WSINDy For PDEs: User's Guide

Daniel A. Messenger

January 7, 2021

1 `wsindy_pde_script.m`

All model selection is executed from this script, which is set up to demonstrate the WSINDy algorithm on PDEs used in [1].

`%% Load data`

This code block is used to load the "exact" data and computational grid as well as coarsen and add noise if desired. Solution data should be loaded as a cell array `U_exact` with each cell holding the data for each state variable (e.g. for a reaction-diffusion system with two components u and v, `U_exact{1}` holds data for u and `U_exact{2}` holds data for v). The computational grid should be loaded as a cell array `xs` with each cell containing the grid points along each dimension $(x_1, \dots, x_D, t)$ such that `xs{1}` holds points along the spatial coordinate x_1, `xs{D}` holds points along the spatial coordinate x_D and `xs{D+1}` holds points along the time coordinate t.

Use the variables below to coarsen data and add noise. The result of this code block will be the "observed" data `U_obs` and rewritten computational grid `xs` (both cell arrays) obtained from `U_exact` and the original `xs` by coarsening and adding noise. `pde_num` is used to select an example dataset from the examples used in the manuscript.

`%% Set Hyperparameters`

This next block of code sets the hyperparameters of the test function ψ, the subsampling rates (s_x, s_t) for convolution query points, the included terms in the model library, and the specifications for solving the sparse regression problem. All matlab variables follow the naming convention of the manuscript.

Variable	Type	Default	Use
`pde_num`	int	1 - inviscid Burgers	Selects PDE
`coarsen_data`	int array	`ones(D+1,2)` - no coarsening	coarsens data by subsampling (1st col) and truncating (2nd col)
`sigma_NR`	float	`0.0` - no added noise	noise ratio (see σ_{NR} in manuscript)
`noise_dist`	binary	`0`	0 - white noise, 1 - uniform
`noise_alg`	binary	`0`	0 - additive noise $\mathbf{U}+\epsilon$ 1 - multiplicative $\mathbf{U}(1+\epsilon)$

Each example PDE in the manuscript has its own default settings.

For the model library, terms $D^\alpha f(u)$ are identified by "tags" which are integer arrays that specify the nonlinear function f and differential operator D^α. Consider again the reaction-diffusion system with components u and v over coordinates (x_1, x_2, t). General monomial and trigonometric terms are given the following tags:

$\frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\partial t^{\alpha_3}}(u^m v^n)$	$\begin{bmatrix} m & n & \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix}$
$\frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\partial t^{\alpha_3}}\cos(ku)$	$\begin{bmatrix} ki & 0 & \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix}$
$\frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\partial t^{\alpha_3}}\sin(ku)$	$\begin{bmatrix} -ki & 0 & \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix}$
$\frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\partial t^{\alpha_3}}\cos(kv)$	$\begin{bmatrix} 0 & ki & \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix}$
$\frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\partial t^{\alpha_3}}\sin(ku)$	$\begin{bmatrix} 0 & -ki & \alpha_1 & \alpha_2 & \alpha_3 \end{bmatrix}$

For example, `[0, -2, 2, 0, 0]` denotes the term $\partial_{x_1x_1}\sin(2v)$ and `[1, 2, 0, 3, 0]` denotes $\partial_{x_2x_2x_2}(uv^2)$.

`%% Find Dynamics`

This section calls the function `wsindy_pde_fun` which takes as input the previously defined variables and outputs a large array of useful variables. Most notable are the first 5 variables: the weight vector `W`, the Gram matrix `G`, the left-hand side matrix `b`, the scale vector `M`, and the learned sparsity threshold `lambda_hat`. Solving `G X` $=$ `b` using sequential-thresholding least squares with threshold `lambda_hat` will produce `X=W./M`.

`%% Display Results`

`print_loc`	string or binary	1	0- don't print results 1- print results to terminal string - print results to string
`toggle_plot_basis_fcn`	binary	0	plot basis functions and partial derivatives
`toggle_plot_sol`	binary	0	if data has only one spatial dimension, plots data `U_obs`
`toggle_plot_loss`	binary	0	if a range of lambda values is given by `lambda`, this plots the loss function $\mathcal{L}$ for each value (defined in manuscript)

Table 4: Display results options.

References

[1] Daniel A Messenger and David M Bortz. Weak sindy for partial differential equations. arXiv preprint arXiv: 2007.02848, 2020.

Variable	Type	Default	Use
`m_x`	int	(depends on PDE)	selects spatial test function ϕ_x support size in number of grid points (m_x in manuscript)
`m_t`	int	(depends on PDE)	selects temporal test function ϕ_t support size in number of grid points (m_t in manuscript)
`s_x`	int	(depends on PDE)	subsampling rate along spatial coordinates such that every s_xth point is a query point
`s_t`	int	(depends on PDE)	subsampling rate along temporal coordinate such that every s_tth point is a query point
`phi_class`	int	1 - piecewise polynomial	selects test function class 1- piecewise poly 2 - Gaussian
`tau`	float	10^{-10}	sets the value of ϕ near edge of support (τ in manuscript)
`tauhat`	float	0 - no automatic selection	sets number of standard deviations into tail of $\hat{\phi}$ to place critical wavenumbers (changepoints) (k_x, k_t) of data (see appendix of manuscript)
`toggle_scale`	$\{0, 1, 2, \texttt{Inf}\}$	2 - bound column ℓ^2-norms	scale coordinates so that the columns of $\mathbf{G}$ are bounded by the $\\|\mathbf{U}\\|$ for selected norms (0 selects no scaling)

Table 1: Variables used to specify weak discretization.

`lambda`	float array	`10.^(linspace(-4,0,50))`	STLS threshold values. if `length(lambda)>1`, MSTLS is used (defined in manuscript)
`gamma`	float	0	Tikhonov regularization

Table 2: Variables used to tune sparse regression problem.

`max_dx`	int	6	Maximum spatial derivative in library
`max_dt`	int	1	Maximum temporal derivative in library
`polys`	int array	`0:6`	summed monomial degrees in library (e.g. uv has a summed monomial degree 2)
`trigs`	float array	`[]`	trigonometric frequencies
`use_all_pt`	binary	0	0 - only include time derivative $\partial^{\texttt{max\_dt}}t$ 1 - include all derivatives up to $\partial^{\texttt{max\_dt}}t$ and cross derivatives (e.g. ∂_{xt})
`use_cross_dx`	binary	0	0 - don't include cross derivatives 1 - do include
`true_nz_tags`	cell array	$\{$`[]`$\}$	tags for true model nonzero terms if known. Each cell contains tags for each equation
`lhs`	float matrix	(PDE dependent)	tags left-hand side of each equation in system to be discovered (e.g. to specifiy $\partial_t u$ and $\partial_t v$, set `lhs` $= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$
`custom_add`	float array	`[]`	Add additional terms (as tags) not automatically generated by `max_dx, max_dt, polys, trigs`
`custom_remove`	float array	`[]`	Remove terms (as tags) that are automatically generated by `max_dx, max_dt, polys, trigs`

Table 3: Variables used to specify model library.