ENGR520 - Spring 2025 **Due: 11:59pm Tuesday May 13^{th}**
Homework 3

Please submit your assignment as a single pdf in Gradescope. Include all code either as an appendix or as code blocks interspersed with discussion question responses. Each time you are asked to generate a SINDy model, please print the model in a way that makes it clear which parameters the model belongs with. Several relevant code examples can be found at `https://pysindy.readthedocs.io/en/stable/examples/index.html`. Please cite any code from these examples or other sources used in your submission.

Exercise 3–1: Generate data from a known *nonlinear* system, the Lotka–Volterra predator-prey model:

$$\dot{x}_1 = \alpha x_1 - \beta x_1 x_2, \quad \dot{x}_2 = \delta x_1 x_2 - \gamma x_2.$$

You can explore how $(\alpha, \beta, \gamma, \delta)$ affect the dynamics using this demonstration: `https://demonstrations.wolfram.com/PhasePortraitOfLotkaVolterraEquation/`. For this assignment, use $\alpha = \beta = \gamma = \delta = 2$

1. Collect trajectory data $\{\mathbf{x}(t_k)\}$ from an initial condition $[1.5, 1]$ with a $\Delta t = 0.01$. Numerically estimate $\dot{\mathbf{x}}(t_k)$, for example using central difference derivatives. Plot a phase portrait of your trajectory (plot x_1 vs x_2.
2. Build a library $\boldsymbol{\Theta}(\mathbf{x})$ with polynomials up to degree 2. Solve $\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\,\boldsymbol{\Xi}$ using **sequential thresholded least squares**, starting with a threshold of 0.2 and re-solving with lower thresholds in increments of 0.01 until you reach 0.15. Please print or otherwise clearly display your threshold values and models.
3. Compare your discovered $\boldsymbol{\Xi}$ with the ground-truth coefficients $(\alpha, \beta, \gamma, \delta)$. Which terms are selected? How does the threshold affect your results? Does the model slowly change as the threshold is varied, or does it change all at once?
4. Solve again, using a library with polynomial terms up to degree 3. Are you able to find a threshold that does a good job finding the correct terms with this library?
5. Add a small amount of noise to the trajectory $\mathbf{x}$, and then compute the derivative and SINDy model. Plot your noisy trajectory, to get a sense of the magnitude of the noise. How did the noise affect your model?

ENGR520 - Spring 2025 **Due: 11:59pm Tuesday May 13^{th}**
Homework 3

Exercise 3–2: This exercise will test the data requirements for identifying an accurate SINDy model for the Lorenz system, following the work of "Discovery of Nonlinear Multiscale Systems: Sampling Strategies and Embeddings", Kathleen P. Champion, Steven L. Brunton, and J. Nathan Kutz, SIAM Journal on Applied Dynamical Systems 2019 18:1, 312-333.

The equations of motion for the Lorenz system are

$$\dot{x} = \sigma(y - x) \tag{1}$$
$$\dot{y} = x(\rho - z) - y \tag{2}$$
$$\dot{z} = xy - \beta z. \tag{3}$$

We will use parameters $(\sigma, \rho, \beta) = (10, 28, 8/3)$ and initial condition $\mathbf{x}_0 = (x(0), y(0), z(0)) = (0, 1, 20)$.

First, we will use clean data without any measurement noise.

(a) Generate a long trajectory with a fine sampling rate of $\Delta t = 0.0001$; discard the first $t = 1$ of the data. Use SINDy to generate models using an increasing length of data T. At what T is it possible to identify the correct Lorenz mode? Plot the data up until T. Do the results surprise you? (Explain why)

(b) Now, repeat this experiment for different sampling rates from $t = 0.0001$ to 0.1. How does the minimum data length T change? How does the number of samples $T/\Delta t$ change?

(c) Repeat the experiment with small additive Gaussian noise on the trajectory. With noise, repeat the identification for several noise realizations to obtain an average success rate.

Exercise 3–3: Now we will revisit the Lorenz system and explore the effect of different de-noising derivative algorithms on identification from noisy trajectory data.

(a) First, generate a trajectory with $\Delta t = 0.01$ with the same parameters and initial conditions as the above exercise. Discard the first 1 time units of data, and use the next 5 time units for the trajectory (i.e., from $t = 1$ to $t = 6$).

(b) Add a small amount of Gaussian noise ($\sigma = 0.001$) to the trajectory, and estimate the derivative using several techniques: forward difference, central difference, and polynomial smoothing. Please code forward and central difference by hand and highlight and comment your code for easier grading. For polynomial smoothing, you may use the pysindy `SmoothedFiniteDifference` function.

(c) Use each of these velocity estimates to identify a SINDy model. Compare and discuss the results.

(d) Repeat for larger and smaller noise and report on the trends.

ENGR520 - Spring 2025 **Due: 11:59pm Tuesday May 13^{th}**
Homework 3

Exercise 3–4: This exercise will explore identifying a PDE using PDE-FIND based on data from the Korteweg-De Vries (KdV) system

$$u_t + u_{xxx} - 6uu_x = 0.$$

This system gives rise to coherent traveling wave solutions, known as solitons, that retain their shape as they travel at a constant wave speed, despite the nonlinearity in the system. In general, a soliton solution will take the form $u(x,t) = -\frac{c}{2}\operatorname{sech}^2\left(\frac{\sqrt{c}}{2}(x-ct)\right)$ for an arbitrary positive wave speed c. Notice that the wave speed c is linked to the amplitude and width of the wave.

(a) Generate data by initializing a single soliton solution $u(x,0) = -\frac{1}{2}\operatorname{sech}^2\left(\frac{1}{2}(x)\right)$, with a wave speed corresponding to $c = 1$ above. A spatial discretization of N = 256 with roughly 300 timesteps should produce accurate results. Try to identify a PDE model using PDE-FIND based on this data. What is the sparsest model that supports the data? Data generation hints:

- Code for generating this data can be found in the Korteweg de Vries equation webpage of the scipy cookbook. You can use this code as a starting point, with the necessary adjustments.
- This may take a little while to run. Consider starting by generating a shorter time with a rougher grid to confirm that the equation is coded correctly, then re-run at full resolution for the full time once the bugs are resolved.
- The scipy implementation assumes periodic boundaries, so ensure your initial condition is periodic. You can do this by adding an offset of (1/3) to the spatial domain of the wave.

(b) Next, use data beginning at two initial conditions. The first initial condition is the one above with $c = 1$, and the second initial condition is a soliton solution with $c = 4$. Concatenate this data and identify a PDE model using PDE-FIND. What is the sparsest model that supports the data?

(c) Discuss the two models that are identified depending on the initial data, and explain any differences.