Online Weak-form Sparse Identification of Partial Differential Equations

Daniel A. Messenger DANIEL.MESSENGER@COLORADO.EDU
Department of Applied Mathematics, University of Colorado, Boulder, CO 80309-0526

Emiliano Dall'anese EMILIANO.DALLANESE@COLORADO.EDU
Department of Electrical, Computer, and Energy Engineering, University of Colorado, Boulder, CO 80309-0425

David M. Bortz DAVID.BORTZ@COLORADO.EDU
Department of Applied Mathematics, University of Colorado, Boulder, CO 80309-0526

Editors: Bin Dong, Qianxiao Li, Lei Wang, Zhi-Qin John Xu

Abstract

This paper presents an online algorithm for identification of partial differential equations (PDEs) based on the weak-form sparse identification of nonlinear dynamics algorithm (WSINDy). The algorithm is online in the sense that if performs the identification task by processing solution snapshots that arrive sequentially. The core of the method combines a weak-form discretization of candidate PDEs with an online proximal gradient descent approach to the sparse regression problem. In particular, we do not regularize the ℓ_0-pseudo-norm, instead finding that directly applying its proximal operator (which corresponds to a hard thresholding) leads to efficient online system identification from noisy data. We demonstrate the success of the method on the Kuramoto-Sivashinsky equation, the nonlinear wave equation with time-varying wavespeed, and the linear wave equation, in one, two, and three spatial dimensions, respectively. In particular, our examples show that the method is capable of identifying and tracking systems with coefficients that vary abruptly in time, and offers a streaming alternative to problems in higher dimensions. Code is available at https://github.com/MathBioCU/WSINDy_PDE_OL.git.

Keywords: Online optimization, sparse regression, system identification, partial differential equations, weak form.

1. Context and Motivations

System identification (SID) and parameter estimation of dynamical systems are ubiquitous tasks in scientific research and engineering, and are required steps in many control frameworks. A typical strategy is to solve a regression problem based on sample trajectories from the underlying system, with few samples available in practice. Identification of dynamical systems is a classical field of research Ljung (1999); recently, several works provided new theoretical insights on the efficacy of classical first-order optimization methods in solving SID problems based on single trajectories (see, e.g., Fattahi et al. (2019); Foster et al. (2020); Sattar and Oymak (2020); Simchowitz et al. (2018) and references therein). Existing results in this context are heavily focused on discrete-time, finite-dimensional systems of known functional form, yet the focus on single-trajectory data paves the way for identification of more complex dynamical systems in the *online* setting, which is the subject of the current article.

By suitably discretizing candidate dynamical systems using data and employing sparse regression, SID and parameter estimation can be accomplished simultaneously. A notable development in this pursuit is the sparse identification of nonlinear dynamics (SINDy) algorithm (Brunton et al. (2016)), a general framework for discovering dynamical systems using sparse regression. Since the inception of SINDy in the context of autonomous ordinary differential equations (ODEs), sparse recovery algorithms have been

developed for autonomous partial differential equations (PDEs) (Rudy et al. (2017); Schaeffer (2017)), stochastic differential equations (SDEs) (Boninsegna et al. (2018)), non-autonomous systems (Rudy et al. (2019)), and coarse-grained equations (Bakarji and Tartakovsky (2021)), to name a few. Outside of sparse regression approaches, deep learning has also been successful in identifying PDEs from data Long et al. (2018, 2019); Wu and Xiu (2020); Qin et al. (2019).

A significant challenge in using SINDy to solve real-world problems is the computation of derivatives from noisy data. This was initially addressed in the context of ODEs in Schaeffer and McCalla (2017), by simply integrating candidate ODEs. Within the last few years, the consensus has emerged that weak-form SINDy (WSINDy, see Messenger and Bortz (2021a,b, 2022)), where integration against test functions replaces numerical differentiation, is a powerful method that is significantly more robust to noisy data, particularly in the context of PDEs. Furthermore, WSINDy's efficient convolutional formulation makes it a viable method for identifying PDEs under the constraints of limited memory capacity and computing power that exist in the online setting[1].

The development of online algorithms is a relatively recent pursuit (Zinkevich (2003); Hazan (2006)), yet much progress has been made in applications to finance (Hazan and Kale (2009)), data processing (Dixit et al. (2019)), and predictive control (Koller et al. (2018)) (see Dall'Anese et al. (2020); Hoi et al. (2021) for a recent surveys). In the context of sparse regression, several works have addressed online ℓ_1-minimization and other methods of regularizing the ℓ_0 pseudo-norm, although not in the context of learning dynamical systems (Yang et al. (2020); Zhai et al. (2019); Jialei Wang et al. (2014); Yuantao Gu et al. (2009); Kopsinis et al. (2011); Yilun Chen et al. (2009); Sun et al. (2018)). To the best of our knowledge, neither SINDy nor WSINDy have been merged with an online learning algorithm for PDEs[2].

A successful approach for identifying PDEs and tracking parameters "on the fly" using multidimensional snapshots of data arriving sequentially over time would greatly benefit many areas of science and engineering. Possible paradigms in this online setting include identifying time-varying coefficients, SID in higher dimensions (where memory constraints require data to be streamed even for offline problems), and detecting changes in the dominant balance physics of the system, as terms become active or inactive dynamically. In this way, online sparse equation discovery has the potential to open doors to new application areas, and even improve performance of existing batch methods.

We confront some of these challenges in this work by considering spatiotemporal dynamical systems and incoming data snapshots at every timestep. In the spirit of classical online algorithms, we develop an online WSINDy framework to this setting of streaming data with memory constraints by replacing full-data availability and batch optimization capabilities with data bursts and light-weight proximal gradient descent iterations to approximately solve the sparse regression problem. At each iteration we process only the incoming snapshot in time, and we do not assume the ability to compute least-squares projections apart from the initial guess. We focus on three prototypical systems, (1) the Kuramoto-Sivashinsky (KS) equation, which exhibits spatiotemporal chaos and thus has time-fluctuating Fourier content, (2) the nonlinear wave equation in a time-variable medium in two spatial dimensions, and (3) the linear wave equation in three spatial dimensions, a preliminary example of a system in higher dimensions.

1. The method developed here could also be adapted to the standard SINDy algorithm, however we choose to focus on the weak form for its demonstrated abilities to handle noisy data with low computational overhead.
2. There has, however, been work related to leveraging the equation learning ability of SINDy with Model Predictive Control (Kaiser et al. (2018)).

1.1. Notation

Vector-valued objects will be bold and lower-case, $\mathbf{x}\in\mathbb{R}^d$ for $d>1$, while multi-dimensional arrays will be bold and upper-case, $\mathbf{X}\in\mathbb{R}^{n_1\times\cdots\times n_d}$ for $n_i\in\mathbb{N}$, $1\leq i\leq d$. To disambiguate between iteration and exponentiation, we refer to the qth element in a list of multi-dimensional arrays using superscripts in parentheses (e.g. $\mathbf{x}^{(q)}$ or $\mathbf{X}^{(q)}$), whereas raising to the power q (where applicable) is simply denoted $\mathbf{X}^q$. Reference to an element within a multi-dimensional array is given as a subscript (e.g. $\mathbf{x}_i$ or $\mathbf{X}_{i_1,\ldots,i_d}$). For a matrix $\mathbf{G}\in\mathbb{C}^{m\times n}$, we denote by $\mathbf{G}_S$ the restriction of $\mathbf{G}$ to the columns in $S\subset\{1,\ldots,n\}$. By some abuse of notation, $\mathbf{G}_S^T=(\mathbf{G}_S)^T$. Similary, for a vector $\mathbf{w}\in\mathbb{C}^n$, we let $\mathbf{w}_S\in\mathbb{R}^{|S|}$ be the restriction of $\mathbf{w}$ to the entries in S, where $|S|$ denotes the number of elements of S. The complement of S within $\{1,\ldots,n\}$ is denoted S^c. All scalar-valued objects will be in lower-case, with iteration, set membership, etc. denoted by subscripts (i.e. u_q is the qth element in the list $\{u_1,\ldots,u_{q-1},u_q,u_{q+1},\ldots\}$).

2. Problem Formulation

We consider PDEs of the form

$$D^{\boldsymbol{\alpha}^{(0)}}u(\mathbf{x},t)=\sum_{i,j=1}^{I,J}\mathbf{w}^\star_{(i-1)J+j}(t)D^{\boldsymbol{\alpha}^{(i)}}f_j(u(\mathbf{x},t),\mathbf{x}),\quad(\mathbf{x},t)\in\Omega\times[0,\infty),\tag{2.1}$$

where $\Omega\subset\mathbb{R}^d$ is a bounded open set. The operators $D^{\boldsymbol{\alpha}^{(i)}}$ for $1\leq i\leq I$ represent any linear differential operator in the variables $(\mathbf{x},t)\in\mathbb{R}^{d+1}$, where $\boldsymbol{\alpha}^{(i)}=(\boldsymbol{\alpha}^{(i)}_1,\ldots,\boldsymbol{\alpha}^{(i)}_{d+1})$ is a multi-index such that

$$D^{\boldsymbol{\alpha}^{(i)}}v=\frac{\partial^{\boldsymbol{\alpha}^{(i)}_1+\cdots+\boldsymbol{\alpha}^{(i)}_d+\boldsymbol{\alpha}^{(i)}_{d+1}}}{\partial\mathbf{x}_1^{\boldsymbol{\alpha}^{(i)}_1}\cdots\partial\mathbf{x}_d^{\boldsymbol{\alpha}^{(i)}_d}\,\partial t^{\boldsymbol{\alpha}^{(i)}_{d+1}}}v.$$

In this work we consider left-hand side operators $D^{\boldsymbol{\alpha}^{(0)}}$ to be either ∂_t or ∂_{tt}, which are given in two spatial dimensions ($d=2$) by the multi-indices $\boldsymbol{\alpha}^{(0)}=(0,0,1)$ and $\boldsymbol{\alpha}^{(0)}=(0,0,2)$, respectively. The functions $f_j:\mathbb{R}\times\mathbb{R}^d\to\mathbb{R}$, $1\leq j\leq J$, include all possible nonlinearities present in the model, and together with the linear operators $D^{\boldsymbol{\alpha}^{(i)}}$ comprise the feature library $\Theta:=\{D^{\boldsymbol{\alpha}^{(i)}}f_j\}_{i,j=1}^{I,J}$. The weight vector $\mathbf{w}^\star(t)\in\mathbb{R}^{IJ}$ is assumed to be *sparse* in Θ at each time t, and is allowed to vary in t.

We assume that at each time $t=k\Delta t$ for $k\in\mathbb{N}$ and fixed timestep Δt we are given a solution snapshot $\mathbf{U}^{(t)}\in\mathbb{R}^{n_1\times\cdots\times n_d}$ of the form

$$\mathbf{U}^{(t)}=u(\mathbf{X},t)+\epsilon\tag{2.2}$$

where u solves (2.1) for some weight vector $\mathbf{w}^\star$ and $\mathbf{X}\in\mathbb{R}^{n_1\times\cdots\times n_d}$ is a fixed known spatial grid of points in Ω having n_i points in the ith dimension and equal spacing Δx in each dimension. Here ϵ represents i.i.d. mean-zero noise with fixed finite variance σ^2 associated with sampling the underlying solution $u(\mathbf{x},t)$ at any point $\mathbf{x}\in\Omega$. We write $\mathbf{U}=(\mathbf{U}^{(0)},\mathbf{U}^{(\Delta t)},\ldots,\mathbf{U}^{(k\Delta t)},\ldots)$ to denote the entire dataset in time. The problem is stated as follows.

Problem: Assume that a total of K_{mem} snapshots $\{\mathbf{U}^{(t-(K_{\text{mem}}-1)\Delta t)},\ldots,\mathbf{U}^{(t)}\}$ can be stored in memory at each time t and that at time $t+\Delta t$ a new snapshot $\mathbf{U}^{(t+\Delta t)}$ arrives, replacing the oldest snapshot in memory. Given the sampling model (2.2) for unknown σ^2, unknown ground truth PDE (2.1), and fixed library $\Theta:=\{D^{\boldsymbol{\alpha}^{(i)}}f_j\}_{i,j=1}^{I,J}$, solve for coefficients $\widehat{\mathbf{w}}^{(t)}$ such that $\sup_{t>0}\left\|\widehat{\mathbf{w}}^{(t)}-\mathbf{w}^\star(t)\right\|$ is bounded.

