Skolkovo Institute of Science and Technology

The internship summary

Research Intern
Margarita Selezneva

Moscow, 2020

Contents

1 Introduction

As part of the internship, the topic ”Discovering governing equations from data” was studied. This topic is relevant since there is a lot of data and they are available, data science is also developing. Extracting governing equations from data is a central challenge in many diverse areas of science and engineering, for example, in climate science, neuroscience, ecology, finance, and epidemiology.

Two articles [1], [3], supplementary materials and programs for these articles were studied. The main idea of the above articles is to use sparse regression to identify equations.

2 The problem statement

We consider nonlinear dynamical systems of the form

$$\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{x}(t)), \tag{1}$$

or a parameterized and nonlinear PDE of the general form

$$u_t = N(u, u_x, u_{xx}, \dots, x, \mu), \tag{2}$$

here $\boldsymbol{f}(\boldsymbol{x}(t))$ and $N(\cdot)$ are unknown right-hand sides. Most physical systems have only a few relevant terms that define the dynamics, making the governing equations sparse in a high-dimensional nonlinear function space. So it is assumed that the functions $\boldsymbol{f}(\boldsymbol{x}(t))$ and $N(\cdot)$ consists of only a few terms.

The sparse regression begins by first collecting a time history of the state $\boldsymbol{x}(t)$ into a matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ in the case (1) or all the spatial time series data into a single column vector $\boldsymbol{U} \in \mathbb{C}^{nm}$ in the case (2), where data is collected over m time points and n spatial locations. The derivative $\dot{\boldsymbol{X}}$ $(\boldsymbol{U}_t)$ is approximated numerically from $\boldsymbol{X}$ $(\boldsymbol{U})$. Any additional input terms to the right hand side (2) is considered in a column vector $\boldsymbol{Q} \in \mathbb{C}^{nm}$. Next step is to construct a library $\Theta(\boldsymbol{X})$ or $\Theta(\boldsymbol{U}, \boldsymbol{Q}) \in \mathbb{C}^{nm\times D}$ consisting of candidate linear and nonlinear functions of the columns of $\boldsymbol{X}$ or $\boldsymbol{U}$ and partial derivatives of $\boldsymbol{U}$ for the PDE. Each column of the library $\Theta(\boldsymbol{X})$ or $\Theta(\boldsymbol{U}, \boldsymbol{Q})$ corresponds to a specific candidate term for the governing equation; for example

$$\Theta(\boldsymbol{X}) = [1\ \boldsymbol{X}\ \boldsymbol{X}^{P_2}\boldsymbol{X}^{P_3}\ \dots \sin(\boldsymbol{X})\ \cos(\boldsymbol{X})],$$
$$\Theta(\boldsymbol{U}, \boldsymbol{Q}) = [1\ \boldsymbol{U}\ \boldsymbol{U}^2 \dots \boldsymbol{Q} \dots \boldsymbol{U}_x\ \boldsymbol{U}\boldsymbol{U}_x \dots],$$

the ODE and PDE evolution can be expressed in this library as follows

$$\boxed{\boldsymbol{U}_t = \Theta(\boldsymbol{U}, \boldsymbol{Q})\xi, \quad \dot{\boldsymbol{X}} = \Theta(\boldsymbol{X})\Xi,}$$

here $\Xi = [\xi_1\ \xi_2\ \dots \xi_n]$.

Each nonzero entry in ξ (ξ_i) corresponds to a term in the equation, the vector ξ (ξ_i) is sparse, meaning that only a few terms are active. Sparse regression is utilized to find a solution ξ

$$\xi = \underset{\hat{\xi}}{argmin}\|\Theta\hat{\xi} - U_t\|_2^2 + \eta\|\hat{\xi}\|_0, \quad \eta = 10^{-3}k(\Theta) \tag{3}$$

where $k(\Theta)$ is the condition number of the matrix Θ, and $\|\cdot\|_0$ is l_0-norm. The l_0 term makes this problem np-hard. Therefore, Sequential Threshold Ridge regression (STRidge) were utilized for approximating solutions to (3)

$$\xi = \underset{\hat{\xi}}{argmin}\|\Theta\hat{\xi} - U_t\|_2^2 + \lambda\|\hat{\xi}\|_2^2 = (\Theta^T\Theta + \lambda I)^{-1}\Theta^T U_t, \tag{4}$$

for $\lambda = 0$ this reduces to Sequentially Thresholded Least Squares (STLS). The solution ξ is found recursively in a loop based on (4) and its components, which are less than a certain threshold (tol), vanish. Experiments have shown, that the regression coefficients strongly depend on this threshold.

