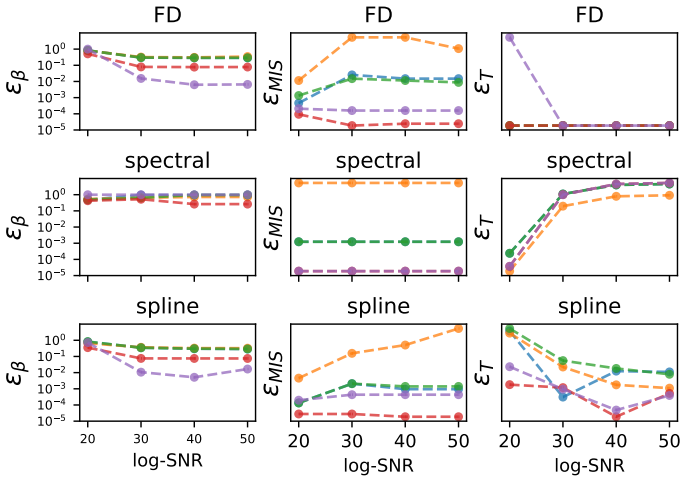


Fig. 5: Results of the learning errors of the LV SODE. Detailed comments in Fig 4.

(a) LAT SODE, Gaussian noises



(b) LAT SODE, Student-t noise

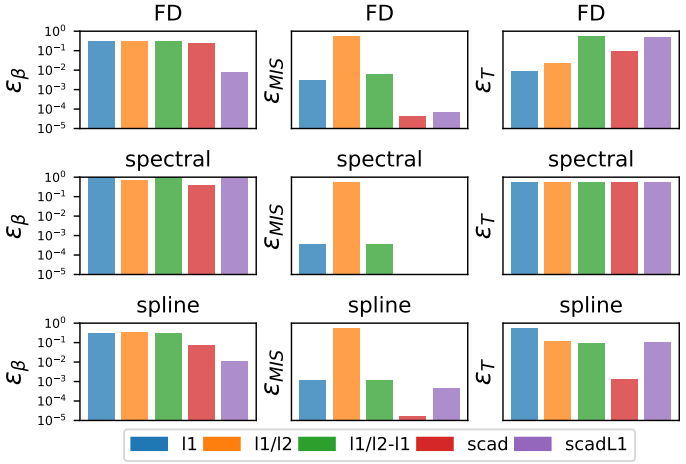


Fig. 6: Results of the learning errors of the LAT SODE with five baseline regularizers. Detailed comments in Fig 4.

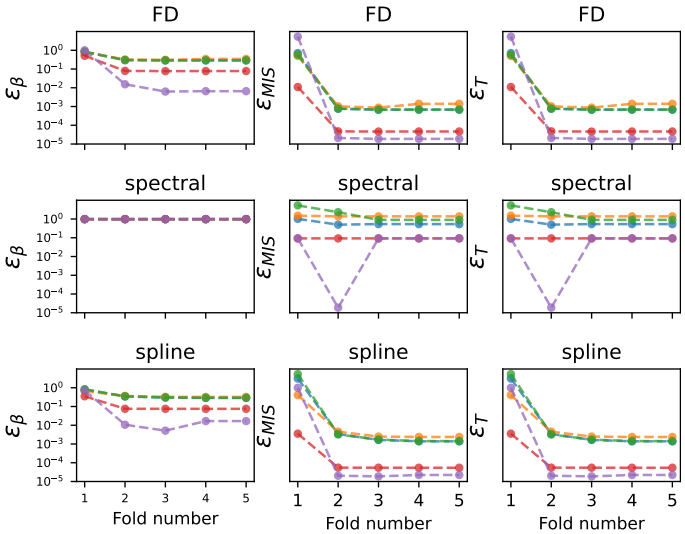


Fig. 7: Test errors of the five cross validation folds for the three synthetic SODEs. Remind that the size of the training set increases along the folds but the size of the test set is fixed. Same legend as in Fig 4.

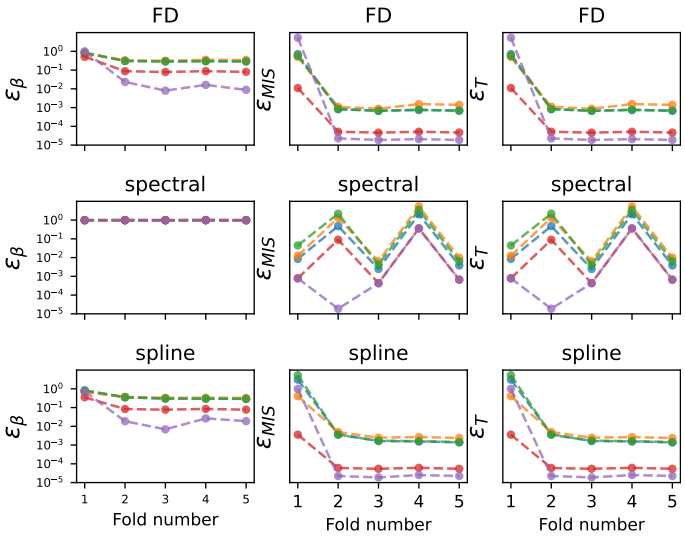


Fig. 8: Test errors of the five cross validation folds for the LV SODEs.