3. Batch WSINDy

In the batch setting, assuming $\mathbf{w}^\star$ is constant in time, the weak-form sparse identification of nonlinear dynamics algorithm (WSINDy) proposed in Messenger and Bortz (2021a,b) solves this problem efficiently by first convolving equation (2.1) with a smooth function $\psi(\mathbf{x},t)$, compactly supported in $\Omega\times[0,T]$. After integrating by parts to put all partial derivatives onto ψ, this leads to the *convolutional weak form*:

$$D^{\boldsymbol{\alpha}^{(0)}}\psi * u(\mathbf{x},t) = \sum_{i,j=1}^{I,J} \mathbf{w}^\star_{(i-1)J+j} D^{\boldsymbol{\alpha}^{(i)}}\psi * f_j(u,\cdot)(\mathbf{x},t), \tag{3.1}$$

where convolutions are performed over space and time. For efficiency, the test function ψ is chosen to be separable,

$$\psi(\mathbf{x},t) = \phi_1(\mathbf{x}_1)\cdots\phi_d(\mathbf{x}_d)\phi_{d+1}(t). \tag{3.2}$$

For example, it can be chosen using the Fourier spectrum of the noisy data to mitigate high-frequency noise (see Messenger and Bortz (2021a)). Once ψ is chosen, we discretize the problem by selecting a finite set of *query points* $\mathcal{Q} := \{(\mathbf{x}^{(q)},t_q)\}_{q=1}^{Q} \subset \Omega\times(0,T)$ and evaluating (3.1) at $\mathcal{Q}$, replacing u with the full dataset $\mathbf{U}$. Convolutions can be efficiently computed using the fast Fourier transform (FFT), which, due to the compact support of ψ, is equivalent to the trapezoidal rule and is highly accurate in the noise-free case ($\sigma^2 = 0$). This gives us the linear system

$$\mathbf{b} \approx \mathbf{G}\mathbf{w}^\star,$$

where the qth entry of $\mathbf{b}$ is $\mathbf{b}_q = D^{\boldsymbol{\alpha}^{(0)}}\psi * \mathbf{U}(\mathbf{x}^{(q)},t_q)$ and qth entry of the $((i-1)J+j)$th column of $\mathbf{G}$ is $\mathbf{G}_{q,(i-1)J+j} = D^{\boldsymbol{\alpha}^{(i)}}\psi * f_j(\mathbf{U},\cdot)(\mathbf{x}^{(q)},t_q)$. Using the assumption that $\mathbf{w}^\star$ is sparse, we solve this linear system for $\widehat{\mathbf{w}} \approx \mathbf{w}^\star$ by solving the sparse recovery problem

$$\min_{\mathbf{w}\in\mathbb{R}^{IJ}} F(\mathbf{w};\lambda) = \min_{\mathbf{w}\in\mathbb{R}^{IJ}} \frac{1}{2}\|\mathbf{G}\mathbf{w}-\mathbf{b}\|_2^2 + \frac{1}{2}\lambda^2\|\mathbf{w}\|_0. \tag{3.3}$$

The sparsity threshold $\lambda > 0$ must be set by the user and is designed to strike a balance between fitting the data, associated with low residual $\|\mathbf{G}\mathbf{w}-\mathbf{b}\|_2$, and finding a parsimonious model, indicated by low $\|\mathbf{w}\|_0$ (and its value is typically calibrated via cross-validation) Hastie et al. (2009); Foucart and Rauhut (2013).

With a large enough library Θ, a sparse vector $\widehat{\mathbf{w}}$ is required in order to interpret and efficiently simulate the resulting PDE. Replacing the ℓ_0-pseudonorm with e.g. an ℓ_2 penalty (i.e. ridge regression) may shrink coefficients, but will not result in a sparse $\widehat{\mathbf{w}}$. In addition, the columns of $\mathbf{G}$ are typically highly correlated since they are each constructed from the same dataset $\mathbf{U}$, which leads to many popular algorithms for solving (3.3) performing poorly, such as convex relaxation using the ℓ_1-norm Meinshausen and Bühlmann (2006); Fan and Liao (2014). In the batch setting, the following approach has proved to be successful under various noise levels and systems of interest. For $\lambda > 0$ define the inner sequential thresholding step

$$\text{MSTLS}(\mathbf{G},\mathbf{b};\lambda) \qquad \begin{cases} \mathbf{w}^{(0)} = \mathbf{G}^\dagger\mathbf{b} \\ \mathcal{I}^{(\ell)} = \{1\le k\le IJ \,:\, L_k(\lambda)\le|\mathbf{w}_k^{(\ell)}|\le U_k(\lambda)\} \\ \mathbf{w}^{(\ell+1)} = \underset{\text{supp}(\mathbf{w})\subset\mathcal{I}^{(\ell)}}{\text{argmin}} \|\mathbf{G}\mathbf{w}-\mathbf{b}\|_2^2. \end{cases} \tag{3.4}$$

Letting $\mathbf{G}_k$ be the kth column of $\mathbf{G}$, the lower and upper bounds are defined

$$\begin{cases} L_k(\lambda) = \lambda \max\left\{1, \dfrac{\|\mathbf{b}\|}{\|\mathbf{G}_k\|}\right\} \\ U_k(\lambda) = \dfrac{1}{\lambda}\min\left\{1, \dfrac{\|\mathbf{b}\|}{\|\mathbf{G}_k\|}\right\} \end{cases}, \qquad 1 \leq k \leq IJ. \tag{3.5}$$

The sparsity threshold $\widehat{\lambda}$ is then selected as the smallest minimizer of the cost function

$$\mathcal{L}(\lambda) = \frac{\|\mathbf{G}(\mathbf{w}(\lambda) - \mathbf{w}(0))\|_2}{\|\mathbf{G}\mathbf{w}(0)\|_2} + \frac{\|\mathbf{w}(\lambda)\|_0}{IJ} \tag{3.6}$$

where $\mathbf{w}(\lambda) := \text{MSTLS}(\mathbf{G},\mathbf{b};\lambda)$. We find $\widehat{\lambda}$ via grid search and set $\widehat{\mathbf{w}} = \text{MSTLS}(\mathbf{G},\mathbf{b};\widehat{\lambda})$ as the output of the algorithm. In words, this is a modified sequential thresholding algorithm with non-uniform thresholds (3.5) chosen based on the norms of the underlying library terms $\mathbf{G}_{(i-1)J+j} \approx D^{\boldsymbol{\alpha}^{(i)}}\psi * f_j(u)$ relative to the response vector $\mathbf{b} \approx D^{\boldsymbol{\alpha}^{(0)}}\psi * u$. The purpose of this is to (a) incorporate relative sizes of library terms $\mathbf{G}_k\mathbf{w}_k^\star$ along with absolute sizes of coefficients $\mathbf{w}^\star$ in the thresholding step, and (b) choose λ automatically.

4. Online WSINDy

The online setting is defined by data snapshots $\mathbf{U}^{(t)}$ arriving sequentially over time. An estimate $\widehat{\mathbf{w}}^{(t)}$ of the true parameters $\mathbf{w}^\star(t)$ must be computed before the arrival of the next snapshot $\mathbf{U}^{(t+\Delta t)}$ using only a fixed number K_{mem} of previous snapshots. Without access to the full time series $\mathbf{U}$, combined effects of the sample rate Δt, the number of snapshots K_{mem}, and the intrinsic timescales of the data determine the identifiability of the system: Δt must be small enough to accurately compute time integrals, but large enough that the data $\mathbf{U}$ is sufficiently dynamic over the time window $K_{\text{mem}}\Delta t$. Corruptions from noise have a greater impact because variance is not reduced by considering many samples in time, as was the case in the batch setting. Moreover, in realistic settings, solving for $\widehat{\mathbf{w}}^{(t)}$ before arrival of the next snapshot $\mathbf{U}^{(t+\Delta t)}$ fundamentally limits the size of $(\mathbf{G},\mathbf{b})$ and the number of iterations one may perform using any sparse solver.

The online setting is inherently restrictive, yet it appears well-suited for an important set of problems that are challenging offline and for settings where $\widehat{\mathbf{w}}^{(t)}$ must be obtained without revisiting past data. In the batch setting, when the coefficient vector $\mathbf{w}^\star$ varies over time, the library Θ must include time-dependent terms and may grow too large to successfully solve for an accurate sparse solution. Another issue arises with high-dimensional datasets (as in cosmology, turbulence, molecular dynamics, etc.), which cannot easily be processed in a single batch. In these cases an online approach is natural and advantageous even if solutions $\widehat{\mathbf{w}}^{(t)}$ are not themselves required "online".

For the online approach, at each time t we seek to minimize the online cost function

$$\min_{\mathbf{w}\in\mathbb{R}^{IJ}} F_t(\mathbf{w};\lambda_t) = \min_{\mathbf{w}\in\mathbb{R}^{IJ}} \frac{1}{2}\left\|\mathbf{G}^{(t)}\mathbf{w} - \mathbf{b}^{(t)}\right\|_2^2 + \frac{1}{2}\lambda_t^2\|\mathbf{w}\|_0, \tag{4.1}$$

where $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$ is the linear system created from the K_{mem} slices $\{\mathbf{U}^{(t-(K_{\text{mem}}-1)\Delta t)},\ldots,\mathbf{U}^{(t)}\}$ at time t. Notice also that we allow λ_t to change, as the initial guess λ_0 may not be optimal. In this online setting, we assume that we do not have the luxury of computing least-squares solutions (other than the initial guess), so we cannot use the approach outlined in (3.4)-(3.6), where (3.4) requires multiple least-squares solves, and performing a grid search over λ values requires multiple solves of (3.4). Hence, we consider

the following online algorithm, which is simply the online proximal gradient descent combined with a decision tree update for λ_t at each step:

$$\begin{cases} \mathbf{z}^{(t)} = \widehat{\mathbf{w}}^{(t)} - \alpha_t (\mathbf{G}^{(t)})^T \left(\mathbf{G}^{(t)} \widehat{\mathbf{w}}^{(t)} - \mathbf{b}^{(t)} \right) \\ \widehat{\mathbf{w}}^{(t+\Delta t)} = H_{\lambda_t} \left(\mathbf{z}^{(t)} \right) \\ \lambda_{t+\Delta t} = \mathcal{T}(\lambda_t, \widehat{\mathbf{w}}^{(t+\Delta t)}, \Delta\lambda, \lambda_{\max}). \end{cases} \tag{4.2}$$

The hard thresholding operator $H_\lambda(\mathbf{w})$ is the proximal operator of $\frac{1}{2}\lambda^2 \|\mathbf{w}\|_0$ and is defined as

$$(H_\lambda(\mathbf{w}))_k = \begin{cases} \mathbf{w}_k, & |\mathbf{w}_k| \geq \lambda \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

The map $\mathcal{T}$ updates λ_t according to

$$\mathcal{T}(\lambda_t, \widehat{\mathbf{w}}^{(t+\Delta t)}; \Delta\lambda, \lambda_{\max}) = \\ \begin{cases} (1-\Delta\lambda)\lambda_t, & F_t(\widehat{\mathbf{w}}^{(t+\Delta t)}, \lambda_t) > F_{t-\Delta t}(\widehat{\mathbf{w}}^{(t)}, \lambda_t) \ \& \ S_{t+\Delta t} \subsetneq S_t \\ (1-\Delta\lambda)\lambda_t + \lambda_{\max}\Delta\lambda, & \begin{cases} F_t(\widehat{\mathbf{w}}^{(t+\Delta t)}, \lambda_t) > F_{t-\Delta t}(\widehat{\mathbf{w}}^{(t)}, \lambda_t) \ \& \ S_t \subsetneq S_{t+\Delta t} \\ F_t(\widehat{\mathbf{w}}^{(t+\Delta t)}, \lambda_t) \leq F_{t-\Delta t}(\widehat{\mathbf{w}}^{(t)}, \lambda_t) \ \& \ S_t = S_{t+\Delta t}. \end{cases} \\ \lambda_t, & \text{otherwise.} \end{cases} \tag{4.4}$$

In words, there are two possible updates to λ_t: a convex combination between λ_t and 0 and a convex combination between λ_t and $\lambda_{\max}$. The former decreases λ_t and occurs when library terms are thresholded to zero *and* the objective function F_t increases. The latter increase λ_t and occurs when either (a) library terms are added *and* F_t increases or (b) the support set $S_t := \text{supp}\left(\widehat{\mathbf{w}}^{(t)}\right)$ doesn't change *and* F_t does not increase[3]. At each step we set $\alpha_t = 1/\|(\mathbf{G}^{(t)})^T \mathbf{G}^{(t)}_{S_t}\|_2$, the optimal stepsize for pure gradient descent given the support S_t. As an initial guess we set $\widehat{\mathbf{w}}^{(0)} = \left(\mathbf{G}^{(0)}\right)^\dagger \mathbf{b}^{(0)}$, which is the only least squares solve performed.

Remark 4.1. It is well-known in the batch case that picking λ is problem specific and prone to errors particularly in the presence of noise (see Messenger and Bortz (2021a) for a discussion). Commonly some form of cross-validation is used to select λ offline. This is carried out in Maddu et al. (2022) for offline PDE identification using several sparse regression algorithms including proximal gradient descent applied to (3.3). It is less common to update λ over the course of the algorithm, although several strategies for this are presented in Donoho et al. (2012). We stress that for variable-coefficient PDEs, as considered here, a time-varying λ is *necessary*, and offline cross validation can at best provide a good initial guess. The update policy given by $\mathcal{T}$ encodes simple objectives of any algorithm for (4.1) and works in all examples presented, however we leave optimizing the update rule as a topic for future work.