3 Experiments

STLS is used for the Lorentz system

$$\begin{cases} \dot{x} = \sigma(y - x), \\ \dot{y} = x(\rho - z) - y, \\ \dot{z} = xy - \beta z \end{cases}$$

with the standard parameters $\sigma = 10$, $\beta = 8/3$, $\rho = 28$ and with an initial condition $(x_0,\, y_0,\, z_0)^T = (-8,\, 7,\, 27)^T$. Data is collected from numerical simulations for $t \in [0, 100]$ with a time-step of $\Delta t = 0.001$. The system is identified in the space of polynomials in (x, y, z) up to fifth order:

$$\Theta(\boldsymbol{X}) = [\boldsymbol{x}(t) \quad \boldsymbol{y}(t) \quad \boldsymbol{z}(t) \quad \boldsymbol{x}^2(t) \quad \boldsymbol{x}(t)\boldsymbol{y}(t) \quad \boldsymbol{x}(t)\boldsymbol{z}(t) \quad \boldsymbol{y}^2(t) \quad \ldots \quad \boldsymbol{z}^5(t)]$$

With clean data, the derivatives are taken via the second order finite differences and it is enough to use the first 1000 points for each coordinate. The error in determining the coefficients is $0.19 \pm 0.31\%$ for $tol = 0.125$.

$$\begin{cases} \dot{x} = 10.002y - 10.002x, \\ \dot{y} = -0.991y + 27.955x - 0.998xz, \\ \dot{z} = -2.667z + 0.999xy \end{cases}$$

The error does not decrease when adding data. If all data is used the error is $0.33 \pm 0.61\%$. The error becomes greater in the equation for the variable y

$$\dot{y} = -0.982y + 27.939x - 0.998xz \quad (\mathit{tol} = 0.01).$$

It was discovered that the equation for y is the most difficult to identify. It can be incorrectly identified at $\mathit{tol} = 0.5$

$$\dot{y} = 9.947y - 12.928x.$$

The system with noise is also considered

$$\hat{\boldsymbol{X}} = \boldsymbol{X} + \eta\boldsymbol{Z}, \quad \boldsymbol{X} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \quad \boldsymbol{Z} \sim N(0,1).$$

In experiments, $\eta = 1$ and using finite differences leads to the large coefficient error in the equations

$$\begin{cases} \dot{x} = 9.108y - 8.966x, (\mathit{tol} = 0.7) \\ \dot{y} = 0.251y + 23.946x - 0.908xz, (\mathit{tol} = 0.2) \\ \dot{z} = -2.601z + 0.976xy, (\mathit{tol} = 0.7) \end{cases}$$

The error is $24.4 \pm 41.4\%$. Different thresholds (tol) are used to identify the equations.

Polynomial interpolation works worse on noisy data. For each point, except for boundary points, a polynomial of degree 4 is constructed on 19 points, and derivatives of the polynomial are taken to approximate those of the numerical data.

$$\begin{cases} \dot{x} = 9.015y - 8.875x, (\mathit{tol} = 0.22) \\ \dot{y} = 24.480x - 0.917xz, (\mathit{tol} = 0.5) \\ \dot{z} = -2.595z + 0.975xy, (\mathit{tol} = 0.8) \end{cases}$$

Depending on tol, the equation for y changes

$$\dot{y} = 0.661 + 3.785y + 19.657x - 0.127yz - 0.768xz, (\mathit{tol} = 0.12)$$
$$\dot{y} = 0.298 + 24.480x - 0.917xz, (\mathit{tol} = 0.13)$$

Figures of the attractor are below (see Figure 2-4). Despite the different equations, the structure of attractors is the same.