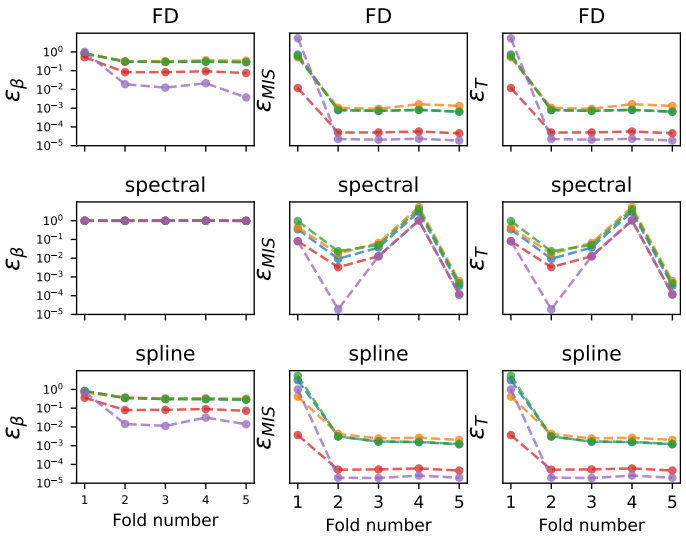
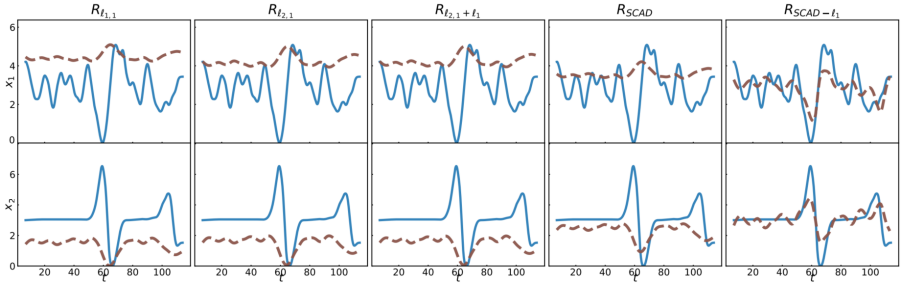


Fig. 9: Test errors of the five cross validation folds for the LAT SODEs.

(a) Time series of the two state variables. Blue: training/test data. Brown: solution of the SODEs learned with each of the four baseline regularizers and our proposal (last column).



(b) Analytic expression of the learned SODEs. For visual convenience, the intercepts and t dependence in parenthesis are omitted.

$$\begin{cases} \dot{x}_1 = -0.04x_1 \\ \quad -0.002x_2 \\ \quad -6.10^{-5}x_1^2 \\ \dot{x}_2 = 0.03x_1 \\ \quad -0.01x_2 \\ \quad 10^{-4}x_1x_2 \end{cases} \quad \begin{cases} \dot{x}_1 = -0.04x_1 \\ \quad -0.002x_2 \\ \quad -2.10^{-4}x_1^2 \\ \dot{x}_2 = 0.03x_1 \\ \quad -0.01x_2 \\ \quad +3.10^{-3}x_1x_2 \end{cases} \quad \begin{cases} \dot{x}_1 = -0.04x_1 \\ \quad -0.002x_2 \\ \quad -2.10^{-4}x_1x_2 \\ \dot{x}_2 = 0.03x_1 \\ \quad -0.01x_2 \\ \quad +3.10^{-3}x_1^2 \end{cases} \quad \begin{cases} \dot{x}_1 = -0.04x_1 \\ \quad -0.004x_2 \\ \quad -9.10^{-4}x_1x_2 \\ \dot{x}_2 = 0.06x_1 \\ \quad -0.028x_2 \\ \quad +3.10^{-4}x_1^2 \end{cases} \quad \begin{cases} \dot{x}_1 = -0.4x_1 \\ \quad -0.004x_2 \\ \quad -7.10^{-4}x_1x_2 \\ \dot{x}_2 = 0.06x_1 \\ \quad -0.028x_2 \\ \quad +4.10^{-4}x_1x_2 \end{cases}$$

Fig. 10: (a) Training/test data and numerical solution the SODEs learned with the four baseline regularizers and our proposal from the Chemostat dataset. (b) Analytic form of the learned SODEs.