Remark 4.2. Similar to the batch case, we find that non-uniform thresholding greatly improves results. For brevity, we include in Appendix 7.1 a description of how non-uniform thresholds such as (3.5) are incorporated into the online framework. We also note that the theoretical results in the next section carry over analogously in the non-uniform thresholding case.

3. The value for λ_t (and similarly for α_t) can easily be replaced by a constant when additional knowledge is available (e.g. when $\mathbf{w}^\star$ is known to satisfy certain bounds).

4.1. Regret and Fixed Point Analysis

The behavior of the online algorithm is in large part dictated by the behavior of the batch proximal gradient descent method. The proximal gradient descent algorithm applied to the ℓ_0 norm is referred to as iterative hard thresholding (IHT) and was first studied rigorously in Blumensath and Davies (2008). The lemmas below review some useful properties that can be found in that work relating solutions of (3.3) and stationary points of the proximal gradient descent algorithm (4.2) in the offline case and for fixed λ. We then use these results to bound the dynamic online regret, which we define as

$$Reg_D(T) := \sum_{\substack{k=0 \\ t=k\Delta t}}^{T} F_t(\widehat{\mathbf{w}}^{(t)};\lambda_t) - F_t(\mathbf{w}^\star(t);\lambda_t), \tag{4.5}$$

where $\mathbf{w}^\star(t)$ is a global minimizer of $F_t(\mathbf{w},\lambda_t)$. In particular, we first have the following:

Lemma 4.1. *Consider* $\mathbf{w}$ *such that one of the following holds:*

(i) $\mathbf{w}$ *is a local minimizer of* (3.3)

(ii) $\mathbf{w} = H_\lambda\big(\mathbf{w} - \mathbf{G}^T(\mathbf{G}\mathbf{w}-\mathbf{b})\big)$

(iii) With $S = supp(\mathbf{w})$*, we have that* $\mathbf{w}_S \in \operatorname{argmin}_{\mathbf{z}} \|\mathbf{G}_S\mathbf{z}-\mathbf{b}\|_2^2$ *and*

$$\max_{i\in S^c} \left|\mathbf{G}_i^T(\mathbf{G}\mathbf{w}-\mathbf{b})\right| < \lambda \leq \min_{i\in S} |\mathbf{w}_i|.$$

Then it holds that $(ii) \iff (iii) \implies (i)$*. Moreover, if* $\mathbf{w}$ *a global minimizer, then* $(i) \implies (iii)$*.*

For completeness, a proof of Lemma 4.1 can be found in Appendix 7.3. For convergence of the algorithm, we also have the following from Blumensath and Davies (2008).

Lemma 4.2. *Assume that* $\|\mathbf{G}\|_2 < 1$*. Then the iterates* $\mathbf{w}^{(n+1)} = H_\lambda(\mathbf{w}^{(n)} - \mathbf{G}^T(\mathbf{G}\mathbf{w}^{(n)} - \mathbf{b}))$ *converge to a fixed point of* (3.3)*.*

Lemma 4.1 implies that fixed points of the batch proximal gradient descent algorithm are local minimizers of $F(\mathbf{w};\lambda)$, and moreover that fixed points satisfy a necessary condition for global optimality given by (iii). Lemma 4.2 then guarantees[4] that iterates $\mathbf{w}^{(n)}$ do indeed converge to a local minimizer $\widehat{\mathbf{w}}$, and further that $\text{supp}\big(\mathbf{w}^{(n)}\big) = \text{supp}(\widehat{\mathbf{w}})$ for all $n \geq N$, for some finite N. However, we are not aware of results that guarantee recovery of the true support $\text{supp}(\mathbf{w}^\star)$, where it is assumed that $\mathbf{b} = \mathbf{G}\mathbf{w}^\star + \mathbf{e}$ for noise $\mathbf{e}$. In Blumensath and Davies (2009), support recovery is proved for a related algorithm where $H_\lambda(\mathbf{w})$ is replaced by $H_s(\mathbf{w})$, which selects the largest s elements of $\mathbf{w}$, but this relies on several assumptions including a restricted isometry property, small noise $\mathbf{e}$, and knowledge of the sparsity level s. In the current setting of PDE identification from noisy data, none of these assumptions are realistic, although a similar support recovery result for algorithm (4.2) in the batch case would fill a gap in the literature.

If a fixed point $\widehat{\mathbf{w}}$ with $\text{supp}(\widehat{\mathbf{w}}) = S$ satisfies that $\mathbf{G}_S^T\mathbf{G}_S$ is full rank, then $\widehat{\mathbf{w}}_S = \mathbf{G}_S^\dagger\mathbf{b}$ is the unique least squares solution over the columns in S. In Nikolova (2013) it is shown that this is sufficient for $\widehat{\mathbf{w}}$ to be a *strict* local minimizer, and moreover the only local minimizer with support S. Also in Nikolova (2013) is an extensive treatment of global minimizers of $F(\mathbf{w};\lambda)$, where it is shown that apart from a measure-zero set of linear systems $(\mathbf{G},\mathbf{b})$, the global minimizer is unique. We use this to bound the dynamic regret below.

4. The condition $\|\mathbf{G}\|_2 < 1$ in Lemma 4.2 can be replaced by stepsize $\alpha > 1$ satisfying $\alpha < 1/\|\mathbf{G}\|_2^2$.

Theorem 4.1. *Let $\sigma_{1,t}$ and $\sigma_{n,t}$ denote the first and last singular values of the matrix $\mathbf{G}^{(t)} \in \mathbb{R}^{m\times n}$. Assume the following:* $\max_t \lambda_t \leq \overline{\lambda} < \infty$, $\min_t \sigma_{n,t} \geq \overline{\sigma}_{\min} > 0$, $\max_t \sigma_{1,t} \leq \overline{\sigma}_{\max}$, *and* $\sup_t \alpha_t < \sigma_{\max}^{-2}$, $\inf_t \alpha_t > 0$. *In addition, assume that the global minimizer $\mathbf{w}^\star(t)$ of $F_t(\mathbf{w};\lambda_t)$ is unique for every t and satisfies $|S_t^\star| \geq \overline{s} > 0$ where $S^\star = supp(\mathbf{w}^\star(t))$. Finally, assume that the tracking gap is globally bounded: $\|\mathbf{w}^\star(t) - \mathbf{w}^\star(t+\Delta t)\|_2 := d_t \leq \overline{d}$. Then the dynamic regret* (4.5) *grows at-worst linearly:*

$$Reg_D(T) \leq C_1 + C_2 T$$

for some $C_1 > 0$ and $C_2 > 0$. In particular, $\frac{1}{T} Reg_D(T)$ remains bounded.

The constants C_1 and C_2 are specified in the proof, which is presented in Appendix 7.4.

Remark 4.3. The above result establishes that $Reg_D(T)$ increases at-worst linearly in T, but this is only qualitative (the constants C_1 and C_2 are not meant to be sharp). Asymptotically, this is the same rate as online gradient descent applied to the time-varying ordinary least squares problem (Zinkevich (2003)), and is a well-known fundamental limit for cases where the tracking gap d_t does not go to zero (see e.g. Besbes et al. (2015)).

Remark 4.4. Lines (7.5)-(7.6) of the proof establish error bounds on the coefficients, which lead to the asymptotic bound

$$\limsup_{t\to\infty} \left\| \widehat{\mathbf{w}}^{(t)} - \mathbf{w}^\star(t) \right\|_2 \leq \frac{1}{1-\overline{\rho}} \limsup_{t\to\infty} \left(d_t + \alpha_t \lambda_t \sqrt{|S_{t+\Delta t} \triangle S_t^\star|} \right),$$

where $\overline{\rho} := \sup_{t\geq 0} \left\| \mathbf{I} - \alpha_t (\mathbf{G}^{(t)})^T \mathbf{G}^{(t)} \right\|_2$ and $S_{t+\Delta t} \triangle S_t^\star$ is the set difference between $S_{t+\Delta t} = \text{supp}\left(\widehat{\mathbf{w}}^{(t+\Delta t)}\right)$ and $S_t^\star := \text{supp}(\mathbf{w}^\star(t))$. This implies that if the tracking error and support difference go to zero ($d_t \to 0$, $S_{t+\Delta t} \triangle S_t^\star \to \emptyset$) then we recover the true coefficients in the limit.

Remark 4.5. The assumptions of Theorem 4.1 are standard for overdetermined $\mathbf{G}^{(t)}$ and data that is not pathological. In particular, upper bounds on $\sigma_{1,t}$ and d_t merely imply that the data does not blow up, while lower bounds on $\sigma_{n,t}$ and $|S_t^\star|$ imply that the data does not reach an equilibrium state. While both of these cases, blow up and equilibration, are interesting, the former rarely occurs in practice, and the latter is sufficiently challenging as to require new developments in a future work. Upper bounds on λ_t and α_t are cosmetic and required for the algorithm to produce nonzero solutions that are bounded. A lower bound on α_t is crucial to ensure $\overline{\rho} < 1$, which is necessary for convergence once the correct support has been recovered. We leave the case of underdetermined $\mathbf{G}^{(t)}$ to future work, but note that in practice the algorithm generally reaches an overdetermined subset after finitely many iterations.

Below we only examine cases where $S_t^\star = S^\star$ is fixed, and find that over a wide range of parameters the correct support is found in finitely many iterations. This leads to scenarios where the dynamic regret depends only on d_t asymptotically (see Figure 3 for a visualization of this case for the time-varying wave equation). We leave online discovery of PDEs with time-varying support to future work.

5. Numerical Experiments

Our primary focuses are the performance of the algorithm as a function of the number of snapshots K_{mem} allowed in memory and the sensitivity of the algorithm to noise. We examine the following three examples which display a range of dynamics over one to three spatial dimensions: the Kuramoto-Sivashinsky equation

in 1D, a time-varying nonlinear wave equation in 2D, and the linear wave equation in 3D. We abbreviate each by KS, W2D, and W3D. For each experiment we simulate a noise-free solution $\mathbf{U}_{exact}$ to the given PDE over a long time horizon. We then add i.i.d. Gaussian noise with mean zero and standard deviation $\sigma = \sigma_{NR}\|\mathbf{U}^\star\|_{rms}$ to each data point for a range of *noise ratios*[5] σ_{NR}. After an offline phase where a least squares solution is found from the first K_{mem} snapshots, we feed in one new snapshot at each time t and apply the online algorithm (4.2). Code is available at https://github.com/MathBioCU/WSINDy_PDE_OL.git including the KS dataset, with W2D and W3D datasets available on request.

ALGORITHM HYPERPARAMETERS

We fix as many hyperparameters across examples as possible, and differences are summarized in Table 1. In all examples we fix the sparsity threshold update to $\Delta\lambda = 0.1$, the initial sparsity threshold to $\lambda_0 = 0.0001$, and the maximum sparsity threshold to $\lambda_{\max} = 0.1$. For the library we use

$$\Theta = \{\partial^k_{\mathbf{x}_i}(u^j)\}, \qquad 1 \le i \le d, 0 \le k \le 4, 0 \le j \le 4$$

in other words all spatial derivatives up to degree 4 of monomials up to degree 4 of the data (excluding mixed derivatives). For direct comparison of the effects of K_{mem} and σ_{NR} across examples, we fix the test function ψ in the representation 3.2 so that[6]

$$\phi_i(\mathbf{x}_i) = \left(1 - \left(\frac{\mathbf{x}_i}{21\Delta x}\right)^2\right)_+^{11}, \qquad 1 \le i \le d \tag{5.1}$$

$$\tag{5.2}$$

and

$$\phi_{d+1}(t) = \left(1 - \left(\frac{t}{(K_{\text{mem}}-1)\Delta t/2}\right)^2\right)_+^{9}, \tag{5.3}$$

where $(z)_+ := \max\{z, 0\}$. In this way ψ is supported on $2\times 21+1 = 43$ points in each spatial dimension and K_{mem} points in time, although note that $(\Delta x, \Delta t)$ change across examples. Since $\phi_{d+1}(t)$ is supported on K_{mem} points, there is only one integration in time at each iteration, so that the query points are given by $\mathcal{Q} = \{(\mathbf{x}^{(q)}, t_q)\}_{q=1}^{Q} = \mathcal{Q}_{\mathbf{x}} \times \{t - (K_{\text{mem}}-1)\Delta t/2\}$ where for each example $\mathcal{Q}_{\mathbf{x}}$ is fixed across all values of K_{mem} and σ_{NR}. We take $\mathcal{Q}_{\mathbf{x}} \subset \mathbf{X}$ to be equally-spaced and such that the linear system $(\mathbf{G}^{(t)}, \mathbf{b}^{(t)})$ contains less than 10,000 rows (see Table 1 for exact dimensions). Online iteration times are reported below for computations performed on a laptop with 1.7GHz base clockspeed AMD Ryzen 7 pro 4750u processor and 38.4 GB of RAM.