If interpolation of data is used instead of initial noisy those and derivatives are calculated via the finite differences, then the correct terms are identified

$$\begin{cases} \dot{x} = 9.786y - 9.757x, (\mathit{tol} = 0.125) \\ \dot{y} = -0.758y + 27.213x - 0.981xz, (\mathit{tol} = 0.125) \\ \dot{z} = -2.65z + 0.994xy, (\mathit{tol} = 0.125) \end{cases}$$

The error is $4.9 \pm 7.9\%$.

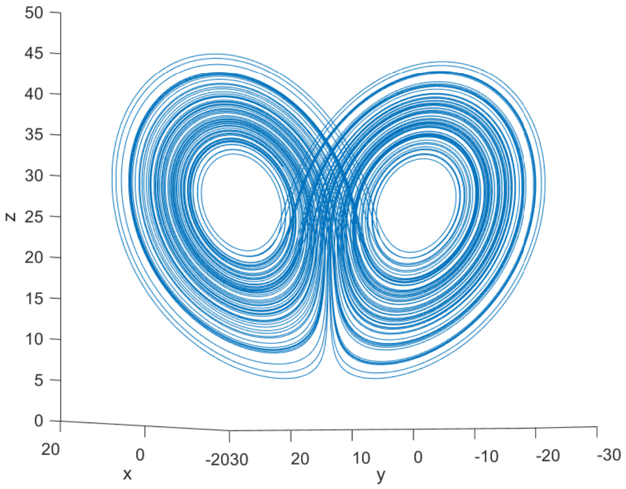

Figure 1: $\dot{y} = 28x - xz - y$ (the right attractor)