Remark 5.1. By defining the temporal test function $\phi_{d+1}(t)$ to depend on K_{mem} according to (5.3), the implied strategy is that increasing K_{mem} (keeping more snapshots in memory) leads to more accurate

5. Note that σ_{NR} is approximately equal to the ratio $\|\epsilon(:)\|_2 / \|\mathbf{U}_{exact}(:)\|_2$ of the noise to the true data, where "$\mathbf{U}_{exact}(:)$" denotes $\mathbf{U}_{exact}$ stretched into a column vector.
6. Test functions (5.2) and (5.3) can be made general by replacing powers $p_x = 11$, $p_t = 9$ and spacings $m_x = 21$, $m_t = (K_{\text{mem}}-1)/2$ with general values p_x, p_t and m_x, m_t as in Messenger and Bortz (2021a, 2022). The resulting general form for ψ has been shown to be successful across a wide range of systems. However, optimal test function selection is an active area of research.

	dims($\mathbf{X}$)	T	dims($\mathbf{G}^{(t)}$)	$(\Delta x,\Delta t)$
KS	256×1	3946	214×21	(0.939,0.586)
W2D	129×403	1639	7964×37	(0.0156,0.0122)
W3D	$128\times 128\times 128$	960	8192×53	(0.0491,0.0122)

Table 1: Resolution and dimensions of datasets used in examples.

integration in the time domain. One could instead fix the test function

$$\phi_{d+1}(t)=\left(1-\left(\frac{t}{m\Delta t}\right)^2\right)_+^9$$

for some $m\leq (K_{\text{mem}}-1)/2$ for all K_{mem} considered, leading to a fixed integration window of length $2m+1$ in time. Increasing K_{mem} would then allow for more integrations in time (i.e. a larger set of query points $\mathcal{Q}$), adding rows to the linear system $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$. Our chosen strategy fixes the dimensions of $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$, leading to a more direct comparison across examples. We leave this trade-off between the number of time integrations and the accuracy of time integrations to future work.

PERFORMANCE ANALYSIS

We are concerned with the ability of the algorithm to recover the support of the true model coefficients $S^\star:=\text{supp}(\mathbf{w}^\star)$ as well as the accuracy of $\widehat{\mathbf{w}}^{(t)}$ over time, depending primarily on the number K_{mem} of solution snapshots allowed in memory and the noise level σ_{NR} corrupting the data. To assess support recovery, we measure the *true positivity ratio* (TPR)

$$\text{TPR}(\widehat{\mathbf{w}}^{(t)}):=\frac{\text{TP}(\widehat{\mathbf{w}}^{(t)})}{\text{TP}(\widehat{\mathbf{w}}^{(t)})+\text{FP}(\widehat{\mathbf{w}}^{(t)})+\text{FN}(\widehat{\mathbf{w}}^{(t)})}$$

where $\text{TP}(\widehat{\mathbf{w}}^{(t)}):=|S_t\cap S^\star|$ is the number of correctly identified nonzero coefficients, $\text{FP}(\widehat{\mathbf{w}}^{(t)}):=|S_t\cap(S^\star)^c|$ is the number of falsely identified nonzero coefficients, and $\text{FN}(\widehat{\mathbf{w}}^{(t)}):=|S_t^c\cap S^\star|$ is the number of falsely identified zero coefficients. A TPR of 1 indicates successful support recovery, while $\text{TPR}=0.75$ indicates 3/4 terms were correctly identified, and so on. We measure the accuracy of $\widehat{\mathbf{w}}^{(t)}$ in the relative ℓ_2-norm:

$$E_2(\widehat{\mathbf{w}}^{(t)}):=\left\|\widehat{\mathbf{w}}^{(t)}-\mathbf{w}^\star(t)\right\|_2/\|\mathbf{w}^\star(t)\|_2.$$

We report the results of $\text{TPR}(\widehat{\mathbf{w}}^{(t)})$ and $E_2(\widehat{\mathbf{w}}^{(t)})$ averaged over 100 instantiations of noise.

5.1. Kuramoto-Sivashinsky (KS)

$$\partial_t u=-\partial_x\left(u^2\right)-\partial_{xx}u-\partial_{xxxx}u. \tag{5.4}$$

The Kuramoto-Sivashinsky (KS) equation is challenging because the solution exhibits spatiotemporal chaos and so has a Fourier spectrum that varies in time. This leads to potentially different dynamics at each timestep in the online learning perspective. The PDE also has a 4th-order derivative in space which is difficult to compute accurately and to identify via sparse regression, especially when noise is present. We

simulate the solution using a high-order method (accurate to 6-7 digits) and use a dataset of 256×3496 points in space and time at resolution $(\Delta x,\Delta t)=(0.393,0.586)$. Online iterations take less than 0.01 seconds, which includes building the linear system $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$, which is the most costly step.

In Figure 1 the average evolution of $E_2(\widehat{\mathbf{w}}^{(t)})$ and TPR$(\widehat{\mathbf{w}}^{(t)})$ is depicted for various noise levels σ_{NR} and memory capacities K_{mem}. The system is correctly identified for all trials when $K_{\text{mem}}\in\{13,17,21,25\}$ and $\sigma_{NR}\in\{0,0.001,0.01\}$, with relative errors E_2 less than 10^{-2} once the system is identified. For larger noise $\sigma_{NR}=0.1$, results stagnate at sub-optimal values, indicating that more data is needed to identify the system (note that $\mathbf{G}^{(t)}$ only has 214 rows). With $K_{\text{mem}}=5$ we recover the correct system only in the noise-less case ($\sigma_{NR}=0$), indicating that 5 points in time does not result in accurate resolution of the dynamics.

5.2. Variable-medium nonlinear wave equation in 2D (W2D)

$$\partial_{tt}u=c(t)(\partial_{xx}u+\partial_{yy}u)-u^3 \tag{5.5}$$

We examine a variable-medium nonlinear wave equation in 2D, given by equation (5.5), where the variable medium is modeled by the time-varying wavespeed

$$c(t)=1+(0.2)\frac{2}{\pi}\arctan(40\cos(2\pi(0.1)t)),$$

The wavespeed is a smoothed square wave and represents a system with abrupt speed modulation (see Figure 3 for depictions). We simulate the solution using a Fourier $\otimes$ Legendre spectral method in space with leap-frog timestepping. The exact data $\mathbf{U}_{exact}$ has dimensions $129\times403\times1639$ in (x,y,t) with resolution $(\Delta x,\Delta t)=(0.0156,0.0122)$. Each snapshot $\mathbf{U}^{(t)}$ is 0.42 megabytes (Mb) and online iterations take approximately 0.08 seconds.

Figure 2 shows robust recovery for $K_{\text{mem}}\in\{13,17,21,25\}$ up to $\sigma_{NR}=0.1$, with rapid identification for small noise. This is despite abrupt changes in the wavespeed c. For $K_{\text{mem}}=9$ we see recovery up to $\sigma_{NR}=0.001$, indicating that for larger noise 9 points in time is insufficient to discretize the integrals $\partial_{tt}\psi*u$ accurately, analogous to the case $K_{\text{mem}}=5$ for KS.

The left panel of Figure 2 shows that once the system is identified, abrupt changes in the wavespeed temporarily increase the coefficient error E_2, but the correct support $S^\star$ remains identified and the errors swiftly decay. In Figure 3 we plot the average learned wavespeed $\widehat{c}(t)$ as well as the maximum and minimum values of $\widehat{c}(t)$ attained over all 100 trials, revealing that increasing K_{mem} from 17 to 25 leads to a significant decrease in the variance of $\widehat{c}$ after the system has been identified. This is purely an affect of using the weak form to discretize the time derivatives, and demonstrates that even under large noise and abruptly changing coefficients, the algorithm is able to maintain support recovery and accuracy.

5.3. Wave equation in 3D

$$\partial_{tt}u=\partial_{xx}u+\partial_{yy}u+\partial_{zz}u \tag{5.6}$$

For our last example we treat the linear wave equation in 3D. Exact data $\mathbf{U}_{exact}$ has dimensions $128\times128\times128\times960$ in (x,y,t) with resolution $(\Delta x,\Delta t)=(0.0491,0.0122)$. Each snapshot $\mathbf{U}^{(t)}$ is 16.8 Mb and online iterations take approximately 1.3 seconds.

Results are depicted in Figure 4. We again find robust recovery for $K_{\text{mem}}\in\{13,17,21,25\}$ up to $\sigma_{NR}=0.1$, although in 5% of trials at $\sigma_{NR}=0.1$ the $K_{\text{mem}}=13$ case finds a spurious term $\approx-0.8u$. Even at $\sigma_{NR}=0.1$ the coefficients are accurate to more than 2 digits once recovered for $K_{\text{mem}}\geq17$. For $K_{\text{mem}}=9$ we see poor performance for the same reason as above with W2D, but now manifesting as

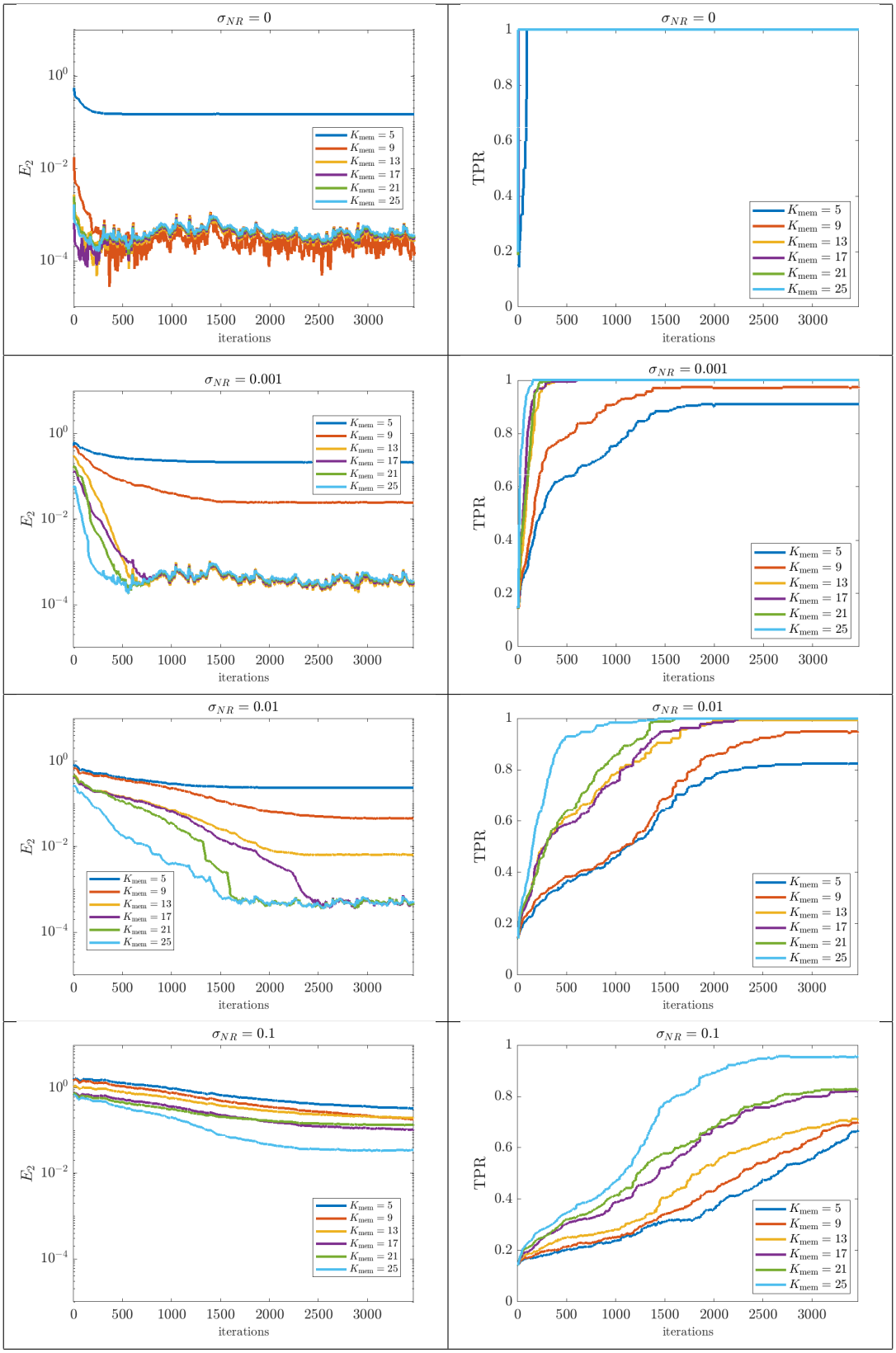

Figure 1: Online identification of the Kuramoto-Sivashinsky equation (5.4) for $K_{\text{mem}} \in \{5,9,13,17,21,25\}$ and (top to bottom) $\sigma_{NR} \in \{0,0.001,0.01,0.1\}$. Left: average coefficient error $E_2(\widehat{\mathbf{w}}^{(t)})$. Right: average total positivity ratio $\text{TPR}(\widehat{\mathbf{w}}^{(t)})$.