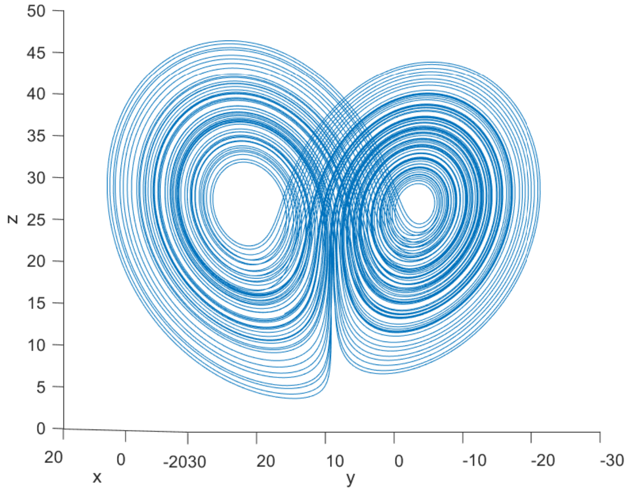

Figure 2: $\dot{y} = 24.480x - 0.917xz$

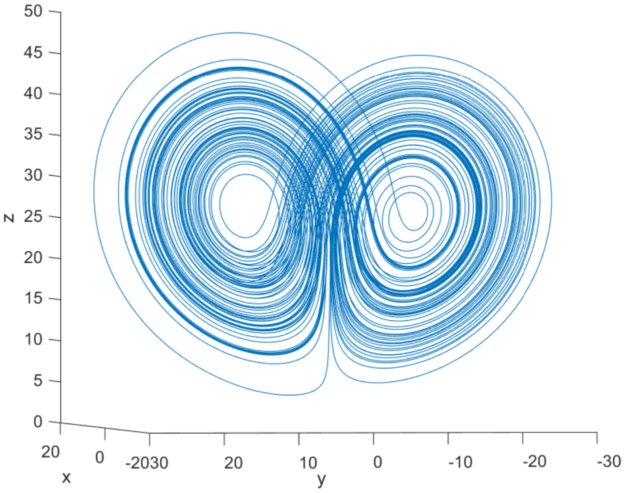

Figure 3: $\dot{y} = 0.661 + 3.785y + 19.657x - 0.127yz - 0.768xz$

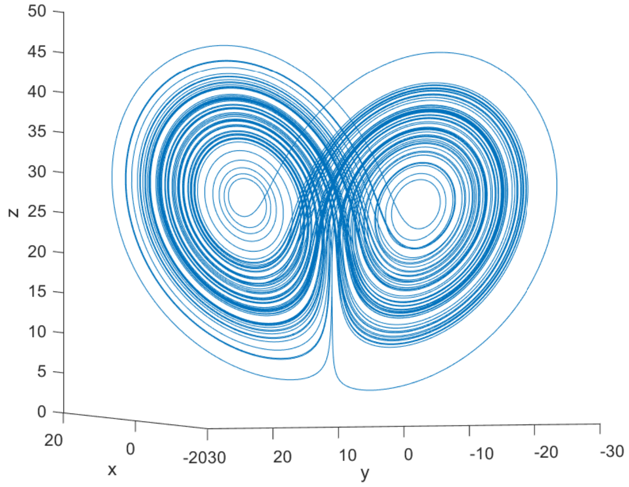

Figure 4: $\dot{y} = 0.298 + 24.480x - 0.917xz$

Tikhonov differentiation is another method for calculating derivatives with noisy data.

$$\hat{f}' = \underset{g}{argmin} \|Ag - f\|_2^2 + \lambda \|Dg\|_2^2 = (A^T A + \lambda D^T D)^{-1} A^T f,$$

where

$$A = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 \\ \frac{dx}{2} & \frac{dx}{2} & 0 & \dots & 0 \\ \frac{dx}{2} & dx & \frac{dx}{2} & 0 & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{dx}{2} & dx & \dots & dx & \frac{dx}{2} \end{pmatrix}, D = \begin{pmatrix} -\frac{1}{dx} & \frac{1}{dx} & 0 & 0 & \dots & 0 \\ 0 & -\frac{1}{dx} & \frac{1}{dx} & 0 & 0 & \vdots \\ 0 & 0 & -\frac{1}{dx} & \frac{1}{dx} & 0 & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \vdots & 0 & -\frac{1}{dx} & \frac{1}{dx} \end{pmatrix},$$

f is the vector of the noisy coordinate at all time points and g is its numerical derivative.

The implementation of the method is attached to the supplementary materials for the article [3]. However, this implementation has a disadvantage: its numerical derivative deviates from a derivative approximation at the ends of the segment (see Figure 5). Finite differences on clean data are used as the derivative approximation (FD(clean data) in Figure 5-6). Own program was written through the conjugate gradient method. As a result, the problem with the ends was eliminated (see Figure 6).

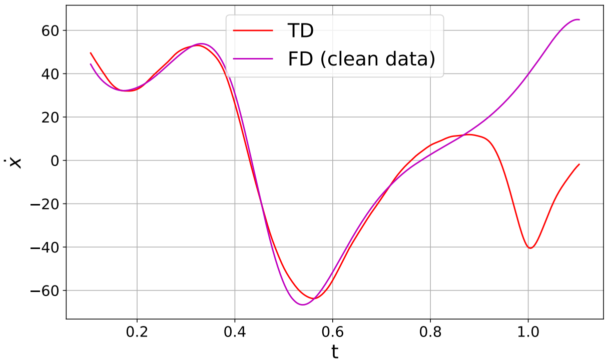

Figure 5: It is based on the method of least squares

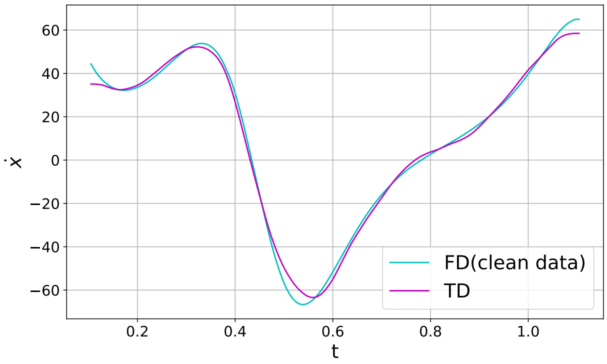

Figure 6: It is based on the conjugate gradient method

Large matrices appear because of a lot of data. They do not fit into my computer's memory. Therefore, the data is divided into subsamples and each of them is differentiated via Tikhonov regularization. Then the derivatives and the data are combined and STLS is applied to them. The best result is obtained using all data i.e. data for each coordinate is split into 100 parts of 1000 points

$$\begin{cases} \dot{x} = 9.365y - 9.396x, (tol = 0.5) \\ \dot{y} = 0.502y + 20.929x - 0.808xz, (tol = 0.125) \\ \dot{z} = -2.433z + 0.909xy, (tol = 0.5) \end{cases}$$

All terms are defined correctly, but the coefficient of the variable y has a different sign in the second equation.

Own program was supposed to be applied to all data at once. However, smoothing the data and then applying finite differences is more effective. This runs in several seconds for all data.

$$\hat{f}' = \underset{g}{argmin} \|g - f\|_2^2 + \lambda \|Dg\|_2^2 = (E + \lambda D^T D)^{-1} f,$$

here we find a vector g that approximates the initial noisy data f, and D is the same matrix as above.

Figures 7-9 show that the smoothed data fit well with the clean data.

The equations are identified with an error equal to $1.98 \pm 3.53\%$ for $\lambda = 10^{-4}$

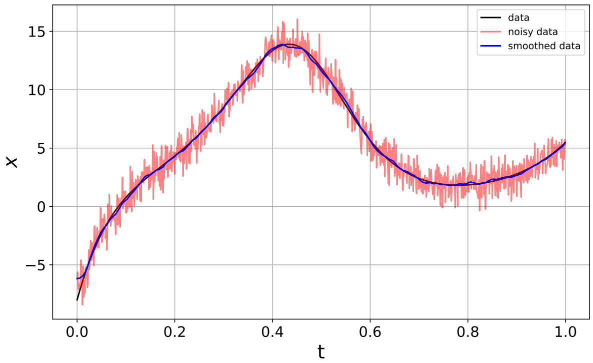

Figure 7: $x(t)$

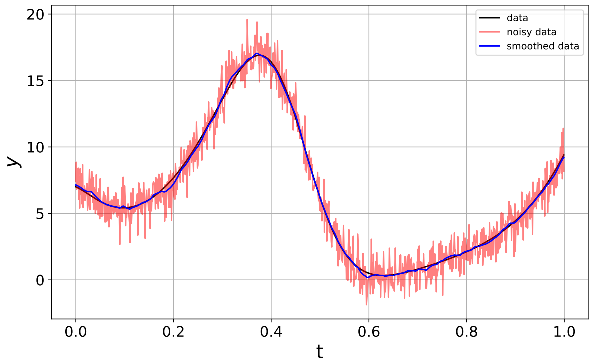

Figure 8: $y(t)$

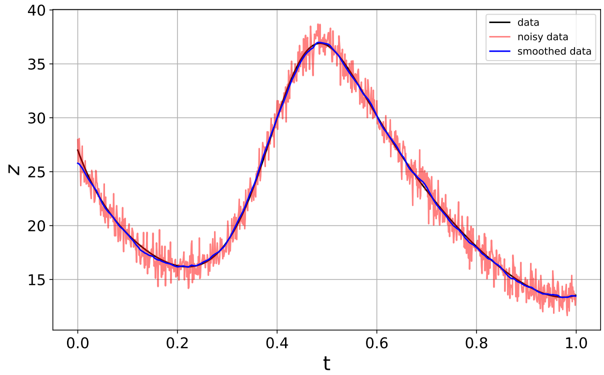

Figure 9: $z(t)$

$$\begin{cases} \dot{x} = 9.96y - 9.955x \\ \dot{y} = -0.894y + 27.601x - 0.992xz \\ \dot{z} = -2.646z + 0.999xy. \end{cases}$$

4 Fluid wake behind a cylinder (POD)

Many real-world systems governed by PDEs, such as weather and climate, are characterized by big data. However, many of these high-dimensional systems evolve on a low-dimensional attractor, making the effective dimension much smaller. The low-Reynolds number flow past a cylinder is an example of such a system.

The two-dimensional flow is described in a Cartesian coordinate system, x, y. The origin is in the centre of the cylinder which has diameter D. Location is denoted by a vector $\boldsymbol{x} = (x, y)$, and time by t. The velocity vector is $u = (u, v)$, where u and v are the components in the $x-$ and $y-$direction, respectively. Pressure is denoted by p. In the following, all variables are assumed to be non-dimensionalized with respect to the cylinder diameter D and the oncoming flow U [2].