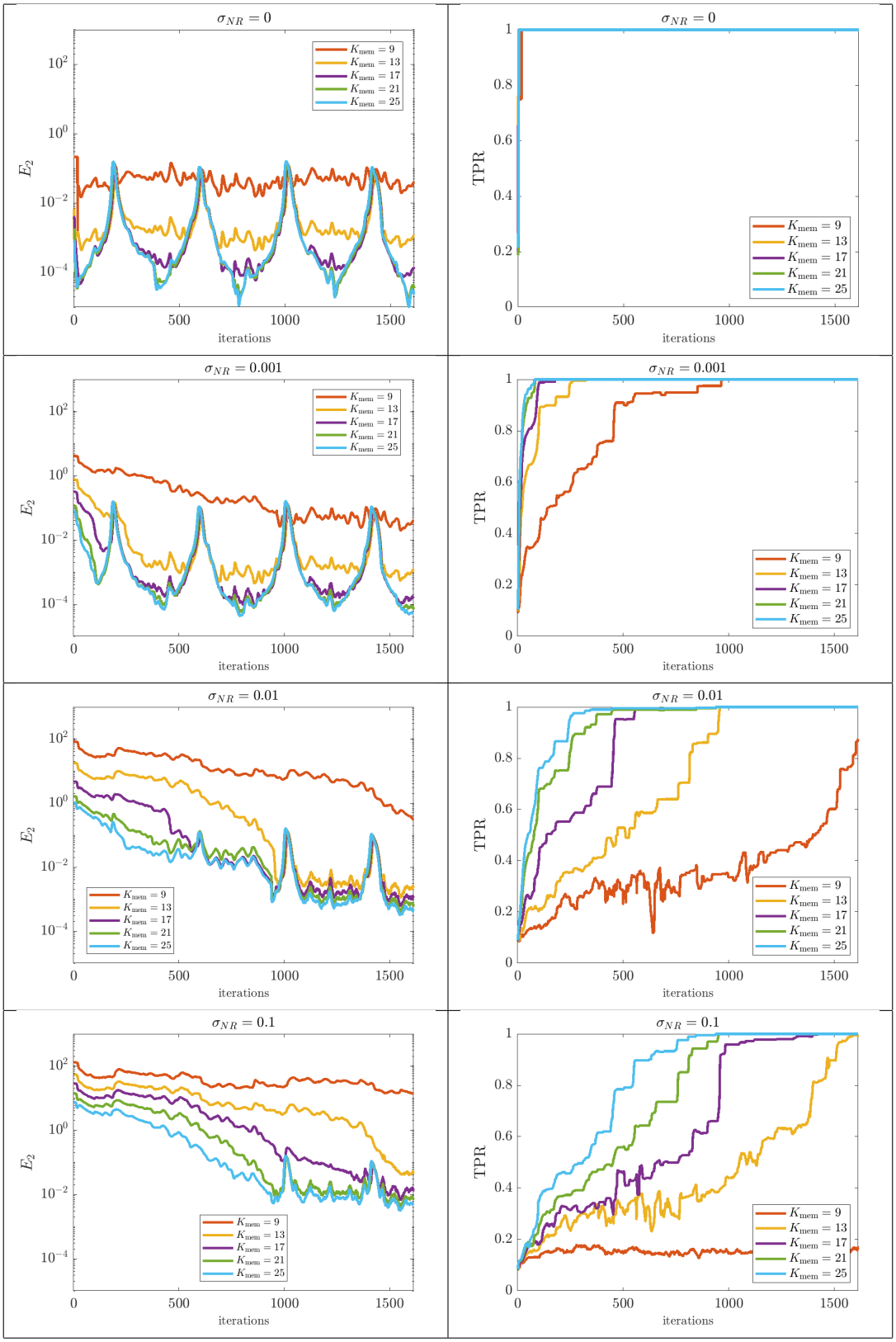

Figure 2: Online identification of the variable medium nonlinear wave equation (5.5) for $K_{\text{mem}} \in \{9, 13, 17, 21, 25\}$ and (top to bottom) $\sigma_{NR} \in \{0, 0.001, 0.01, 0.1\}$. Left: average coefficient error $E_2(\widehat{\mathbf{w}}^{(t)})$. Right: average total positivity ratio $\text{TPR}(\widehat{\mathbf{w}}^{(t)})$.

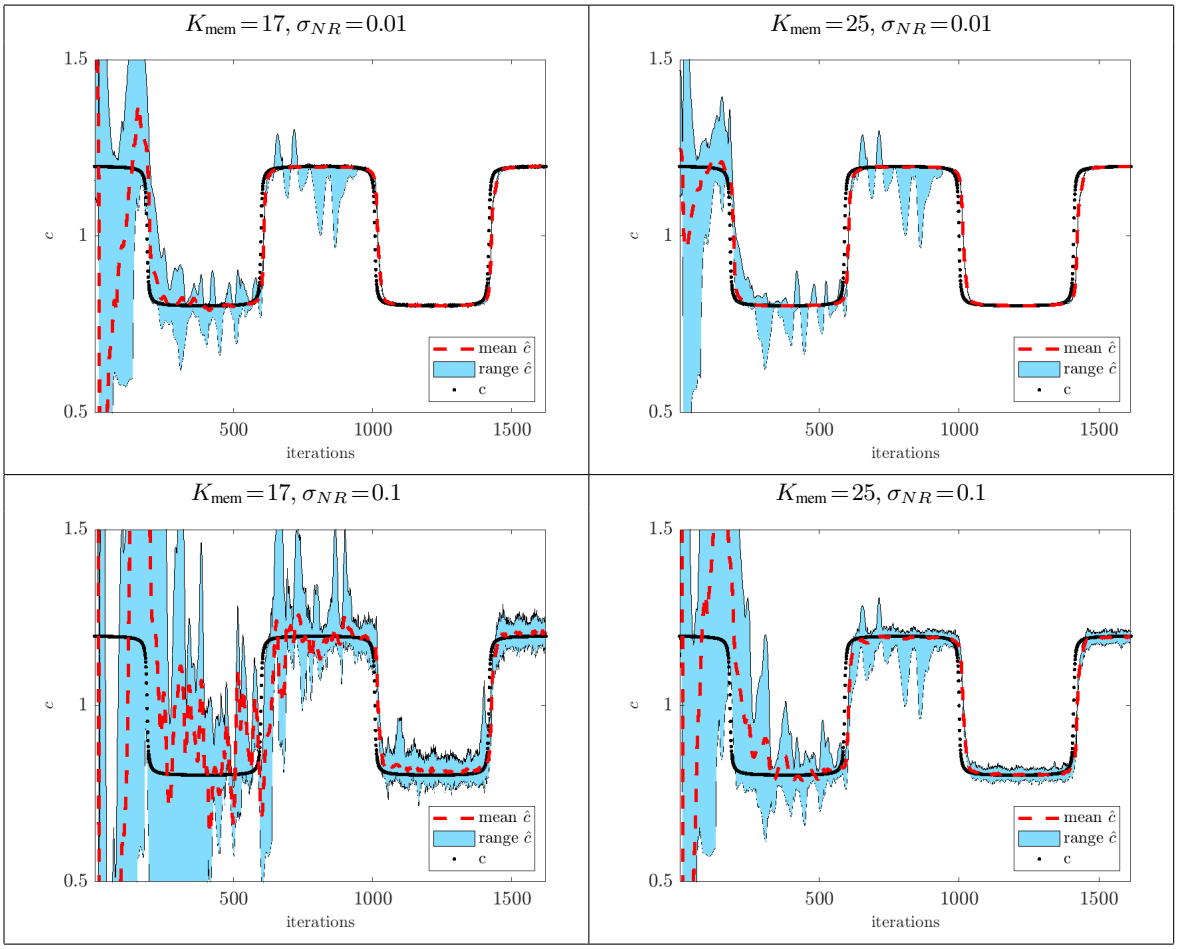

Figure 3: Online estimation of the wavespeed $c(t)$ (shown in black) for PDE (5.5). The average learned wavespeed $\widehat{c}(t)$ is shown in red while the blue shaded region shows the maximum and minimum values attained over all 100 trials. Notice the accuracy for later iterations when $\sigma_{NR}=0.01$, and the reduction in variance moving from $K_{\text{mem}}=17$ to $K_{\text{mem}}=25$ when $\sigma_{NR}=0.1$.

recovery of the spurious term $\approx -0.8u$, indicating that the inaccurate computation of $\partial_{tt}\psi * u$ produces spurious damping. This is not an altogether unreasonable affect if computing $\partial_{tt}\psi * u$ numerically is viewed as an attenuated second derivative calculation, although it does imply that higher-order time derivatives require more snapshots to be saved in memory.

6. Conclusions

We have demonstrated on several protoypical examples, and over a wide range of noise and memory scenarios, the viability of an online algorithm for PDE identification based on the weak-form sparse identification of nonlinear dynamics algorithm (WSINDy). The core of the method combines a weak-form discretization of candidate PDEs with the online proximal gradient descent algorithm applied directly to the least squares cost function with ℓ_0-pseudo-norm regularization (4.2). Compared with the more common approach of regularizing the ℓ_0-pseudo-norm (e.g. with $\|\cdot\|_1$ or weighted variants Candes et al. (2008)), we find that directly applying $\text{prox}_{\lambda\|\cdot\|_0}$, leading to hard thresholding, and adaptively selecting λ_t, exhibits good performance in efficiently identifying systems, handling noise, and tracking time-varying coefficients.

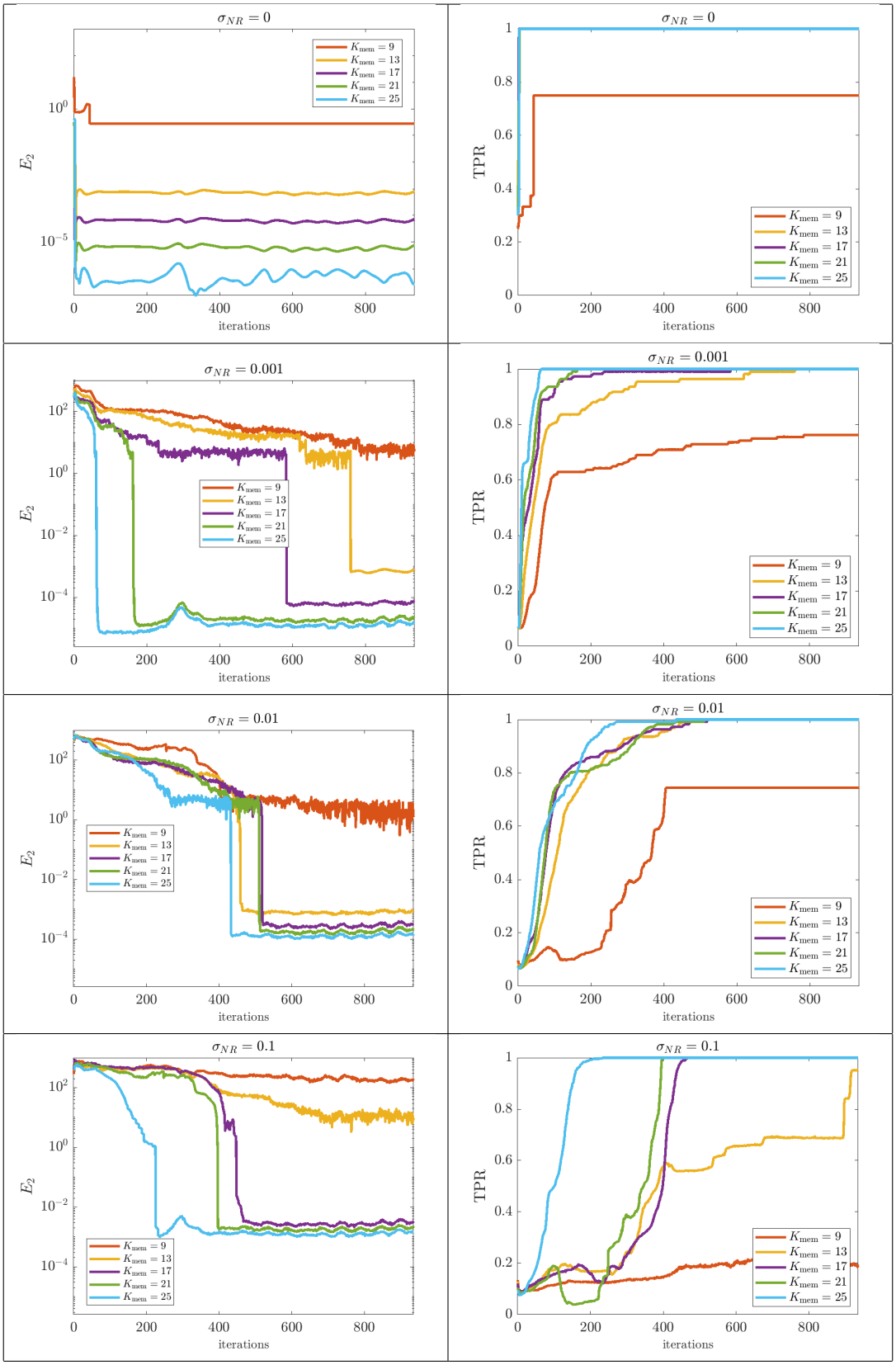

Figure 4: Online identification of the wave equation in three spatial dimensions (5.6) for $K_{\text{mem}} \in \{9, 13, 17, 21, 25\}$ and (top to bottom) $\sigma_{NR} \in \{0, 0.001, 0.01, 0.1\}$. Left: average coefficient error $E_2(\widehat{\mathbf{w}}^{(t)})$. Right: average total positivity ratio $\text{TPR}(\widehat{\mathbf{w}}^{(t)})$.