The Navier-Stokes equations and the incompressibility condition describe the evolution of the flow

$$\begin{aligned}&\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla p + \frac{1}{Re}\nabla^2 \boldsymbol{u},\\ &\nabla \cdot \boldsymbol{u} = 0,\end{aligned}$$

where $Re = \dfrac{UD}{\nu}$ represents the Reynolds number with kinematic viscosity ν. The flow u is approximated by a finite Galerkin approximation $u^{[N]}$

$$\boldsymbol{u}(\boldsymbol{x}, t) \approx \boldsymbol{u}^{[N]} := \boldsymbol{u}_0(\boldsymbol{x}) + \sum_{i=1}^{N} a_i(t)\boldsymbol{u}_i(t), \tag{5}$$

$\boldsymbol{u}_0$ represents the mean flow, $\{\boldsymbol{u}_i\}_{i=1}^{N}$ the first N Karhunen-Loève modes, and a_i the time-dependent Fourier coefficients: $a_i = \langle \boldsymbol{u} - \boldsymbol{u}_0, \boldsymbol{u}_i \rangle_\Omega$, where $\langle \boldsymbol{v}, \boldsymbol{w} \rangle_\Omega$ represents the inner product between two solenoidal fields $\boldsymbol{v}$, $\boldsymbol{w}$ on the computational domain Ω.

The Karhunen–Loève modes are computed with a snapshot method. It should be noted 20 snapshots are sufficient for the construction of the first eight eigenmodes. However, the fixed point of the Galerkin model (5) is the averaged flow $\boldsymbol{u}_0$ as opposed to the correct steady Navier–Stokes solution $\boldsymbol{u}_s$.

A natural extension of the Galerkin approximation for transient flow is the inclusion of an additional vector $\boldsymbol{u}_\triangle$, called the shift-mode. It represents the 'shift' of the short-term averaged flow away from the Karhunen–Loève space. The new phase-space direction is constructed in the following Gram–Schmidt procedure starting from the mean-field correction $\boldsymbol{u}_0 - \boldsymbol{u}_s$:

$$\begin{aligned}\boldsymbol{u}_\triangle^a &:= \boldsymbol{u}_0 - \boldsymbol{u}_s,\\ \boldsymbol{u}_\triangle^b &:= \boldsymbol{u}_\triangle^a - \sum_{i=1}^{N} \langle \boldsymbol{u}_\triangle^a, \boldsymbol{u}_i \rangle_\Omega \boldsymbol{u}_i,\\ \boldsymbol{u}_\triangle &:= \frac{\boldsymbol{u}_\triangle^b}{\|\boldsymbol{u}_\triangle^b\|_\Omega},\end{aligned}$$

This shift-mode can formally be considered as the $(N+1)th$ expansion mode $\boldsymbol{u}_{N+1} := \boldsymbol{u}_\triangle$ in a generalized Karhunen–Loeve decomposition and $\{\boldsymbol{u}_i\}_{i=1}^{N}$ remains an orthonormal system by construction. The generalized Karhunen–Loeve decomposition is expressed by

$$\boldsymbol{u}(\boldsymbol{x}, t) \approx \boldsymbol{u}^{[N+1]} := \sum_{i=0}^{N+1} a_i(t)\boldsymbol{u}_i(\boldsymbol{x}), \tag{6}$$

where $a_0 := 1$.

The Galerkin system is derived from (6) with a standard Galerkin projection on the Navier–Stokes equation

$$\frac{d}{dt}a_i = \frac{1}{Re}\sum_{j=0}^{N+1} l_{ij}a_j + \sum_{j,k=0}^{N+1} q_{ijk}a_j a_k \quad \text{for} \quad i = 1, \dots, N+1 \tag{7}$$

with coefficients $l_{ij} := \langle \boldsymbol{u}_i, \nabla^2 \boldsymbol{u}_j \rangle_\Omega$ and $q_{ijk} := \langle \boldsymbol{u}_i, (\boldsymbol{u}_j \cdot \nabla)\boldsymbol{u}_k \rangle_\Omega$. The Galerkin projection of the pressure term is found to be negligible and omitted.