Numerical experiments with an abruptly changing wavespeed indicate that our method is a lightweight counterpart to existing methods for variable coefficients (e.g. Rudy et al. (2019)), which may be of independent interest in the control of wave equations in variable-media (Fante (1971); Felsen and Whitman (1970); Ning and Yan (2010); Seymour and Varley (1987); Chen (1979); Vila et al. (2017)). Examination of the wave equation in 3D also offers a different perspective on PDE identification in higher dimensions: problems with large datasets can be implemented in an online data-streaming fashion (not necessarily along the time axis as implemented here). It may therefore be advantageous from the standpoint of memory usage to solve certain batch problems in the online manner we have presented.

The algorithm's successes warrant further investigation in a number of areas. While we have characterized stationary points of the batch algorithm and proved boundedness of the average dynamic regret, we leave a more complete analysis to future work. In particular, one could analyze the error $\left\|\mathbf{w}^{(t)}-\mathbf{w}^{\star}(t)\right\|$ as a function of library Θ, test function ψ, data sampling rates $(\Delta x,\Delta t)$, memory size K_{mem}, noise ratio σ_{NR}, etc. It may also advantageous to design adaptive schemes which update Θ and ψ throughout the course of the algorithm, depending on the dynamics of the data and previously learned equations. We also note that an obvious next direction is to identify switching systems where the true support $S_t^{\star}$ changes with time. Nevertheless, the current framework is well-suited for a large variety of problems and opens the door to online PDE identification as well as the possibility of solving batch problems in an online manner.

7. Appendix

7.1. Column scaling and non-uniform thresholds

For stability, we normalize the columns of $\mathbf{G}^{(t)}$ at each step, defining $\widetilde{\mathbf{G}}^{(t)} = \mathbf{G}^{(t)}\mathbf{M}^{(t)}$ with

$$\mathbf{M}^{(t)} = \text{diag}\left(\left\|\mathbf{G}_1^{(t)}\right\|_2^{-1},...,\left\|\mathbf{G}_{IJ}^{(t)}\right\|_2^{-1}\right).$$

In particular, this allows for a larger stepsize $\widetilde{\alpha}_t = 1/\left\|(\widetilde{\mathbf{G}}^{(t)})^T\widetilde{\mathbf{G}}_{S_t}^{(t)}\right\|_2$ and leads to a reasonable estimate $\widetilde{\alpha}_t = 1/\sqrt{|S_t|IJ}$ for a stepsize that does not require computation of the matrix 2-norm.

For more flexibility, we allow for non-uniform thresholding. For a set of thresholds $\boldsymbol{\lambda} \in \mathbb{R}^{IJ}$, we define the non-uniform thresholding operator $H_{\boldsymbol{\lambda}}$ by

$$(H_{\boldsymbol{\lambda}}(\mathbf{x}))_i = \begin{cases} \mathbf{x}_i, & |\mathbf{x}_i| \geq \boldsymbol{\lambda}_i \\ 0, & \text{otherwise.} \end{cases}$$

This happens to be the proximal operator of the non-uniform ℓ_0-norm

$$\|\mathbf{x}\|_{0,\boldsymbol{\lambda}} := \sum_{i=1}^{IJ} \boldsymbol{\lambda}_i^2 \mathbb{1}_{\mathbb{R}\setminus\{0\}}(\mathbf{x}_i), \tag{7.1}$$

where $\|\mathbf{x}\|_{0,\boldsymbol{\lambda}} = \lambda^2\|\mathbf{x}\|_0$ when $\boldsymbol{\lambda} = (\lambda,...,\lambda)$. The resulting online cost function being minimized after incorporation of both non-uniform thresholding and column rescaling is

$$\widetilde{F}_t(\mathbf{w};\boldsymbol{\lambda}^{(t)}) = \frac{1}{2}\left\|\widetilde{\mathbf{G}}^{(t)}\mathbf{w} - \mathbf{b}^{(t)}\right\|_2^2 + \frac{1}{2}\|\mathbf{w}\|_{0,(\mathbf{M}^{(t)})^{-1}\boldsymbol{\lambda}^{(t)}}, \tag{7.2}$$

whose fixed points $\widetilde{\mathbf{w}}^{\star,(t)}$ coincide with those of the desired cost function

$$F_t(\mathbf{w};\boldsymbol{\lambda}^{(t)}) = \frac{1}{2}\left\|\mathbf{G}^{(t)}\mathbf{w} - \mathbf{b}^{(t)}\right\|_2^2 + \frac{1}{2}\|\mathbf{w}\|_{0,\boldsymbol{\lambda}^{(t)}} \tag{7.3}$$

after a diagonal transformation $\mathbf{w}^{\star,(t)} = \mathbf{M}^{(t)}\widetilde{\mathbf{w}}^{\star,(t)}$. With these two pieces, the online algorithm for (7.2) becomes

$$\begin{cases} \widetilde{\mathbf{w}}^{(t)} = (\mathbf{M}^{(t)})^{-1}\widehat{\mathbf{w}}^{(t)} \\ \widehat{\mathbf{w}}^{(t+\Delta t)} = H_{\widetilde{\alpha}_t\boldsymbol{\lambda}^{(t)}}\left(\mathbf{M}^{(t)}\left(\widetilde{\mathbf{w}}^{(t)} - \widetilde{\alpha}_t(\widetilde{\mathbf{G}}^{(t)})^T\left(\widetilde{\mathbf{G}}^{(t)}\widetilde{\mathbf{w}}^{(t)} - \mathbf{b}^{(t)}\right)\right)\right), \end{cases}$$

however this can equivalently be written in terms of the desired coefficients $\widehat{\mathbf{w}}^{(t)}$ as

$$\widehat{\mathbf{w}}^{(t+\Delta t)} = H_{\widetilde{\alpha}_t\boldsymbol{\lambda}^{(t)}}\left(\widehat{\mathbf{w}}^{(t)} - \widetilde{\alpha}_t(\mathbf{M}^{(t)})^2(\mathbf{G}^{(t)})^T\left(\mathbf{G}^{(t)}\widehat{\mathbf{w}}^{(t)} - \mathbf{b}^{(t)}\right)\right). \tag{7.4}$$

In direct analogy to the batch WSINDy thresholding scheme (3.4)-(3.5), we use thresholds $\boldsymbol{\lambda}^{(t)} = \max(1,\left\|\mathbf{b}^{(t)}\right\|\text{diag}(\mathbf{M}^{(t)}))\lambda_t$, which eliminate small coefficient values $\min_{i\in S_t}|\widehat{\mathbf{w}}_i^{(t)}| \geq \lambda_t$ as well as small terms in the sense of dominant balance with respect to $\mathbf{b}^{(t)}$:

$$\min_{i\in S_t}\frac{\left\|\mathbf{G}_i^{(t)}\widehat{\mathbf{w}}_i^{(t)}\right\|_2}{\left\|\mathbf{b}^{(t)}\right\|_2} \geq \lambda_t.$$

The update rule (4.4) for λ_t is unchanged after replacing $F_t(\mathbf{w};\lambda_t)$ with $F_t(\mathbf{w};\boldsymbol{\lambda}^{(t)})$ defined in (7.3).

7.2. Implementation and Computational Complexity

The offline phase has four components:

1. Initialize hyperparameters $\psi(\mathbf{x},t)=\phi(\mathbf{x})\theta(t)$, $\Theta=\{D^{\boldsymbol{\alpha}^{(i)}}f_j\}_{i=0,j=1}^{I,J}$, $\Delta\lambda$, $\lambda_{\max}$, λ_0, where the test function ψ is either prescribed manually or selected using the changepoint algorithm from Messenger and Bortz (2021a) using the initial K_{mem} slices $\{\mathbf{U}^{(0)},...,\mathbf{U}^{((K_{\text{mem}}-1)\Delta t)}\}$.

2. Compute and store the Fourier transforms $\{\widehat{D^{\boldsymbol{\alpha}^{(i)}}\psi}\}_{i=0}^{I}$ to reuse at each step when computing convolutions (recall ψ is separable so this storage cost is negligible).

3. Compute initial library of spatially integrated terms

$$\Psi := \{\Psi^{(t)}\}_{t=0}^{(K_{\text{mem}}-1)\Delta t} := \left\{\left\{D^{\boldsymbol{\beta}^{(i)}}\phi * f_j(\mathbf{U}^{(t)})(\mathcal{Q}_{\mathbf{x}},t)\right\}_{i=0,j=1}^{I,J}\right\}_{t=0}^{(K_{\text{mem}}-1)\Delta t}$$

   where $\boldsymbol{\beta}^{(i)}=(\boldsymbol{\alpha}_1^{(i)},...,\boldsymbol{\alpha}_d^{(i)})$ is the spatial part of the multi-index $\boldsymbol{\alpha}^{(i)}$ operating on the spatial part ϕ of the test function ψ (recall that $\mathcal{Q}_{\mathbf{x}}$ is the set of spatial points over which convolutions are evaluated, also equal to the number of rows in $\mathbf{G}^{(t)}$).

4. Compute initial weights $\widehat{\mathbf{w}}^{(0)}=(\mathbf{G}^{(0)})^{\dagger}\mathbf{b}^{(0)}$ where $\mathbf{b}^{(0)}$ and $\mathbf{G}^{(0)}$ are obtained by integrating the elements of Ψ in time against the corresponding temporal test functions $D^{\boldsymbol{\alpha}_{d+1}^{(i)}}\theta$.

For each t in the online phase we compute $\Psi^{(t)}$ using only the incoming slice $\mathbf{U}^{(t)}$, which replaces $\Psi^{(t-K_{\text{mem}}\Delta t)}$ in memory. $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$ are then computed by integrating the elements of Ψ in time against the corresponding temporal test functions $D^{\boldsymbol{\alpha}_{d+1}^{(i)}}\theta$, which amounts to a series of dot products between length-K_{mem} vectors. Computation of $\mathbf{G}^{(t)}$ at each time t thus requires $J|\mathbf{X}|$ function evaluations $f_j(\mathbf{U}^{(t)})$ (each counted as 1 floating point operation (flop)) followed by IJ convolutions against $D^{\boldsymbol{\beta}^{(i)}}\phi$, and finally integration in time. The total flop count at each step is at most

$$J|\mathbf{X}|\left(1+CI\log N+2IK_{\text{mem}}\frac{|\mathcal{Q}_{\mathbf{X}}|}{|\mathbf{X}|}\right)$$

where C is such that $\mathbf{x}*\mathbf{y}$ costs $CN\log N$ using FFTs for length-N vectors $\mathbf{x}$ and $\mathbf{y}$, minus the cost of one FFT (since we have precomputed these for $D^{\boldsymbol{\beta}^{(i)}}\phi$) and $N\approx|\mathbf{X}|^{1/d}$ is the one-dimensional length scale of the data. In other words, only

$$F=J\left(1+CI\log N+IK_{\text{mem}}\frac{|\mathcal{Q}_{\mathbf{x}}|}{|\mathbf{X}|}\right)$$

flops are performed per incoming data point in $\mathbf{U}^{(t)}$ (and a more careful analysis leads to a lower cost in the factor $CI\log N$ by incorporating the subsampling $\mathbf{X}\to\mathcal{Q}_{\mathbf{x}}$). Note that F does not depend on the spatial dimension d of the data set (except through library term I, which might increase with d as more differential operators become added). The total working memory W to store Ψ and $(\mathbf{G}^{(t)},\mathbf{b}^{(t)})$ as outlined above is given by $W=IJ|\mathcal{Q}_{\mathbf{x}}|K_{\text{mem}}+(I+1)J|\mathcal{Q}_{\mathbf{x}}|$ double-precision floating point numbers (DPs).

Remark 7.1. There are several natural choices to consider to either decrease storage restrictions or increase computational speed. However, it is not clear that the anticipated savings will manifest. For instance, we

could instead store the spatial Fourier transforms of the nonlinearities $\{\widehat{f_j(\mathbf{U}^{(t)})}\}_{j=1,t=\ell\Delta t}^{J,(\ell+K_{\text{mem}}-1)\Delta t}$, resulting in a working memory of $J\cdot K_{\text{mem}}\cdot|\mathbf{X}|$ instead of $IJ|\mathcal{Q}_{\mathbf{X}}|K_{\text{mem}}$ to store Ψ. This would require that we compute spatial convolutions over all K_{mem} time slices at each time point, instead of spatial convolutions over just the incoming time slice $\mathbf{U}^{(t)}$, hence resulting in a K_{mem}-fold increase in computation time, as this is the leading-order cost. In addition, the storage "savings" may actually be worse, specifically if $I|\mathcal{Q}_{\mathbf{x}}|\leq|\mathbf{X}|$. We believe that the method outlined above provides a near-optimal balance of computational complexity and storage requirements, with a heavier emphasis on reducing computational complexity.

7.3. Proof of Lemma 4.1

Consider $\mathbf{w}$ such that one of the following holds:

(i) $\mathbf{w}$ is a local minimizer of (3.3)

(ii) $\mathbf{w}=H_\lambda\big(\mathbf{w}-\mathbf{G}^T(\mathbf{G}\mathbf{w}-\mathbf{b})\big)$

(iii) With $S=\text{supp}(\mathbf{w})$, we have that $\mathbf{w}_S\in\text{argmin}_{\mathbf{z}}\|\mathbf{G}_S\mathbf{z}-\mathbf{b}\|_2^2$ and

$$\max_{i\in S^c}\big|\mathbf{G}_i^T(\mathbf{G}\mathbf{w}-\mathbf{b})\big|<\lambda\leq\min_{i\in S}|\mathbf{w}_i|.$$

Then it holds that $(ii)\iff(iii)\implies(i)$. Moreover, if $\mathbf{w}$ a global minimizer, then $(i)\implies(iii)$.

Proof $(iii)\implies(ii)$ is immediate. To show $(ii)\implies(iii)$, let $S=\text{supp}(\mathbf{w})$. Then we have

$$\mathbf{w}_S=\mathbf{w}_S-\mathbf{G}_S^T(\mathbf{G}\mathbf{w}-\mathbf{b}),$$

which implies that $\min_{i\in S}|\mathbf{w}_i|\geq\lambda$ so that $\mathbf{G}_S^T\mathbf{G}_S\mathbf{w}_S=\mathbf{G}_S^T\mathbf{b}$, so that $\mathbf{w}_S\in\text{argmin}_{\mathbf{z}}\|\mathbf{G}_S\mathbf{z}-\mathbf{b}\|_2^2$. On S^c we have

$$H_\lambda\big(\mathbf{G}_{S^c}^T(\mathbf{G}\mathbf{w}-\mathbf{b})\big)=0\implies\max_{i\in S^c}\big|\mathbf{G}_i^T(\mathbf{G}\mathbf{w}-\mathbf{b})\big|<\lambda.$$

To show that (ii) and (iii) imply (i), we note that under usual assumptions of two closed, convex and proper functions f and g, we have

$$\mathbf{w}\in\text{prox}_g(\mathbf{w}-\partial f(\mathbf{w}))\iff 0\in\partial f(\mathbf{w})+\partial g(\mathbf{w})\implies\mathbf{w}\in\text{argmin}(f+g),$$

however $\|\cdot\|_0$ is clearly not convex[7]. Instead we can directly show that for a perturbed vector $\widetilde{\mathbf{w}}=\mathbf{w}+\boldsymbol{\eta}$, for suitably small $\|\boldsymbol{\eta}\|$ the objective is non-decreasing. Using that $\mathbf{w}_S\in\text{argmin}_{\mathbf{z}}\|\mathbf{G}_S\mathbf{z}-\mathbf{b}\|_2^2$, let $\mathbf{P}_S^\perp$ be the projection onto $\{\text{span}(\mathbf{G}_S)\}^\perp$. The difference in objective F is then given by

$$F(\widetilde{\mathbf{w}};\lambda)-F(\mathbf{w};\lambda)=\frac{1}{2}\left(\left\|\mathbf{P}_S^\perp\mathbf{b}+\mathbf{G}\boldsymbol{\eta}\right\|_2^2-\left\|\mathbf{P}_S^\perp\mathbf{b}\right\|_2^2\right)+\frac{\lambda^2}{2}(\|\widetilde{\mathbf{w}}\|_0-\|\mathbf{w}\|_0)$$

$$=\frac{1}{2}\|\mathbf{G}\boldsymbol{\eta}\|_2^2+\left\langle\mathbf{P}_S^\perp\mathbf{b},\mathbf{G}\boldsymbol{\eta}\right\rangle+\frac{\lambda^2}{2}(\|\widetilde{\mathbf{w}}\|_0-\|\mathbf{w}\|_0).$$

7. In fact the subdifferential $\partial\|\cdot\|_0(\mathbf{w})=\emptyset$ unless $\mathbf{w}=\mathbf{0}$, upon which $\partial\|\cdot\|_0(\mathbf{w})=\{\mathbf{0}\}$.

If $\text{supp}(\boldsymbol{\eta}) \subset \text{supp}(\mathbf{w})$ and $\|\boldsymbol{\eta}\|_\infty < \lambda$, then $\|\widetilde{\mathbf{w}}\|_0 = \|\mathbf{w}\|_0$ and $\langle \mathbf{P}_S^\perp \mathbf{b}, \mathbf{G}\boldsymbol{\eta} \rangle = 0$, hence $F(\widetilde{\mathbf{w}};\lambda) - F(\mathbf{w};\lambda) \geq 0$, with equality only if $\mathbf{G}\boldsymbol{\eta} = \mathbf{0}$, which in particular is not possible when $\mathbf{G}_S$ is full rank unless $\boldsymbol{\eta} = 0$. If $\text{supp}(\boldsymbol{\eta}) \notin S$ and $\|\boldsymbol{\eta}\|_\infty < \lambda$, then $\mathbf{P}_S^\perp \mathbf{b} = \mathbf{0}$ implies a strict increase in F, while if $\mathbf{P}_S^\perp \mathbf{b} \neq \mathbf{0}$ then

$$\|\boldsymbol{\eta}_{S^c}\|_2 < \epsilon := \frac{\lambda^2}{2} \frac{1}{\|\mathbf{P}_S^\perp \mathbf{b}\|_2 \|\mathbf{G}_{S^c}\|_2},$$

implies a strict increase in F. To see this, note that $\langle \mathbf{P}_S^\perp, \mathbf{G}\boldsymbol{\eta} \rangle = \langle \mathbf{P}_S^\perp, \mathbf{G}_{S^c}\boldsymbol{\eta}_{S^c} \rangle$ implies the bound

$$F(\widetilde{\mathbf{w}};\lambda) - F(\mathbf{w};\lambda) \geq -\left\|\mathbf{P}_S^\perp \mathbf{b}\right\|_2 \|\mathbf{G}_{S^c}\|_2 \|\boldsymbol{\eta}_{S^c}\|_2 + \frac{\lambda^2}{2} > 0.$$

Note that ϵ is not tight. Combining these conditions gives a ball around $\mathbf{w}$ over which F is non-decreasing, hence $\mathbf{w}$ is a local min. Finally, that $\mathbf{w}$ a global minimizer implies (iii) can be found in Zhang and Schaeffer (2019).

7.4. Proof of Theorem 4.1

For convenience, we restate the theorem here. Without loss of generality in the following we set $\Delta t = 1$.

Theorem 7.1. *Let $\sigma_{1,t}$ and $\sigma_{n,t}$ denote the first and last singular values of the matrix $\mathbf{G}^{(t)} \in \mathbb{R}^{m \times n}$. Assume the following: $\max_t \lambda_t \leq \overline{\lambda} < \infty$, $\min_t \sigma_{n,t} \geq \overline{\sigma}_{\min} > 0$, $\max_t \sigma_{1,t} \leq \overline{\sigma}_{\max}$, and $\sup_t \alpha_t < \sigma_{\max}^{-2}$, $\inf_t \alpha_t > 0$. In addition, assume that the global minimizer $\mathbf{w}^\star(t)$ of $F_t(\mathbf{w};\lambda_t)$ is unique for every t and satisfies $|S_t^\star| \geq \overline{s} > 0$ where $S^\star = supp(\mathbf{w}^\star(t))$. Finally, assume that the tracking gap is globally bounded: $\|\mathbf{w}^\star(t) - \mathbf{w}^\star(t+1)\|_2 := d_t \leq \overline{d}$. Then the dynamic regret* (4.5) *grows at-worst linearly:*

$$Reg_D(T) \leq C_1 + C_2 T$$

for some $C_1 > 0$ and $C_2 > 0$. In particular, $\frac{1}{T} Reg_D(T)$ remains bounded.

Proof First we decompose $F_t(\mathbf{w};\lambda_t) = g_t(\mathbf{w}) + h_t(\mathbf{w}) = \left\|\mathbf{G}^{(t)}\mathbf{w} - \mathbf{b}^{(t)}\right\|_2^2 + \lambda^2 \|\mathbf{w}\|_0$. We can bound the difference in g_t as follows:

$$\begin{aligned} g_t(\mathbf{w}^{(t)}) - g_t(\mathbf{w}^\star(t)) &= \left\|\mathbf{G}^{(t)}\mathbf{w}^{(t)}\right\|_2^2 - \left\|\mathbf{G}^{(t)}\mathbf{w}^\star(t)\right\|_2^2 - 2\left\langle \mathbf{b}^{(t)}, \mathbf{G}^{(t)}\left(\mathbf{w}^{(t)} - \mathbf{w}^\star(t)\right)\right\rangle \\ &= \left\|\mathbf{G}^{(t)}(\mathbf{w}^{(t)} - \mathbf{w}^\star(t))\right\|_2^2 - 2\left\langle \mathbf{G}^{(t)}(\mathbf{w}^{(t)} - \mathbf{w}^\star(t)), \mathbf{G}^{(t)}\mathbf{w}^\star(t) - \mathbf{b}^{(t)}\right\rangle \end{aligned}$$

taking $|\cdot|$ of both sides and noting from the Lemma that $\left\|(\mathbf{G}^{(t)})^T(\mathbf{G}^{(t)}\mathbf{w}^\star(t) - \mathbf{b}^{(t)})\right\|_\infty < \lambda_t$ implies that

$$\begin{aligned} g_t(\mathbf{w}^{(t)}) - g_t(\mathbf{w}^\star(t)) &\leq \overline{\sigma}_{\max}^2 \left\|\mathbf{w}^{(t)} - \mathbf{w}^\star(t)\right\|_2^2 + 2\lambda_t \sqrt{|(S^\star)^c|} \left\|\mathbf{w}^{(t)} - \mathbf{w}^\star(t)\right\|_2 \\ &\leq \overline{\sigma}_{\max}^2 \left\|\mathbf{w}^{(t)} - \mathbf{w}^\star(t)\right\|_2^2 + 2\overline{\lambda}\sqrt{n - \overline{s}} \left\|\mathbf{w}^{(t)} - \mathbf{w}^\star(t)\right\|_2. \end{aligned}$$

For h_t we have simply

$$\left|h_t(\mathbf{w}^{(t)}) - h_t(\mathbf{w}^\star(t))\right| = \lambda_t^2 ||S_t| - |S_t^\star|| \leq \overline{\lambda}^2 (n - \overline{s}).$$

For any vectors $\mathbf{x},\mathbf{y}\in\mathbb{R}^n$, it holds that

$$\|H_\lambda(\mathbf{x})-H_\lambda(\mathbf{y})\|_2\leq\|\mathbf{x}-\mathbf{y}\|_2+\lambda\sqrt{|S_x\triangle S_y|}$$

where $S_x\triangle S_y=(S_x\cap S_y^c)\cup(S_y\cap S_x^c)$ is the symmetric difference of the sets $S_x=\text{supp}(H_\lambda(\mathbf{x}))$ and $S_y=\text{supp}(H_\lambda(\mathbf{y}))$. This implies, together with stationarity of $\mathbf{w}^\star(t)$,

$$\begin{aligned}
&\left\|\mathbf{w}^{(t+1)}-\mathbf{w}^\star(t)\right\|_2\\
&=\left\|H_{\alpha_t\lambda_t}\left(\mathbf{w}^{(t)}-\alpha_t(\mathbf{G}^{(t)})^T\left(\mathbf{G}^{(t)}\mathbf{w}^{(t)}-\mathbf{b}^{(t)}\right)\right)-H_{\alpha_t\lambda_t}\left(\mathbf{w}^\star(t)-\alpha_t(\mathbf{G}^{(t)})^T\left(\mathbf{G}^{(t)}\mathbf{w}^\star(t)-\mathbf{b}^{(t)}\right)\right)\right\|_2\\
&\leq\left\|\left(\mathbf{I}-\alpha_t(\mathbf{G}^{(t)})^T\mathbf{G}^{(t)}\right)(\mathbf{w}^{(t)}-\mathbf{w}^\star(t))\right\|_2+\alpha_t\lambda_t\sqrt{|S_{t+1}\triangle S_t^\star|}\\
&\leq\max\left(|1-\alpha_t\sigma_{1,t}^2|,|1-\alpha_t\sigma_{n,t}^2|\right)\left\|\mathbf{w}^{(t)}-\mathbf{w}^\star(t)\right\|_2+\alpha_t\lambda_t\sqrt{|S_{t+1}\triangle S_t^\star|}\\
&:=\rho_t\left\|\mathbf{w}^{(t)}-\mathbf{w}^\star(t)\right\|_2+\alpha_t\lambda_t\sqrt{|S_{t+1}\triangle S_t^\star|}.
\end{aligned}$$