At $N = 2$ the decomposition (6) is the minimal generalized Galerkin model

$$\boldsymbol{u} = \boldsymbol{u}_0 + a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2 + a_\triangle \boldsymbol{u}_\triangle. \tag{8}$$

In mean-field theory, the steady solution is the basic mode $\boldsymbol{u}_0$, the oscillation modes $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ are (essentially) the real and imaginary part of the associated most unstable eigenmode, and the shift-mode $\boldsymbol{u}_\triangle$ is derived from a linearized Reynolds equation [2].

In the mean-field model, the Galerkin projection on the Navier–Stokes equation leads to

$$\begin{cases} \dot{x} = \mu y - \omega x + Axy, \\ \dot{y} = \omega x + \mu y + Ayz, \\ \dot{z} = -\lambda(z - x^2 - y^2) \end{cases}$$

STLS is also applied for the identification of the system. Despite the trajectory of the identified system is similar to the correct one, the authors of the article [1] could not correctly identify the equations. Perhaps this is due to the fact that POD-Galerkin systems are often unstable, and analyzing these nonlinear systems is difficult.

5 TrainSTRidge

In the article [3], the authors propose the complicated algorithm TrainSTRidge. The data is split into training and testing sets in a ratio of 80 to 20

$$\Theta \longrightarrow [\Theta^{train}, \Theta^{test}], \quad U_t \longrightarrow [U_t^{train}, U_t^{test}].$$

The algorithm STRidge finds the solution ξ for training data at the fixed threshold *tol*, and outside STRidge, an error is calculated by

$$error = \|\Theta^{test}\xi - U_t^{test}\|_2^2 + \eta\|\xi\|_0, \quad \eta = 10^{-3}k(\Theta).$$

If the current error is less than the previous one, then the threshold (*tol*) increases, otherwise, it decreases. This happens a given number of times. As a result, the solution ξ with the least error is assumed to be correct.

Despite the improvement in the algorithm, it has the disadvantage. There is still a dependence on the threshold. Consider Korteweg-de Vries equation

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

tol_{opt} represents the threshold at which the least error is received. If we put $tol = 6$ in TrainSTRidge, then obtain identified PDE

$$u_t + 5.957uu_x + 0.988u_{xxx} = 0$$

with $error = 19.61$ and $tol_{opt} = 6$.

At $tol = 7$ TrainSTRidge returns

$$u_t + 7.242u^2u_x = 0$$

with $error = 13.54$ and $tol_{opt} = 14$. In this case, we get an equation of a completely different kind. If we do not know the answer in advance, we will not understand which equation should be.

6 Conclusion

Defining models using data is a beautiful idea, but this method does not fully implement it. Besides the explicit dependence on the threshold tol, other difficulties arise. First, we need to understand what functions should be included in the library Θ, which means there should be some assumptions about the form of the right side of the equation. The supplementary materials to the article [3] provide equations that are obtained when the necessary functions are not in the library Θ.

Second, the examples given in the articles [1], [3] have simple non-linearity in the form of a combination of monomials. The question arises whether the method will work if the nonlinearity is exponential or trigonometric.

Third, sufficient data should be used to identify equations, and the data should include important process characteristics. These problems are reflected in the article [3]. The article describes how the error in identifying the diffusion equation changes depending on the amount of data and that the Korteweg-de Vries equation becomes indistinguishable from the one-way wave equation if a single propagating soliton is included in the data.

Fourth, noise in the data increases the error in identifying equations. However, there are different tools to fix this problem, one of which is data smoothing.

The algorithm STRidge can be interpreted as follows. We build a sequence to converge to the sparse solution. However, the choice of the threshold determines the sequence, and therefore the final result. The applicability of the method is in question since the method does not guarantee the correct answer.

References

[1] *S. L. Brunton, J. L. Proctor, J. N. Kutz* Discovering governing equations from data by sparse identification of nonlinear dynamical systems, PNAS, vol. 113, 3932-3937 (2016).

[2] *B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, F. Thiele* A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. J Fluid Mech, vol. 497, 335–363 (2003).

[3] *S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz* Data-driven discovery of partial differential equations, Science Advances, vol. 3 (2017).