Using that $\left\|\mathbf{w}^{(t+1)}-\mathbf{w}^\star(t+1)\right\|_2\leq\left\|\mathbf{w}^{(t+1)}-\mathbf{w}^\star(t)\right\|_2+d_t$, we have the recurrence relation

$$\left\|\mathbf{w}^{(t+1)}-\mathbf{w}^\star(t+1)\right\|_2\leq\rho_t\left\|\mathbf{w}^{(t)}-\mathbf{w}^\star(t)\right\|_2+d_t+\alpha_t\lambda_t\sqrt{|S_{t+1}\triangle S_t^\star|},\tag{7.5}$$

where, by assumptions on $\sigma_{1,t},\sigma_{n,t}$ and α_t, it holds that $\max_t\rho_t\leq\overline{\rho}$ for some $\overline{\rho}<1$, hence we get the bound

$$\left\|\mathbf{w}^{(t)}-\mathbf{w}^\star(t)\right\|_2\leq\overline{\rho}^t\left\|\mathbf{w}^{(0)}-\mathbf{w}^\star(0)\right\|_2+(\overline{d}+\overline{\alpha}\overline{\lambda}\sqrt{n})\sum_{s=0}^{t}\overline{\rho}^s\leq\overline{\rho}^t\left\|\mathbf{w}^{(0)}-\mathbf{w}^\star(0)\right\|_2+\frac{\overline{d}+\overline{\alpha}\overline{\lambda}\sqrt{n}}{1-\overline{\rho}}.\tag{7.6}$$

We note in passing that this implies a uniform error bound on $\left\|\mathbf{w}^{(t)}-\mathbf{w}^\star(t)\right\|_2$ which asymptotically depends only on the tracking gap d_t and the support difference $|S_t\triangle S_t^\star|$. Finally, using this bound and previous calculations for g and h, we get

$$Reg_D(T)\leq\widetilde{C}_1\sum_{t=0}^{T}\overline{\rho}^t+C_2T\leq C_1+C_2T$$

where

$$C_1=\frac{\widetilde{C}_1}{1-\overline{\rho}}=\frac{\overline{\sigma}_{\max}^2\overline{\rho}}{1-\overline{\rho}}\left\|\mathbf{w}^{(0)}-\mathbf{w}^\star(0)\right\|_2^2+\frac{2\left\|\mathbf{w}^{(0)}-\mathbf{w}^\star(0)\right\|_2}{(1-\overline{\rho})^2}\left(\overline{\lambda}\sqrt{n-\overline{s}}+\frac{(\overline{d}+\overline{\alpha}\overline{\lambda}\sqrt{n})\overline{\sigma}_{\max}^2}{1-\overline{\rho}}\right)\tag{7.7}$$

$$C_2=(\overline{\sigma}_{\max}^2-1)\left(\frac{\overline{d}+\overline{\alpha}\overline{\lambda}\sqrt{n}}{1-\overline{\rho}}\right)^2+\left(\frac{\overline{d}+\overline{\alpha}\overline{\lambda}\sqrt{n}}{1-\overline{\rho}}+\overline{\lambda}\sqrt{n-\overline{s}}\right)^2.\tag{7.8}$$

This completes the proof.

Acknowledgments

This research was supported in part by the NSF/NIH Joint DMS/NIGMS Mathematical Biology Initiative grant R01GM126559, in part by the NSF Mathematical Biology MODULUS grant 2054085, and in part by the NSF Computing and Communications Foundations grant 1815983. This work also utilized resources from the University of Colorado Boulder Research Computing Group, which is supported by the National Science Foundation (awards ACI-1532235 and ACI-1532236), the University of Colorado Boulder, and Colorado State University. Code is available at https://github.com/MathBioCU/WSINDy_PDE_OL.git.

References

Joseph Bakarji and Daniel M Tartakovsky. Data-driven discovery of coarse-grained equations. Journal of Computational Physics, 434:110219, 2021.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. Operations research, 63(5):1227–1244, 2015.

Thomas Blumensath and Mike E Davies. Iterative thresholding for sparse approximations. Journal of Fourier analysis and Applications, 14(5):629–654, 2008.

Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. Applied and computational harmonic analysis, 27(3):265–274, 2009.

Lorenzo Boninsegna, Feliks Nüske, and Cecilia Clementi. Sparse learning of stochastic dynamical equations. The Journal of chemical physics, 148(24):241723, 2018.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the national academy of sciences, 113(15):3932–3937, 2016.

Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. Journal of Fourier analysis and applications, 14(5):877–905, 2008.

Goong Chen. Control and stabilization for the wave equation in a bounded domain. SIAM Journal on Control and Optimization, 17(1):66–81, 1979.

Emiliano Dall'Anese, Andrea Simonetto, Stephen Becker, and Liam Madden. Optimization and learning with information streams: Time-varying algorithms and applications. IEEE Signal Processing Magazine, 37(3):71–83, 2020.

Rishabh Dixit, Amrit Singh Bedi, Ruchi Tripathi, and Ketan Rajawat. Online learning with inexact proximal online gradient descent algorithms. IEEE Transactions on Signal Processing, 67(5):1338–1352, 2019.

David L Donoho, Yaakov Tsaig, Iddo Drori, and Jean-Luc Starck. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. IEEE transactions on Information Theory, 58(2):1094–1121, 2012.

Jianqing Fan and Yuan Liao. Endogeneity in high dimensions. Annals of statistics, 42(3):872, 2014.

R Fante. Transmission of electromagnetic waves into time-varying media. IEEE Transactions on Antennas and Propagation, 19(3):417–424, 1971.

Salar Fattahi, Nikolai Matni, and Somayeh Sojoudi. Learning sparse dynamical systems from a single sample trajectory. In 2019 IEEE 58th Conference on Decision and Control (CDC), pages 2682–2689. IEEE, 2019.

L Felsen and G Whitman. Wave propagation in time-varying media. IEEE Transactions on Antennas and Propagation, 18(2):242–253, 1970.

Dylan Foster, Tuhin Sarkar, and Alexander Rakhlin. Learning nonlinear dynamical systems from a single trajectory. In Learning for Dynamics and Control, pages 851–861. PMLR, 2020.

Simon Foucart and Holger Rauhut. A Mathematical Introduction to Compressive Sensing. Birkhäuser Basel, 2013. ISBN 0817649476.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. The elements of statistical learning: data mining, inference, and prediction, volume 2. Springer, 2009.

Elad Hazan. Efficient algorithms for online convex optimization and their applications. Princeton University, 2006.

Elad Hazan and Satyen Kale. On stochastic and worst-case models for investing. Advances in Neural Information Processing Systems, 22, 2009.

Steven C.H. Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. Neurocomputing, 459:249–289, October 2021. ISSN 09252312. doi: 10.1016/j.neucom.2021.04.112. URL https://linkinghub.elsevier.com/retrieve/pii/S0925231221006706.

Jialei Wang, Peilin Zhao, Steven C. H. Hoi, and Rong Jin. Online Feature Selection and Its Applications. IEEE Transactions on Knowledge and Data Engineering, 26 (3):698–710, March 2014. ISSN 1041-4347. doi: 10.1109/TKDE.2013.32. URL http://ieeexplore.ieee.org/document/6522405/.

Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. Proceedings of the Royal Society A, 474(2219):20180335, 2018.

Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration. In 2018 IEEE conference on decision and control (CDC), pages 6059–6066. IEEE, 2018.

Yannis Kopsinis, Konstantinos Slavakis, and Sergios Theodoridis. Online Sparse System Identification and Signal Reconstruction Using Projections Onto Weighted ℓ_1 Balls. IEEE Transactions on Signal Processing, 59(3):936–952, March 2011. ISSN 1053-587X, 1941-0476. doi: 10.1109/TSP.2010.2090874.

L. Ljung. System identification: theory for the user. 2nd edition Prentice-Hall, Upper Saddle River, NJ, 1999.

Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. Pde-net: Learning pdes from data. In International Conference on Machine Learning, pages 3208–3216. PMLR, 2018.

Zichao Long, Yiping Lu, and Bin Dong. Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. Journal of Computational Physics, 399:108925, 2019.

Suryanarayana Maddu, Bevan L Cheeseman, Ivo F Sbalzarini, and Christian L Müller. Stability selection enables robust learning of differential equations from limited noisy data. Proceedings of the Royal Society A, 478(2262):20210916, 2022.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. The annals of statistics, 34(3):1436–1462, 2006.

Daniel A. Messenger and David M. Bortz. Weak SINDy For Partial Differential Equations. J. Comput. Phys., 443:110525, October 2021a. doi: 10.1016/j.jcp.2021.110525.

Daniel A. Messenger and David M. Bortz. Weak SINDy: Galerkin-Based Data-Driven Model Selection. SIAM Multiscale Model. Simul., 19(3):1474–1497, 2021b.

Daniel A Messenger and David M Bortz. Learning mean-field equations from particle data using wsindy. Physica D: Nonlinear Phenomena, 439:133406, 2022.

Mila Nikolova. Description of the minimizers of least squares regularized with ℓ_0-norm. uniqueness of the global minimizer. SIAM Journal on Imaging Sciences, 6(2):904–937, 2013.

Zhen-Hu Ning and Qing-Xu Yan. Stabilization of the wave equation with variable coefficients and a delay in dissipative boundary feedback. Journal of Mathematical Analysis and Applications, 367(1): 167–173, 2010.

Tong Qin, Kailiang Wu, and Dongbin Xiu. Data driven governing equations approximation using deep neural networks. Journal of Computational Physics, 395:620–635, 2019.

Samuel Rudy, Alessandro Alla, Steven L Brunton, and J Nathan Kutz. Data-driven identification of parametric partial differential equations. SIAM Journal on Applied Dynamical Systems, 18(2):643–660, 2019.

Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. Science Advances, 3(4):e1602614, 2017.

Yahya Sattar and Samet Oymak. Non-asymptotic and accurate learning of nonlinear dynamical systems. arXiv preprint arXiv:2002.08538, 2020.

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

Hayden Schaeffer and Scott G McCalla. Sparse model selection via integral terms. Physical Review E, 96(2):023302, 2017.

Brian Seymour and Eric Varley. Exact representations for acoustical waves when the sound speed varies in space and time. Studies in Applied Mathematics, 76(1):1–35, 1987.

Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. In Conference On Learning Theory, pages 439–473. PMLR, 2018.

Lizhe Sun, Mingyuan Wang, Yangzi Guo, and Adrian Barbu. A novel framework for online supervised learning with feature selection. arXiv preprint arXiv:1803.11521, 2018.

Javier Vila, Raj Kumar Pal, Massimo Ruzzene, and Giuseppe Trainiti. A bloch-based procedure for dispersion analysis of lattices with periodic time-varying properties. Journal of Sound and Vibration, 406:363–377, 2017.

Kailiang Wu and Dongbin Xiu. Data-driven deep learning of partial differential equations in modal space. Journal of Computational Physics, 408:109307, 2020.

Zhenhuan Yang, Baojian Zhou, Yunwen Lei, and Yiming Ying. Stochastic Hard Thresholding Algorithms for AUC Maximization. In 2020 IEEE International Conference on Data Mining (ICDM), pages 741–750, Sorrento, Italy, November 2020. IEEE.

Yilun Chen, Yuantao Gu, and Alfred O. Hero. Sparse LMS for system identification. In 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 3125–3128, Taipei, Taiwan, April 2009. IEEE.

Yuantao Gu, Jian Jin, and Shunliang Mei. ℓ_0 norm constraint LMS algorithm for sparse system identification. IEEE Signal Processing Letters, 16(9):774–777, September 2009. ISSN 1070-9908, 1558-2361. doi: 10.1109/LSP.2009.2024736.

Tingting Zhai, Frederic Koriche, Hao Wang, and Yang Gao. Tracking Sparse Linear Classifiers. IEEE Transactions on Neural Networks and Learning Systems, 30(7):2079–2092, July 2019. ISSN 2162-237X, 2162-2388. doi: 10.1109/TNNLS.2018.2877433. URL https://ieeexplore.ieee.org/document/8533616/.

Linan Zhang and Hayden Schaeffer. On the convergence of the SINDy algorithm. Multiscale Modeling & Simulation, 17(3):948–972, 2019.

Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In Proceedings of the 20th international conference on machine learning (icml-03), pages 928–936, 2003.